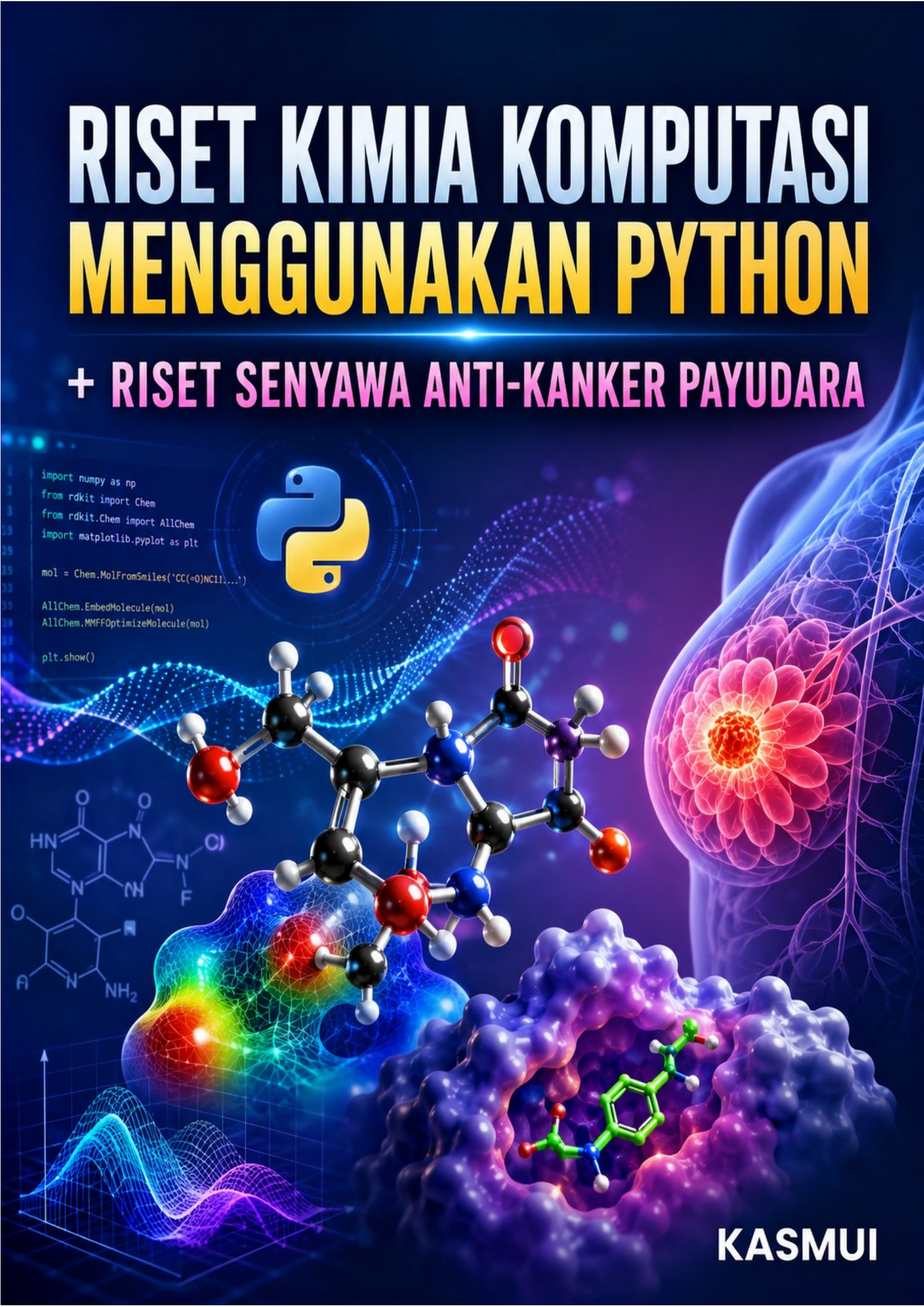
RISET KIMIA KOMPUTASI MENGGUNAKAN PYTHON

+ RISET SENYAWA ANTI-KANKER PAYUDARA



KASMUI

RISET KIMIA KOMPUTASI MENGGUNAKAN PYTHON

UNTUK SENYAWA ANTI-KANKER PAYUDARA

Format Buku Akademik, Panduan Praktikum, Proposal Riset, dan Implementasi Python/PySCF

Disusun dan dikembangkan sebagai buku ajar riset kimia komputasi yang mempertahankan seluruh naskah sumber, lalu menambahkan penataan bab, pengantar, penjelasan metodologis, panduan kerja, check-list riset, dan lampiran pendukung agar dapat dibaca sebagai satu dokumen buku yang runtut.

Penulis/Pengembang: Kasmui

Bidang: Kimia Komputasi, Cheminformatics, Molecular Docking, DFT, QSAR, dan Python untuk Riset Senyawa Anti-Kanker Payudara.

Cek link

https://kasmui.cloud/aibuku/index.html#komputasi

KATA PENGANTAR

Puji syukur ke hadirat Allah Swt. atas limpahan rahmat, taufik, hidayah, dan inayah-Nya sehingga buku **"RISET KIMIA KOMPUTASI MENGGUNAKAN PYTHON + RISET SENYAWA ANTI-KANKER PAYUDARA"** ini dapat disusun sebagai salah satu ikhtiar akademik untuk memperkuat pembelajaran, praktikum, dan riset kimia komputasi di lingkungan pendidikan tinggi, khususnya bagi mahasiswa program sarjana kimia dan bidang terkait.

Kimia komputasi telah berkembang sangat pesat. Jika pada masa awalnya kimia komputasi lebih banyak dipahami sebagai alat bantu visualisasi struktur molekul, saat ini bidang tersebut telah menjadi bagian penting dari riset modern yang menghubungkan teori kimia, fisika kuantum, biologi molekuler, farmasi, ilmu data, kecerdasan buatan, dan pemrograman ilmiah. Perkembangan perangkat lunak, metode komputasi, serta ketersediaan library Python telah membuka peluang besar bagi mahasiswa, dosen, dan peneliti pemula untuk melakukan eksplorasi molekul secara lebih sistematis, efisien, dan terukur.

Buku ini disusun untuk menjawab kebutuhan tersebut. Di satu sisi, mahasiswa perlu memahami dasar-dasar kimia komputasi, mulai dari representasi molekul, optimasi geometri, energi elektronik, orbital molekul, descriptor molekul, docking, hingga prediksi ADMET. Di sisi lain, mahasiswa juga perlu menguasai keterampilan teknis dalam menggunakan Python sebagai bahasa pemrograman ilmiah yang fleksibel dan banyak digunakan dalam riset modern. Karena itu, buku ini tidak hanya menyajikan konsep, tetapi juga memperlihatkan bagaimana konsep tersebut diterjemahkan ke dalam alur kerja riset yang nyata.

Salah satu kekuatan utama buku ini adalah penekanannya pada integrasi antara **kimia komputasi** dan **Python**. Python tidak diperlakukan sekadar sebagai bahasa pemrograman, melainkan sebagai ekosistem riset yang memungkinkan pembaca membangun, membaca, memodifikasi, menganalisis, dan mendokumentasikan data molekul secara lebih terbuka. Library seperti **PySCF, RDKit, Open Babel, ASE, Psi4, cclib, py3Dmol, scikit-learn**, serta berbagai perangkat pendukung lain dibahas dalam kerangka penggunaannya untuk riset kimia komputasi. Dengan pendekatan ini, pembaca diharapkan tidak hanya mengetahui nama library, tetapi juga memahami fungsi ilmiah, posisi metodologis, kelebihan, keterbatasan, serta kemungkinan integrasinya dalam satu pipeline penelitian.

Tema khusus yang diangkat dalam buku ini adalah **riset senyawa anti-kanker payudara**. Tema ini dipilih karena relevan secara ilmiah, penting secara medis, dan menarik untuk pembelajaran kimia komputasi. Namun demikian, buku ini menegaskan bahwa hasil komputasi tidak boleh dipahami sebagai klaim terapi langsung. Perhitungan in silico, docking, prediksi ADMET, dan analisis descriptor molekul harus dibaca sebagai pendekatan awal untuk membangun hipotesis ilmiah. Validasi eksperimental tetap diperlukan sebelum suatu senyawa dapat dinilai memiliki potensi biologis yang kuat. Dengan demikian, buku ini berusaha menanamkan sikap ilmiah yang hati-hati, jujur, dan bertanggung jawab dalam menafsirkan hasil komputasi.

Buku ini juga disusun dengan orientasi praktis bagi mahasiswa S1 yang sedang atau akan menyiapkan proposal skripsi di bidang kimia komputasi. Banyak mahasiswa tertarik pada riset komputasi, tetapi sering mengalami kesulitan ketika harus merumuskan judul, menyusun rumusan masalah, menentukan metode, memilih senyawa, menyiapkan struktur molekul, menjalankan perhitungan, membaca output, dan menyusun pembahasan ilmiah. Oleh karena itu, buku ini menyediakan alur kerja yang lebih runtut, mulai dari pemilihan topik, penyusunan proposal, pemilihan metode, perhitungan DFT, docking molekuler, prediksi ADMET, analisis data, hingga penyusunan laporan.

Dalam konteks praktikum, buku ini dapat digunakan sebagai panduan bertahap. Pembaca dapat memulai dari pengenalan ekosistem Python, instalasi environment, pengenalan format molekul seperti SMILES, XYZ, SDF, dan PDB, kemudian masuk ke generasi konformer, optimasi awal struktur, perhitungan DFT menggunakan PySCF, analisis HOMO-LUMO, momen dipol, energi total, visualisasi molekul, docking, hingga interpretasi hasil. Dengan demikian, buku ini dapat berfungsi sebagai buku ajar, modul praktikum, sekaligus referensi awal untuk riset skripsi.

Penekanan khusus juga diberikan pada aspek **reproducibility** atau keterulangan riset. Dalam riset komputasi, hasil yang baik bukan hanya hasil yang tampak menarik, tetapi hasil yang dapat dilacak, diulang, diperiksa, dan dijelaskan kembali. Oleh karena itu, pembaca didorong untuk mencatat versi software, library, basis set, metode, parameter, struktur input, file output, log running, serta asumsi-asumsi yang digunakan dalam perhitungan. Kesadaran terhadap reproducibility sangat penting agar riset mahasiswa tidak berhenti pada hasil numerik, tetapi berkembang menjadi laporan ilmiah yang dapat dipertanggungjawabkan.

Buku ini juga berusaha menjembatani tiga kompetensi penting yang sering berjalan terpisah, yaitu **penguasaan konsep kimia**, **kemampuan pemrograman ilmiah**, dan **keterampilan menyusun riset akademik**. Dalam praktiknya, riset kimia komputasi menuntut ketiganya hadir secara bersamaan. Mahasiswa tidak cukup hanya dapat menjalankan program. Mahasiswa juga harus memahami makna hasil perhitungan. Sebaliknya, pemahaman konsep kimia yang baik akan lebih kuat jika didukung kemampuan mengolah data, menulis kode, membuat visualisasi, dan menyusun interpretasi berbasis bukti.

Harapan utama dari penyusunan buku ini adalah agar pembaca mampu melihat kimia komputasi sebagai bidang yang terbuka, menarik, dan dapat dipelajari secara bertahap. Mahasiswa tidak harus langsung menguasai semua metode komputasi tingkat lanjut. Yang lebih penting adalah membangun fondasi yang benar: memahami masalah ilmiah, memilih metode yang sesuai, menggunakan software secara bertanggung jawab, membaca hasil dengan kritis, serta menghindari klaim yang berlebihan. Dengan fondasi tersebut, mahasiswa dapat mengembangkan riset sederhana menjadi kajian yang bermakna dan layak dipresentasikan secara akademik.

Buku ini tentu masih memiliki keterbatasan. Dunia kimia komputasi, cheminformatics, machine learning chemistry, molecular docking, molecular dynamics, dan artificial intelligence dalam desain obat terus berkembang sangat cepat. Library Python yang hari ini populer dapat berubah, diperbarui, atau digantikan oleh pendekatan baru. Karena itu, buku ini sebaiknya dibaca sebagai fondasi dan panduan kerja awal, bukan sebagai batas akhir pengetahuan. Pembaca tetap dianjurkan untuk merujuk dokumentasi resmi, artikel ilmiah terbaru, serta sumber akademik yang relevan ketika akan melakukan penelitian lebih lanjut.

Penulis menyampaikan terima kasih kepada semua pihak yang telah memberikan inspirasi, dorongan, kritik, dan masukan dalam penyusunan buku ini. Semoga buku ini dapat memberikan manfaat bagi mahasiswa, dosen pembimbing, peneliti pemula, praktisi pendidikan kimia, serta siapa saja yang ingin mempelajari riset kimia komputasi berbasis Python. Semoga buku ini juga dapat menjadi salah satu kontribusi kecil dalam memperkuat budaya riset ilmiah yang terbuka, terukur, kritis, dan bertanggung jawab.

Akhirnya, semoga buku ini menjadi jembatan antara teori dan praktik, antara kimia dan pemrograman, antara pembelajaran dan penelitian, serta antara rasa ingin tahu ilmiah dan tanggung jawab akademik. Riset komputasi yang baik bukan hanya ditentukan oleh kecanggihan software, tetapi oleh ketepatan pertanyaan penelitian, kesesuaian metode, kerapian data, ketelitian analisis, kejujuran interpretasi, dan kesadaran bahwa setiap hasil ilmiah harus ditempatkan dalam batas-batas metodologisnya.

PETUNJUK PENGGUNAAN BUKU

Untuk mahasiswa pemula: bacalah Bab 1 terlebih dahulu untuk memahami ekosistem library Python, kemudian lanjutkan ke bab senyawa dan judul riset. Jangan langsung menjalankan skrip sebelum memahami input, metode, basis set, dan parameter yang digunakan.

Untuk mahasiswa yang sedang menyiapkan proposal: mulailah dari bagian judul riset, proposal, rumusan masalah, tujuan, hipotesis, batasan, desain data, dan jadwal penelitian. Gunakan lampiran check-list untuk memastikan proposal memiliki alur yang lengkap.

Untuk pengguna yang fokus pada pemrograman: perhatikan bagian kode PySCF secara bertahap. Baca konfigurasi default, koordinat XYZ, fungsi pembaca struktur, resolusi metode dan basis set, kalkulator, analisis struktur, penulisan hasil, dan argumen terminal.

Untuk dosen pembimbing: gunakan buku ini sebagai basis diskusi tentang kelayakan topik, keterjangkauan komputasi, validasi docking, pemilihan metode DFT, keterbatasan prediksi ADMET, dan kualitas interpretasi kimia.

Untuk praktikum: bagilah materi ke dalam pertemuan: instalasi, RDKit/Open Babel, struktur molekul, optimasi awal, DFT/PySCF, docking, descriptor, ADMET, analisis data, dan penyusunan laporan.

CAPAIAN PEMBELAJARAN

Setelah mempelajari buku ini, pembaca diharapkan mampu menjelaskan peran Python dalam kimia komputasi modern, termasuk posisi PySCF, Psi4, RDKit, Open Babel, ASE, OpenMM, MDAnalysis, pymatgen, cclib, dan py3Dmol dalam workflow riset molekul dan material.

1) Pembaca mampu membedakan fungsi library untuk kimia kuantum, cheminformatics, dinamika molekul, material science, machine learning, workflow high-throughput, visualisasi, serta quantum computing chemistry.
2) Pembaca mampu merumuskan topik riset senyawa anti-kanker payudara yang realistis untuk level S1 dengan mengintegrasikan DFT, docking, descriptor molekul, drug-likeness, ADMET, serta analisis keterkaitan struktur-aktivitas.
3) Pembaca mampu menyiapkan proposal riset komputasi yang memiliki latar belakang, rumusan masalah, tujuan, hipotesis, kebaruan, batasan, manfaat, metode, desain data, rencana jadwal, luaran, risiko, dan strategi mitigasi.
4) Pembaca mampu membaca, memodifikasi, dan menjalankan skrip Python berbasis PySCF untuk perhitungan energi, optimasi geometri, analisis orbital, dan pelaporan hasil komputasi molekul hibrida.
5) Pembaca mampu menafsirkan hasil komputasi secara hati-hati dengan membedakan antara prediksi in silico, dugaan mekanistik, dan klaim biologis yang memerlukan validasi eksperimental.

DAFTAR ISI

DAFTAR ISI RINGKAS

PETA ALUR RISET DALAM BUKU

Alur riset dalam buku ini dapat dibaca sebagai rantai kerja: pemilihan masalah biologis, penentuan target molekuler, pemilihan senyawa induk, desain hibrida, penyiapan struktur, optimasi geometri, perhitungan DFT, docking, prediksi ADMET, analisis data, pemilihan kandidat, dan penulisan laporan.

Pada tahap awal, pembaca perlu memahami bahwa setiap library Python memiliki posisi tertentu. RDKit dan Open Babel kuat pada representasi molekul dan format file; PySCF dan Psi4 berfungsi untuk kimia kuantum; ASE membantu workflow atomistik; OpenMM dan MDAnalysis mendukung dinamika molekul; sedangkan scikit-learn, DeepChem, DScribe, dan model ML atomistik membuka ruang riset berbasis data.

Pada tahap desain riset, senyawa anti-kanker payudara tidak diperlakukan sebagai klaim terapi langsung. Senyawa-senyawa tersebut dipakai sebagai model molekul kecil untuk mengajarkan pemetaan target, pembacaan SMILES, perbandingan struktur, dan pemilihan descriptor. Setiap hasil komputasi harus dibaca sebagai hipotesis ilmiah yang masih perlu validasi lebih lanjut.

Pada tahap implementasi, skrip PySCF di dalam naskah menjadi contoh penting karena memperlihatkan bagaimana file riset dapat dibuat lengkap: identitas molekul, koordinat XYZ, metode, basis set, fungsi validasi, perhitungan, analisis, output, dan log running. Format seperti ini sangat cocok untuk pembelajaran skripsi karena mahasiswa dapat melihat hubungan antara proposal dan implementasi program.

BAB 1

EKOSISTEM PYTHON UNTUK RISET KIMIA KOMPUTASI

Pengantar Bab

Bab ini menjadi pintu masuk untuk memahami ekosistem Python dalam riset kimia komputasi. Pembaca diajak melihat bahwa library bukan sekadar daftar perangkat lunak, melainkan perangkat metodologis yang menentukan bagaimana struktur molekul dibangun, energi dihitung, data dianalisis, dan hasil riset dilaporkan.

Pengantar ini menempatkan PySCF, Psi4, RDKit, Open Babel, ASE, OpenMM, MDAnalysis, pymatgen, cclib, py3Dmol, dan library lain ke dalam alur kerja yang utuh. Dengan demikian, mahasiswa tidak hanya menghafal nama paket, tetapi memahami fungsi ilmiah, batas penggunaan, dan hubungan antar-library dalam satu pipeline penelitian.

Setelah pengantar ini, pembahasan dilanjutkan dengan orientasi bab, peta kompetensi, uraian materi inti, contoh kerja, dan catatan metodologis yang diperlukan untuk menuntun pembaca dari konsep menuju praktik.

Pembaca diharapkan mampu mengelompokkan library berdasarkan bidang utama, memilih paket minimal untuk praktikum, menyiapkan environment Conda yang stabil, serta memahami risiko konflik dependency ketika semua paket dipaksa masuk ke satu environment.

Peta Kompetensi Bab 1

Kunci pembacaan bab ini adalah memahami hubungan antara konsep kimia dan alat komputasi. Library bukan tujuan akhir, melainkan jembatan untuk menjawab pertanyaan ilmiah. Karena itu, daftar paket di bawah sebaiknya dibaca bersama konsep yang didukung, kegunaan rinci, dan posisi paket dalam workflow riset.

Bab ini menjadi fondasi buku karena pembaca diajak mengenali peralatan utama sebelum masuk ke rancangan riset. Dalam riset kimia komputasi, pemilihan library tidak boleh asal banyak. Setiap paket harus dipilih berdasarkan fungsi ilmiahnya: membangun molekul, menghitung energi, mengoptimasi geometri, membaca output, menganalisis trajectory, membuat descriptor, atau menampilkan hasil.

Orientasi Bab

Subbab ini menguraikan Orientasi Bab. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

LIBRARY PYTHON UNTUK RISET KIMIA KOMPUTASI

Berikut daftar **pustaka Python untuk hitungan kimia komputasi** dan **konsep kimia/komputasi yang terkait**, disusun sebagai tabel rujukan praktis. Catatan penting: "semua" dalam ekosistem Python tidak mungkin benar-benar absolut karena pustaka baru terus muncul; daftar ini mencakup pustaka utama, matang, banyak dipakai, dan relevan untuk **kimia kuantum, dinamika molekul, cheminformatics, material, spektroskopi, machine learning, workflow, dan visualisasi**.

A. Tabel Library Python Kimia Komputasi

Bagian ini menyajikan A. Tabel Library Python Kimia Komputasi sebagai rujukan ringkas. Tabel perlu dibaca bersama penjelasan sebelum dan sesudahnya agar pembaca memahami fungsi setiap istilah, library, parameter, atau pilihan metode dalam alur riset.

No.	Library / Paket	Bidang Utama	Konsep Kimia yang Didukung	Kegunaan Rinci
1	**PySCF**	Kimia kuantum ab initio	HF, DFT, MP2, CC, CASSCF, TDDFT, basis set Gaussian, sistem molekul dan periodik	Menghitung energi elektronik, optimasi geometri, orbital molekul, muatan, momen dipol, eksitasi elektronik, sistem padatan/periodik. Cocok untuk riset dan pengembangan metode karena berbasis Python dan modular. (PySCF)
2	**Psi4**	Kimia kuantum ab initio	HF, DFT, MP2, CCSD, properti molekul, basis set	Paket kimia kuantum open-source dengan inti C++/Python; dapat dipakai sebagai modul Python untuk workflow otomatis, praktikum, benchmarking, dan komputasi energi/properti molekul. (PsiCode)
3	**GPAW**	DFT material dan molekul	DFT, PAW, plane wave, real-space grid, LCAO, TDDFT, material periodik	Menghitung struktur elektronik material, band structure, DOS, optimasi geometri, eksitasi optik, defect, permukaan, nanopartikel; terintegrasi dengan ASE. (GPAW)
4	**DFTB+ Python API**	Semiempiris kuantum	DFTB, energi, gaya atom, muatan Mulliken, dinamika atomistik	Alternatif cepat terhadap DFT untuk molekul besar/material; dapat diekstrak energinya, gaya atomnya, dan muatannya melalui Python API. (DFTB+ Recipes)
5	**xtb / tblite**	Semiempiris cepat	GFN-xTB, optimasi geometri, energi, muatan, frekuensi	Sangat berguna untuk optimasi awal struktur, screening molekul organik, konformer, energi relatif, dan geometri sebelum DFT. Dokumentasi xtb berfokus pada metode semiempirical tight-binding GFN-xTB. (XTB Docs)
6	**ASE – Atomic Simulation Environment**	Kerangka simulasi atomistik	Atom, molekul, kristal, kalkulator energi, optimasi, MD, surface, defect	"Lem" utama untuk membuat struktur, menjalankan kalkulasi melalui kalkulator eksternal, optimasi geometri, NEB, permukaan, kristal, dan workflow material. (ASE community)
7	**geomeTRIC**	Optimasi geometri	Energy minimization, constrained optimization, transition state, NEB, conical intersection	Optimasi struktur molekul berbasis koordinat internal TRIC; dapat memanggil Psi4, PySCF, Gaussian, Q-Chem, OpenMM, Gromacs, dan lainnya. (GitHub)
8	**Basis Set Exchange**	Basis set kimia kuantum	STO-3G, 3-21G, 6-31G, cc-pVXZ, def2, ECP	Mengambil, mengonversi, memanipulasi, dan menampilkan basis set untuk berbagai program kimia kuantum. Sangat penting

				untuk praktikum PySCF/Psi4. (MOLSSI BSE)
9	**QCEngine**	Eksekutor program kimia kuantum	QCSchema, standardisasi input-output, eksekusi program QC	Wrapper untuk menjalankan berbagai program kimia kuantum dengan format data standar; berguna untuk workflow otomatis dan high-throughput. (MolSSI)
10	**QCElemental**	Struktur data kimia kuantum	konstanta fisika, tabel periodik, molekul, satuan	Menyediakan struktur data inti untuk kimia kuantum, konstanta NIST, tabel periodik, dan handler molekul. (GitHub)
11	**cclib**	Parser output kimia kuantum	energi, koordinat, orbital, frekuensi, TDDFT, muatan	Membaca output Gaussian, ORCA, GAMESS, NWChem, Psi4, Q-Chem, dan lain-lain; memudahkan ekstraksi data hasil komputasi untuk analisis Python. (Cclib)
12	**IOData**	Input-output format kimia	format QC, MD, plane-wave DFT, konversi file	Membaca, menyimpan, mengonversi file kimia kuantum, dinamika molekul, dan DFT plane-wave; juga bisa membantu membuat input program. (IOData)
13	**RDKit**	Cheminformatics	SMILES, SMARTS, fingerprint, descriptor, substructure, conformer	Membuat, membaca, memanipulasi molekul; menghitung descriptor, fingerprint, similarity, pencarian substruktur, generasi konformer, QSAR, drug design. (RDKit)
14	**Open Babel / Pybel**	Konversi format kimia	MOL, SDF, PDB, XYZ, SMILES, InChI, aromatisitas, muatan parsial	Konversi antarformat kimia, manipulasi molekul, protonasi sederhana, persepsi ikatan, fingerprint, dan persiapan struktur. Python binding memberi akses ke hampir semua interface Open Babel. (Open Babel)
15	**Mordred**	Descriptor molekul	descriptor 2D/3D, QSAR, QSPR	Menghitung ribuan descriptor molekul untuk pemodelan sifat, toksisitas, aktivitas biologis, dan pembelajaran mesin.
16	**MolVS**	Standardisasi molekul	normalisasi, tautomer, fragment, muatan	Membersihkan struktur kimia sebelum QSAR/ML: standardisasi garam, fragment, tautomer, muatan, dan bentuk representasi.
17	**SELFIES**	Representasi molekul AI	string molekul valid, generative chemistry	Alternatif SMILES yang lebih robust untuk generasi molekul dengan machine learning karena string cenderung selalu valid secara kimia.
18	**OpenMM**	Dinamika molekul	molecular dynamics, force field, integrator, ensemble, GPU	Simulasi MD biomolekul/material lunak; mendukung custom force, integrator, GPU, dan dapat dipakai sebagai library Python. (OpenMM)
19	**MDAnalysis**	Analisis trajectory MD	RMSD, RMSF, RDF, kontak, seleksi atom, trajectory	Membaca dan menganalisis trajectory dari Gromacs, CHARMM, Amber, NAMD, LAMMPS, DL_POLY, PDB, XYZ, dan lain-lain. (MDAnalysis)
20	**MDTraj**	Analisis trajectory MD	RMSD, SASA, hydrogen bond, DSSP, order parameter	Pustaka ringan dan cepat untuk membaca, menulis, serta menganalisis trajectory MD dalam banyak format. (MDTraj)
21	**ParmEd**	Topologi molecular mechanics	Amber, CHARMM, OpenMM, AMOEBA, GROMOS	Mengedit, mengonversi, dan memeriksa topologi/parameter force field biomolekul. Berguna saat menyiapkan sistem MD. (ParMed)
22	**pytraj**	Analisis MD Amber/CPPTRAJ	RMSD, distance, angle, dihedral, clustering	Antarmuka Python untuk analisis trajectory bergaya Amber/CPPTRAJ. Berguna untuk pengguna AmberTools.
23	**ProDy**	Dinamika protein	normal mode analysis, elastic network, PCA, domain motion	Analisis struktur, dinamika, variasi konformasi, dan ko-evolusi protein. (GitHub)
24	**Biopython Bio.PDB**	Struktur biomolekul	PDB, mmCIF, struktur protein, atom/residu/rantai	Membaca dan memanipulasi struktur biomakromolekul; cocok untuk preprocessing protein sebelum docking/MD. (Biopython)

25	**PDBFixer**	Persiapan struktur biomolekul	missing atoms, missing residues, protonation, solvation	Memperbaiki file PDB sebelum simulasi OpenMM: atom hilang, residu hilang, hidrogen, pelarut, ion.
26	**mBuild**	Pembangun sistem molekuler	polimer, permukaan, sistem kompleks, coarse-grained	Menyusun sistem molekuler besar secara programatik: monomer, polimer, membran, nanopori, material lunak.
27	**Foyer**	Force field assignment	atom typing, XML force field, OpenMM/Gromacs	Menetapkan tipe atom dan parameter force field untuk sistem yang dibangun dengan mBuild/ParmEd/OpenMM.
28	**pymbar**	Termodinamika statistik	MBAR, free energy, reweighting	Menghitung beda energi bebas dari simulasi banyak keadaan, umbrella sampling, alchemical transformation.
29	**alchemlyb**	Analisis energi bebas	FEP, TI, BAR, MBAR	Analisis hasil simulasi alkimia untuk energi bebas ikatan, solvasi, mutasi, dan perubahan ligan.
30	**PLUMED Python tools / py-plumed**	Enhanced sampling	metadynamics, umbrella sampling, collective variables	Membantu membaca/analisis output PLUMED untuk simulasi MD dengan variabel kolektif.
31	**pymatgen**	Material science	struktur kristal, defect, phase diagram, electronic structure	Analisis struktur dan molekul, input-output program struktur elektronik, phase diagram, band structure, DOS, material data. (pymatgen)
32	**spglib**	Simetri kristal	space group, symmetry operation, primitive cell	Menentukan grup ruang, sel primitif, standar sel, dan simetri kristal untuk komputasi material.
33	**matminer**	Data mining material	featurization, dataset material, ML properties	Mengambil dataset material, membuat fitur/deskriptor material, dan membangun pipeline ML material. (Hacking Materials)
34	**phonopy**	Fonon dan vibrasi kristal	phonon, force constants, quasi-harmonic approximation	Menghitung fonon, dispersi fonon, DOS fonon, kapasitas panas, entropi vibrasi, dan stabilitas dinamik kristal. (Phonopy)
35	**phono3py**	Transport termal fonon	phonon-phonon interaction, thermal conductivity	Menghitung interaksi fonon-fonon dan konduktivitas termal kisi untuk material.
36	**sumo**	Plot hasil ab initio	band structure, DOS, optical absorption, effective mass	Membuat plot publikasi untuk band structure, DOS, phonon band, absorpsi optik, dan massa efektif. (SMTG Bham)
37	**atomate2**	Workflow material	DFT high-throughput, VASP, CP2K, force field, database	Workflow otomatis untuk komputasi material skala besar, screening, database properti, dan integrasi Materials Project. (GitHub)
38	**AiiDA**	Workflow dan provenance	otomasi kalkulasi, provenance, reproducibility, plugin	Mengelola, menjalankan, menyimpan, mereproduksi workflow komputasi kimia/material; kuat untuk HPC dan proyek riset besar. (AiiDA)
39	**FireWorks**	Workflow HPC	task, launchpad, queue, high-throughput	Menjalankan ribuan pekerjaan komputasi secara otomatis, umum dipakai bersama atomate/pymatgen.
40	**jobflow**	Workflow Python modern	DAG workflow, job, response, store	Membuat workflow komputasi modular; dipakai dalam atomate2.
41	**custodian**	Error handling workflow	deteksi error, koreksi otomatis, restart job	Memantau kalkulasi VASP/QChem/dll., mengenali error umum, dan mencoba perbaikan otomatis.
42	**DeepChem**	AI/ML kimia	QSAR, GNN, molecular property prediction, drug discovery, material	Toolkit deep learning untuk kimia, biologi, drug discovery, material science, dan quantum chemistry. (GitHub)
43	**DScribe**	Descriptor atomistik ML	Coulomb matrix, SOAP, ACSF, MBTR, Ewald matrix	Mengubah struktur atom menjadi fingerprint/deskriptor numerik untuk machine learning sifat molekul/material. (GitHub)

44	**SchNetPack**	Neural network atomistik	potential energy surface, atomistic neural network, quantum properties	Membangun dan melatih neural network untuk memprediksi energi, gaya, permukaan energi potensial, dan properti kuantum molekul/material. (GitHub)
45	**MACE**	Machine-learning interatomic potential	equivariant message passing, ML potential, energi/gaya	Membuat potential energi antaratom berbasis machine learning yang cepat dan akurat untuk simulasi atomistik. (GitHub)
46	**TorchANI**	Neural network potential	ANI potential, energi molekul organik, gaya atom	Prediksi energi dan gaya molekul organik secara cepat; cocok untuk screening dan optimasi awal.
47	**NequIP / Allegro**	ML interatomic potential	equivariant neural network, energi/gaya, MD	Potential ML berbasis simetri E(3), kuat untuk material dan sistem atomistik.
48	**PyTorch Geometric**	Graph neural network	molecular graph, message passing, QSAR	Membuat model GNN untuk molekul: prediksi aktivitas, toksisitas, sifat fisikokimia, dan material graph.
49	**scikit-learn**	Machine learning klasik	regression, classification, clustering, PCA	QSAR/QSPR, prediksi energi, clustering konformer, regresi sifat molekul/material, validasi model.
50	**XGBoost / LightGBM**	ML tabular	descriptor-based QSAR, property prediction	Model kuat untuk dataset descriptor molekul/material; sering lebih stabil untuk data kecil-menengah daripada deep learning.
51	**Qiskit Nature**	Quantum computing chemistry	ground state, excited state, dipole moment, electronic/vibrational structure	Menyusun problem kimia kuantum untuk algoritma komputasi kuantum seperti VQE dan estimasi energi keadaan dasar/tereksitasi. (GitHub)
52	**OpenFermion**	Quantum computing chemistry	fermionic Hamiltonian, qubit Hamiltonian, Jordan-Wigner, Bravyi-Kitaev	Mengubah masalah struktur elektronik menjadi Hamiltonian fermion/qubit untuk simulasi pada komputer kuantum. (GitHub)
53	**PennyLane qchem**	Differentiable quantum chemistry	VQE, differentiable HF, molecular Hamiltonian, gradients	Quantum machine learning dan quantum chemistry terdiferensiasi; berguna untuk VQE, optimasi geometri kuantum, dan eksperimen hybrid quantum-classical. (arXiv)
54	**Tangelo**	Quantum chemistry workflow	quantum chemistry on quantum computers, state preparation, error mitigation	Workflow end-to-end untuk simulasi kimia pada komputer kuantum dan berbagai backend seperti Qiskit, Cirq, Braket, Azure Quantum. (GitHub)
55	**py3Dmol**	Visualisasi molekul	struktur 3D, Jupyter, WebGL, PDB/SDF/XYZ	Menampilkan struktur molekul 3D interaktif di Jupyter Notebook; dapat dipakai bersama RDKit, MDAnalysis, MDTraj, OpenBabel, cclib. (GitHub)
56	**nglview**	Visualisasi biomolekul/trajectory	protein, ligand, trajectory MD, Jupyter widget	Visualisasi struktur molekul dan trajectory MD langsung di Jupyter Notebook. (GitHub)
57	**NetworkX**	Graf molekul	graph, adjacency, ring, path, topology	Analisis molekul sebagai graf: atom sebagai node, ikatan sebagai edge; berguna untuk cheminformatics, reaksi, dan GNN sederhana.
58	**NumPy**	Komputasi numerik dasar	vektor, matriks, koordinat, array atom	Fondasi perhitungan koordinat, matriks Hamiltonian, matriks jarak, gaya, energi, dan data numerik kimia.
59	**SciPy**	Komputasi ilmiah	optimasi, integral, eigenvalue, fitting, ODE	Optimasi geometri sederhana, fitting parameter force field, penyelesaian persamaan diferensial, diagonalization, integral numerik.
60	**SymPy**	Aljabar simbolik	persamaan, turunan, integral, operator	Menurunkan persamaan kimia kuantum sederhana, mekanika kuantum dasar, termodinamika, dan model matematis.
61	**Pandas**	Data kimia	tabel descriptor, hasil energi, dataset molekul	Mengelola dataset hasil perhitungan: energi, HOMO-LUMO, descriptor, frekuensi, tabel QSAR, hasil screening.

62	**Matplotlib / Plotly**	Visualisasi data	spektrum, grafik energi, PES, band, DOS	Membuat grafik energi, kurva PES, spektrum IR/UV-Vis, RMSD, RDF, band structure, dan grafik pembelajaran mesin.
63	**Pint**	Satuan fisika-kimia	Hartree, eV, kcal/mol, Å, bohr, Debye	Konversi satuan kimia komputasi agar tidak salah antara Hartree, eV, kcal/mol, kJ/mol, bohr, Å, nm.
64	**periodictable / mendeleev**	Data unsur	massa atom, nomor atom, isotop, konfigurasi elektron	Mengambil data unsur untuk input molekul, massa molar, isotope labeling, dan pembelajaran kimia dasar.
65	**JupyterLab / ipywidgets**	Lingkungan interaktif	notebook, widget, praktikum digital	Membuat praktikum kimia komputasi interaktif, visualisasi struktur, input molekul, kontrol parameter metode/basis set.

B. Tabel Konsep Kimia Komputasi dan Library yang Sesuai

Bagian ini menyajikan B. Tabel Konsep Kimia Komputasi dan Library yang Sesuai sebagai rujukan ringkas. Tabel perlu dibaca bersama penjelasan sebelum dan sesudahnya agar pembaca memahami fungsi setiap istilah, library, parameter, atau pilihan metode dalam alur riset.

No.	Konsep Kimia / Komputasi	Makna Singkat	Kegunaan dalam Hitungan Kimia	Library yang Relevan
1	**Struktur molekul**	Susunan atom dan koordinat 3D	Input awal semua komputasi: energi, optimasi, MD, docking	RDKit, Open Babel, ASE, PySCF, Psi4, pymatgen
2	**Format file kimia**	XYZ, MOL, SDF, PDB, CIF, CML, Gaussian log	Menukar data antarprogram kimia	Open Babel, RDKit, cclib, IOData, ASE, pymatgen
3	**SMILES**	Notasi teks molekul	Input ringkas untuk molekul organik, QSAR, database	RDKit, Open Babel, DeepChem
4	**SMARTS**	Pola substruktur kimia	Mencari gugus fungsi, motif farmakofor, reaksi	RDKit, Open Babel
5	**InChI / InChIKey**	Identitas standar molekul	Deduplicasi database, pencarian senyawa	RDKit, Open Babel
6	**Konformer**	Bentuk 3D berbeda dari molekul sama	Mencari geometri awal terbaik, docking, energi relatif	RDKit, xtb, PySCF, Psi4, OpenMM
7	**Optimasi geometri**	Mencari struktur energi minimum	Menentukan panjang ikatan, sudut, energi minimum	PySCF, Psi4, ASE, geomeTRIC, xtb, DFTB+, GPAW
8	**Potential Energy Surface – PES**	Permukaan energi terhadap koordinat inti	Studi reaksi, transisi, konformer, vibrasi	PySCF, Psi4, ASE, geomeTRIC, SchNetPack, MACE
9	**Single-point energy**	Energi pada geometri tetap	Membandingkan stabilitas struktur tanpa optimasi ulang	PySCF, Psi4, GPAW, DFTB+, xtb
10	**Hartree-Fock – HF**	Metode medan rata-rata elektron	Dasar kimia kuantum, referensi untuk post-HF	PySCF, Psi4
11	**DFT**	Teori fungsional kerapatan	Energi dan struktur molekul/material dengan biaya sedang	PySCF, Psi4, GPAW, DFTB+
12	**Functional DFT**	B3LYP, PBE, PBE0, M06, ωB97X	Menentukan akurasi energi, struktur, spektrum	PySCF, Psi4, GPAW
13	**Basis set**	Fungsi matematika untuk orbital atom	Menentukan kualitas hasil HF/DFT/post-HF	Basis Set Exchange, PySCF, Psi4
14	**Minimal basis**	STO-3G, basis kecil	Praktikum dasar, hitungan cepat	PySCF, Psi4
15	**Split-valence basis**	3-21G, 6-31G	Kompromi cepat-akurat untuk molekul organik kecil	PySCF, Psi4
16	**Polarization function**	Fungsi d, f, p tambahan	Memperbaiki bentuk orbital saat ikatan/polarisasi	PySCF, Psi4
17	**Diffuse function**	Fungsi orbital menyebar	Anion, ikatan hidrogen, eksitasi, sistem lemah	PySCF, Psi4
18	**ECP / pseudopotential**	Mengganti elektron inti berat	Perhitungan unsur berat lebih ringan	PySCF, Psi4, GPAW, pymatgen

19	**SCF convergence**	Konvergensi siklus medan sendiri	Menentukan berhasil/tidaknya HF/DFT	PySCF, Psi4, GPAW
20	**Orbital molekul**	HOMO, LUMO, orbital terisi/kosong	Menafsirkan reaktivitas, warna, transfer elektron	PySCF, Psi4, cclib, py3Dmol
21	**HOMO-LUMO gap**	Selisih energi orbital	Indikator reaktivitas, kestabilan, sifat optik	PySCF, Psi4, RDKit, cclib
22	**Muatan atom parsial**	Mulliken, Löwdin, RESP, ESP	Analisis polaritas, reaktivitas, force field	PySCF, Psi4, DFTB+, RDKit, Open Babel
23	**Momen dipol**	Ukuran polaritas molekul	Menilai polaritas, interaksi antarmolekul, IR	PySCF, Psi4
24	**Frekuensi vibrasi**	Getaran normal molekul	Spektrum IR/Raman, verifikasi minimum, ZPE	PySCF, Psi4, xtb, DFTB+, phonopy
25	**IR spectrum**	Absorpsi vibrasi inframerah	Identifikasi gugus fungsi dan mode vibrasi	PySCF, Psi4, cclib, Matplotlib
26	**Raman spectrum**	Hamburan Raman	Analisis mode vibrasi aktif Raman	Psi4, PySCF, cclib
27	**UV-Vis / TDDFT**	Transisi elektronik	Prediksi panjang gelombang serapan, warna, eksitasi	PySCF, Psi4, GPAW, cclib
28	**NMR shielding**	Perisai magnetik inti	Prediksi chemical shift NMR	PySCF, Psi4
29	**Thermochemistry**	ZPE, entalpi, entropi, Gibbs	Menilai spontanitas dan kestabilan termal	PySCF, Psi4, xtb, cclib
30	**Transition state**	Titik pelana reaksi	Menentukan keadaan transisi dan energi aktivasi	geomeTRIC, ASE, PySCF, Psi4
31	**IRC**	Jalur reaksi dari TS	Memastikan TS menghubungkan reaktan-produk	Psi4, PySCF, geomeTRIC
32	**NEB**	Nudged elastic band	Mencari jalur reaksi pada permukaan/material	ASE, geomeTRIC, GPAW, MACE
33	**Solvation model**	PCM, implicit solvent, explicit solvent	Memodelkan pengaruh pelarut pada energi/properti	Psi4, PySCF, OpenMM
34	**Ikatan hidrogen**	Interaksi donor-akseptor H	Analisis struktur biomolekul, pelarut, kristal	MDAnalysis, MDTraj, RDKit, OpenMM
35	**Interaksi nonkovalen**	van der Waals, π-π, ionik, hidrofobik	Protein-ligan, kristal, supramolekul	RDKit, MDAnalysis, OpenMM, ProLIF
36	**Molecular mechanics – MM**	Energi dari force field klasik	Simulasi cepat molekul besar	OpenMM, ParmEd, RDKit, ASE
37	**Force field**	AMBER, CHARMM, OPLS, GAFF, AMOEBA	Parameter ikatan, sudut, torsi, nonbonded	OpenMM, ParmEd, Foyer, mBuild
38	**Molecular dynamics – MD**	Evolusi atom terhadap waktu	Struktur dinamis protein, pelarut, material lunak	OpenMM, MDAnalysis, MDTraj, pytraj
39	**Integrator MD**	Verlet, Langevin, Brownian	Mengontrol evolusi waktu dalam simulasi	OpenMM
40	**Ensemble MD**	NVE, NVT, NPT	Mengontrol energi, suhu, tekanan	OpenMM, MDAnalysis
41	**Thermostat/barostat**	Kontrol suhu/tekanan	Simulasi kondisi realistis	OpenMM
42	**RMSD**	Deviasi struktur terhadap referensi	Stabilitas protein/ligan selama MD	MDAnalysis, MDTraj, pytraj
43	**RMSF**	Fluktuasi residu/atom	Menilai fleksibilitas bagian molekul	MDAnalysis, MDTraj, pytraj
44	**RDF**	Radial distribution function	Struktur cairan, solvasi, distribusi ion	MDAnalysis, MDTraj
45	**SASA**	Luas permukaan terakses pelarut	Lipatan protein, binding, hidrofobisitas	MDTraj, MDAnalysis, FreeSASA
46	**Hydrogen bond analysis**	Deteksi ikatan hidrogen trajectory	Analisis stabilitas protein-ligan/air	MDAnalysis, MDTraj
47	**Docking molekul**	Prediksi pose ligan pada reseptor	Drug design, screening senyawa aktif	RDKit, Meeko, AutoDock Vina Python, Open Babel
48	**Pharmacophore**	Pola fitur interaksi ligan	Virtual screening berbasis fitur	RDKit, ODDT, ProLIF

49	**QSAR/QSPR**	Hubungan struktur-aktivitas/properti	Prediksi aktivitas biologis, toksisitas, logP, pKa	RDKit, Mordred, DeepChem, scikit-learn, XGBoost
50	**Molecular fingerprint**	Representasi bit/fingerprint molekul	Similarity search, clustering, QSAR	RDKit, Open Babel
51	**Descriptor molekul**	Angka yang menggambarkan sifat molekul	Input ML untuk prediksi sifat	RDKit, Mordred, DeepChem
52	**Graph molekul**	Atom-node, ikatan-edge	GNN, pencarian subgraf, reaksi	RDKit, NetworkX, PyTorch Geometric
53	**Machine learning potential**	Energi/gaya dari model ML	MD cepat mendekati akurasi DFT	SchNetPack, MACE, NequIP, TorchANI
54	**Atomistic descriptor**	SOAP, ACSF, MBTR, Coulomb matrix	Fitur struktur atom untuk ML	DScribe, matminer
55	**High-throughput screening**	Banyak kalkulasi otomatis	Screening material, katalis, obat, konformer	AiiDA, atomate2, FireWorks, jobflow, ASE
56	**Provenance**	Riwayat lengkap workflow/data	Reproduksibilitas riset komputasi	AiiDA, QCEngine
57	**Struktur kristal**	Sel satuan dan posisi atom periodik	Material, mineral, katalis, padatan	pymatgen, ASE, spglib, GPAW
58	**Space group**	Simetri kristal	Standarisasi struktur, analisis material	spglib, pymatgen
59	**Band structure**	Energi elektron vs k-point	Sifat semikonduktor/logam/isolator	GPAW, pymatgen, sumo
60	**Density of states – DOS**	Distribusi tingkat energi	Analisis sifat elektronik material	GPAW, pymatgen, sumo
61	**Phonon**	Kuanta getaran kisi	Stabilitas dinamik, kapasitas panas, konduktivitas termal	phonopy, phono3py, GPAW, ASE
62	**Defect material**	Vacancy, interstitial, substitution	Sifat semikonduktor, katalis, baterai	pymatgen, GPAW, atomate2
63	**Surface/slab model**	Model permukaan material	Adsorpsi, katalisis heterogen	ASE, pymatgen, GPAW
64	**Adsorpsi**	Molekul menempel di permukaan	Katalis, sensor, material energi	ASE, GPAW, pymatgen
65	**Bader/charge analysis**	Pembagian kerapatan elektron	Muatan atom dalam material	GPAW, pymatgen, tools eksternal
66	**Quantum computing chemistry**	Simulasi kimia dengan qubit	VQE, Hamiltonian molekul, simulasi fermion	Qiskit Nature, OpenFermion, PennyLane, Tangelo
67	**VQE**	Variational quantum eigensolver	Estimasi energi keadaan dasar molekul di komputer kuantum	Qiskit Nature, PennyLane, Tangelo, OpenFermion
68	**Fermion-to-qubit mapping**	Jordan-Wigner, Bravyi-Kitaev	Mengubah Hamiltonian elektron ke operator qubit	OpenFermion, Qiskit Nature
69	**Visualisasi molekul 3D**	Tampilan atom-ikatan/permukaan	Presentasi, praktikum, validasi struktur	py3Dmol, nglview, RDKit, MDAnalysis
70	**Konversi satuan kimia**	Hartree, eV, kcal/mol, Å, bohr	Menghindari salah interpretasi energi/jarak	Pint, QCElemental, NumPy

C. Paket yang Paling Direkomendasikan Berdasarkan Tujuan

Bagian ini menjelaskan C. Paket yang Paling Direkomendasikan Berdasarkan Tujuan sebagai panduan teknis. Setiap perintah perlu dijalankan secara bertahap, diperiksa pesan errornya, dan dicatat versinya agar lingkungan komputasi dapat direproduksi.

Tujuan Praktikum / Riset	Paket Minimal	Paket Tambahan
Kimia kuantum molekul kecil	PySCF, Psi4, Basis Set Exchange	geomeTRIC, cclib, py3Dmol
Optimasi struktur cepat sebelum DFT	RDKit, xtb/tblite, Open Babel	ASE, geomeTRIC
DFT material/kristal	ASE, GPAW, pymatgen, spglib	sumo, phonopy, atomate2
Dinamika molekul biomolekul	OpenMM, MDAnalysis, MDTraj, ParmEd	PDBFixer, pytraj, ProDy, nglview
QSAR/QSPR dan cheminformatics	RDKit, Mordred, pandas, scikit-learn	DeepChem, XGBoost, SELFIES
Machine learning atomistik	DScribe, SchNetPack, MACE	PyTorch Geometric, NequIP, TorchANI
Workflow high-throughput	ASE, AiiDA, atomate2, jobflow	custodian, FireWorks, QCEngine
Visualisasi untuk pembelajaran	py3Dmol, nglview, matplotlib	plotly, ipywidgets
Quantum computing chemistry	Qiskit Nature, OpenFermion	PennyLane, Tangelo

---

D. Contoh Paket Instalasi Conda untuk Lingkungan "Kimia Komputasi"

Bagian ini menjelaskan D. Contoh Paket Instalasi Conda untuk Lingkungan "Kimia Komputasi" sebagai panduan teknis. Setiap perintah perlu dijalankan secara bertahap, diperiksa pesan errornya, dan dicatat versinya agar lingkungan komputasi dapat direproduksi.

```
conda create -n kimia-komputasi python=3.12 -y
conda activate kimia-komputasi
```

```
conda install -c conda-forge numpy scipy pandas matplotlib jupyterlab ipywidgets -y
conda install -c conda-forge rdkit openbabel py3dmol nglview -y
conda install -c conda-forge pyscf psi4 geometric basis_set_exchange cclib -y
conda install -c conda-forge ase pymatgen spglib phonopy sumo -y
conda install -c conda-forge openmm mdanalysis mdtraj parmed pdbfixer -y
conda install -c conda-forge scikit-learn xgboost deepchem dscribe -y
```

Untuk paket tertentu seperti **GPAW, MACE, SchNetPack, NequIP, Qiskit Nature, PennyLane, Tangelo, AiiDA, atomate2**, instalasinya sebaiknya dibuat lingkungan terpisah karena dependensinya cukup besar dan kadang konflik dengan paket lain.

E. Ringkasan Paling Penting

Level Kebutuhan	Library Wajib
Pemula/praktikum kimia komputasi	RDKit, PySCF, Psi4, ASE, py3Dmol, pandas, matplotlib
Komputasi molekul organik	RDKit, Open Babel, xtb, PySCF, Psi4, cclib
Spektroskopi komputasi	PySCF, Psi4, cclib, matplotlib
Dinamika molekul	OpenMM, MDAnalysis, MDTraj, ParmEd, PDBFixer
Material dan kristal	ASE, pymatgen, GPAW, spglib, phonopy, sumo
AI kimia	RDKit, Mordred, DeepChem, DScribe, SchNetPack, MACE
Workflow riset besar	AiiDA, atomate2, jobflow, custodian, QCEngine
Quantum computing chemistry	Qiskit Nature, OpenFermion, PennyLane, Tangelo

Kesimpulan praktis:

Untuk membangun ekosistem Python kimia komputasi yang lengkap, kombinasi paling seimbang adalah:

RDKit + Open Babel + PySCF + Psi4 + ASE + OpenMM + MDAnalysis + pymatgen + cclib + py3Dmol + pandas + matplotlib

Lalu ditambah sesuai bidang khusus: **GPAW/phonopy/sumo** untuk material, **DeepChem/DScribe/MACE** untuk AI kimia, dan **Qiskit Nature/OpenFermion/PennyLane** untuk quantum computing chemistry.

F. Skrip Instalasi Library

Berikut skrip instalasi **panjang dan lengkap** untuk ekosistem Python kimia komputasi. Dibuat untuk **Ubuntu/WSL Linux**, karena library seperti **Psi4, PySCF, OpenMM, GPAW, AiiDA, atomate2, MACE, NequIP, RDKit, Open Babel** paling stabil di Linux/WSL.

Dipecah ke beberapa environment Conda agar **lebih benar dan tidak mudah konflik dependency**. RDKit, OpenMM, MDAnalysis, AiiDA, dan beberapa paket besar memang lazim dipasang lewat Conda/conda-forge; MACE memakai paket PyPI bernama **mace-torch**, bukan mace; OpenFermion juga secara resmi memakai pip. (RDKit)

Simpan sebagai:

install-kimia-komputasi-full.sh

Lalu jalankan:

```
chmod +x install-kimia-komputasi-full.sh
./install-kimia-komputasi-full.sh
```

Skrip instalasi legkap:

```
#!/usr/bin/env bash
# ============================================================
```

 # INSTALLER LENGKAP PYTHON KIMIA KOMPUTASI

Subbab ini menguraikan # INSTALLER LENGKAP PYTHON KIMIA KOMPUTASI. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# Untuk Ubuntu / WSL Linux
#
# author: Kasmui
# Telp.: 0818294312
#
# Environment yang dibuat:
# 1. kimia-qc          : kimia kuantum, cheminformatics, struktur molekul
# 2. kimia-md          : molecular dynamics dan biomolekul
# 3. kimia-material    : material science, kristal, fonon, DFT material
# 4. kimia-ml          : AI/ML kimia, descriptor, potential ML
# 5. kimia-workflow    : workflow high-throughput, AiiDA, atomate2
# 6. kimia-qcomp       : quantum computing chemistry
```

```
# 7. kimia-all-lite    : lingkungan ringkas gabungan untuk praktikum umum
#
# Catatan:
# - Jangan memaksa semua paket berat masuk satu environment jika ingin stabil.
# - Beberapa paket AI atomistik dapat berubah cepat; script ini memasang versi terbaru
yang kompatibel saat dijalankan.
# - Untuk GPU/CUDA, sesuaikan bagian PyTorch pada environment kimia-ml.
# ============================================================
```

set -euo pipefail

```
# ------------------------------------------------------------
```

PENGATURAN DASAR

Subbab ini menguraikan # PENGATURAN DASAR. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ------------------------------------------------------------
```

PYTHON_VERSION="3.12"

Subbab ini menguraikan PYTHON_VERSION="3.12". Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
CONDA_CHANNELS="-c conda-forge"
LOG_DIR="$HOME/kimia-komputasi-install-logs"
```

mkdir -p "$LOG_DIR"

```
echo "============================================================"
echo "INSTALLER PYTHON KIMIA KOMPUTASI LENGKAP"
echo "author: Kasmui"
echo "Telp.: 0818294312"
echo "============================================================"
```

echo

```
echo "Target OS       : Ubuntu / WSL Linux"
echo "Python          : $PYTHON_VERSION"
echo "Log instalasi   : $LOG_DIR"
```

echo

```
# ------------------------------------------------------------
```

FUNGSI UTILITAS

Subbab ini menguraikan # FUNGSI UTILITAS. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ------------------------------------------------------------
```

run_step() {

local step_name="$1"

shift

echo

```
  echo "============================================================"
  echo "MENJALANKAN: $step_name"
  echo "============================================================"
```

"$@" 2>&1 | tee "$LOG_DIR/${step_name// /_}.log"

}

have_cmd() {

```
    command -v "$1" >/dev/null 2>&1
}

activate_conda_base() {
    if have_cmd conda; then
        local base_path
        base_path="$(conda info --base)"
        # shellcheck disable=SC1090
        source "$base_path/etc/profile.d/conda.sh"
    elif [ -f "$HOME/miniforge3/etc/profile.d/conda.sh" ]; then
        # shellcheck disable=SC1090
        source "$HOME/miniforge3/etc/profile.d/conda.sh"
    else
        echo "Conda belum tersedia."
        exit 1
    fi
}

create_env_if_missing() {
    local env_name="$1"
    if conda env list | awk '{print $1}' | grep -qx "$env_name"; then
        echo "Environment '$env_name' sudah ada, dilewati pembuatan."
    else
        echo "Membuat environment '$env_name'..."
        mamba create -y -n "$env_name" python="$PYTHON_VERSION"
    fi
}

install_conda_packages() {
    local env_name="$1"
    shift
    echo
    echo "Memasang paket Conda pada environment: $env_name"
    echo "Paket:"
    printf ' - %s\n' "$@"
    mamba install -y -n "$env_name" $CONDA_CHANNELS "$@"
}

install_pip_packages() {
    local env_name="$1"
    shift
    echo
    echo "Memasang paket Pip pada environment: $env_name"
    echo "Paket:"
    printf ' - %s\n' "$@"
```

```
    conda run -n "$env_name" python -m pip install --upgrade pip setuptools wheel
    conda run -n "$env_name" python -m pip install "$@"
}

install_pip_packages_no_fail() {
    local env_name="$1"
    shift
    echo
    echo "Memasang paket Pip opsional pada environment: $env_name"
    echo "Jika gagal, instalasi paket lain tetap dilanjutkan."
    echo "Paket:"
    printf '  - %s\n' "$@"
    conda run -n "$env_name" python -m pip install --upgrade pip setuptools wheel || true
    for pkg in "$@"; do
        echo
        echo "Mencoba memasang paket opsional: $pkg"
        conda run -n "$env_name" python -m pip install "$pkg" || {
            echo "PERINGATAN: Paket opsional gagal dipasang: $pkg"
        }
    done
}

register_kernel() {
    local env_name="$1"
    local display_name="$2"
    echo
    echo "Mendaftarkan kernel Jupyter: $display_name"
    conda run -n "$env_name" python -m ipykernel install --user --name "$env_name" --
display-name "$display_name" || true
}

quick_import_test() {
    local env_name="$1"
    shift
    echo
    echo "Tes import cepat untuk environment: $env_name"
    conda run -n "$env_name" python - <<PY
mods = """$*""".split()
for m in mods:
    try:
        __import__(m)
        print(f"[OK] {m}")
    except Exception as e:
        print(f"[GAGAL/OPSIONAL] {m}: {type(e).__name__}: {e}")
PY
}

# ------------------------------------------------------------
# 0. DEPENDENSI SISTEM UBUNTU / WSL
```

Subbab ini menguraikan # 0. DEPENDENSI SISTEM UBUNTU / WSL. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ------------------------------------------------------------

echo "Memasang dependensi sistem Ubuntu/WSL..."
if have_cmd apt-get; then
    sudo apt-get update
    sudo apt-get install -y \
      build-essential \
      gfortran \
      cmake \
      make \
      git \
      wget \
      curl \
      unzip \
      bzip2 \
      pkg-config \
      ca-certificates \
      libgl1 \
      libglib2.0-0 \
      libxrender1 \
      libxext6 \
      libsm6 \
      libfontconfig1 \
      libfreetype6 \
      libxml2-dev \
      libxslt1-dev \
      zlib1g-dev \
      libbz2-dev \
      liblzma-dev \
      libffi-dev \
      libssl-dev \
      graphviz
else
    echo "apt-get tidak ditemukan. Lewati instalasi dependensi sistem."
fi

# ------------------------------------------------------------
# 1. INSTAL MINIFORGE / CONDA JIKA BELUM ADA
```

Bagian ini menjelaskan # 1. INSTAL MINIFORGE / CONDA JIKA BELUM ADA sebagai panduan teknis. Setiap perintah perlu dijalankan secara bertahap, diperiksa pesan errornya, dan dicatat versinya agar lingkungan komputasi dapat direproduksi.

```
# ------------------------------------------------------------
```

```
if have_cmd conda; then
    echo "Conda sudah tersedia."
```

else

```
    echo "Conda belum ditemukan. Menginstal Miniforge3 ke $HOME/miniforge3 ..."
```

cd "$HOME"

wget -O Miniforge3-Linux-x86_64.sh \

"https://github.com/conda-forge/miniforge/releases/latest/download/Miniforge3-Linux-x86_64.sh"

bash Miniforge3-Linux-x86_64.sh -b -p "$HOME/miniforge3"

rm -f Miniforge3-Linux-x86_64.sh

fi

activate_conda_base

```
# ------------------------------------------------------------
```

2. KONFIGURASI CONDA-FORGE DAN MAMBA

Bagian ini menjelaskan # 2. KONFIGURASI CONDA-FORGE DAN MAMBA sebagai panduan teknis. Setiap perintah perlu dijalankan secara bertahap, diperiksa pesan errornya, dan dicatat versinya agar lingkungan komputasi dapat direproduksi.

```
# ------------------------------------------------------------
```

```
echo "Mengatur channel conda-forge dengan strict priority..."
conda config --add channels conda-forge || true
conda config --set channel_priority strict
```

```
echo "Memasang / memperbarui mamba..."
conda install -y -n base -c conda-forge mamba
```

```
# ------------------------------------------------------------
# 3. ENVIRONMENT kimia-qc
# Kimia kuantum + cheminformatics + visualisasi molekul
# ------------------------------------------------------------
```

create_env_if_missing "kimia-qc"

install_conda_packages "kimia-qc" \

numpy \

scipy \

pandas \

matplotlib \

plotly \

sympy \

pint \

periodictable \

```
    mendeleev \
    networkx \
    tqdm \
    pyyaml \
    h5py \
    jupyterlab \
    notebook \
    ipywidgets \
    ipykernel \
    rdkit \
    openbabel \
    py3dmol \
    nglview \
    selfies \
    mordred \
    molvs \
    pyscf \
    psi4 \
    geometric \
    basis_set_exchange \
    qcelemental \
    qcengine \
    cclib \
    iodata \
    ase \
    xtb \
    tblite \
    dftbplus

install_pip_packages_no_fail "kimia-qc" \
    pybel \
    xyz2mol

register_kernel "kimia-qc" "Python (kimia-qc)"

quick_import_test "kimia-qc" \
    numpy scipy pandas matplotlib sympy pint periodictable mendeleev networkx \
    rdkit openbabel py3Dmol nglview selfies mordred \
```

pyscf psi4 geometric basis_set_exchange qcelemental qcengine cclib iodata ase

```
# ------------------------------------------------------------
# 4. ENVIRONMENT kimia-md
# Molecular dynamics, biomolekul, protein, trajectory
# ------------------------------------------------------------
```

create_env_if_missing "kimia-md"

install_conda_packages "kimia-md" \
numpy \
scipy \
pandas \
matplotlib \
plotly \
networkx \
tqdm \
jupyterlab \
notebook \
ipywidgets \
ipykernel \
rdkit \
openbabel \
py3dmol \
nglview \
openmm \
openmmtools \
mdanalysis \
mdtraj \
parmed \
pdbfixer \
prody \
biopython \
pymbar \
alchemlyb \
mbuild \
foyer \
freud \
plumed

```
install_pip_packages_no_fail "kimia-md" \
    prolif \
    freesasa \
    pytraj

register_kernel "kimia-md" "Python (kimia-md)"

quick_import_test "kimia-md" \
    numpy scipy pandas matplotlib rdkit openbabel py3Dmol nglview \
    openmm MDAnalysis mdtraj parmed pdbfixer prody Bio pymbar alchemlyb mbuild foyer

# ------------------------------------------------------------
# 5. ENVIRONMENT kimia-material
# Material science, kristal, fonon, DFT material
# ------------------------------------------------------------

create_env_if_missing "kimia-material"

install_conda_packages "kimia-material" \
    numpy \
    scipy \
    pandas \
    matplotlib \
    plotly \
    sympy \
    tqdm \
    h5py \
    jupyterlab \
    notebook \
    ipywidgets \
    ipykernel \
    ase \
    pymatgen \
    spglib \
    matminer \
    phonopy \
    phono3py \
    sumo \
    seekpath \
    pycifrw \
```

netcdf4 \
h5py \
dscribe \
custodian \
jobflow \
fireworks \
atomate2

```
# GPAW cukup berat; pada beberapa mesin WSL bisa gagal jika MPI/BLAS bermasalah.
# Dimasukkan sebagai paket Conda utama, tetapi tidak menghentikan semua instalasi jika
gagal.
```

echo

```
echo "Mencoba memasang GPAW pada kimia-material..."
mamba install -y -n kimia-material -c conda-forge gpaw || {
    echo "PERINGATAN: GPAW gagal dipasang. Lingkungan material tetap dapat digunakan tanpa
GPAW."
    echo "Coba manual nanti: conda activate kimia-material && mamba install -c conda-forge
gpaw"
}
```

register_kernel "kimia-material" "Python (kimia-material)"

quick_import_test "kimia-material" \
numpy scipy pandas matplotlib ase pymatgen spglib matminer phonopy phono3py sumo dscribe custodian jobflow fireworks

```
# ------------------------------------------------------------
# 6. ENVIRONMENT kimia-ml
# AI/ML kimia, QSAR/QSPR, descriptor, GNN, ML potential
# ------------------------------------------------------------
```

create_env_if_missing "kimia-ml"

install_conda_packages "kimia-ml" \
numpy \
scipy \
pandas \
matplotlib \
plotly \
sympy \
networkx \
tqdm \
h5py \
jupyterlab \
notebook \

```
    ipywidgets \
    ipykernel \
    rdkit \
    openbabel \
    py3dmol \
    selfies \
    mordred \
    scikit-learn \
    xgboost \
    lightgbm \
    dscribe \
    deepchem \
    ase \
    pymatgen

# PyTorch CPU default. Untuk CUDA, lihat catatan di bawah script.
echo
echo "Memasang PyTorch CPU untuk environment kimia-ml..."
conda run -n kimia-ml python -m pip install --upgrade pip setuptools wheel
conda run -n kimia-ml python -m pip install \
    torch \
    torchvision \
    torchaudio \
    --index-url https://download.pytorch.org/whl/cpu

# Paket AI atomistik
install_pip_packages_no_fail "kimia-ml" \
    torchani \
    schnetpack \
    mace-torch \
    nequip \
    e3nn \
    torch-geometric

# Allegro dari MIR group biasanya lebih aman dipasang dari GitHub.
# Jika gagal, paket lain tetap aman.
echo
echo "Mencoba memasang Allegro dari GitHub..."
conda run -n kimia-ml python -m pip install \
    "git+https://github.com/mir-group/allegro.git" || {
    echo "PERINGATAN: Allegro gagal dipasang. Paket ini opsional dan sering bergantung
pada versi PyTorch/e3nn."
}
```

```
register_kernel "kimia-ml" "Python (kimia-ml)"

quick_import_test "kimia-ml" \
    numpy scipy pandas matplotlib sklearn xgboost lightgbm rdkit selfies mordred deepchem dscribe torch torchani e3nn ase pymatgen

# ------------------------------------------------------------
# 7. ENVIRONMENT kimia-workflow
# Workflow high-throughput, provenance, otomasi HPC
# ------------------------------------------------------------

create_env_if_missing "kimia-workflow"

install_conda_packages "kimia-workflow" \
    numpy \
    scipy \
    pandas \
    matplotlib \
    pyyaml \
    tqdm \
    h5py \
    jupyterlab \
    notebook \
    ipywidgets \
    ipykernel \
    ase \
    pymatgen \
    custodian \
    jobflow \
    fireworks \
    atomate2 \
    aiida-core \
    aiida-core.services \
    qcelemental \
    qcengine \
    cclib

install_pip_packages_no_fail "kimia-workflow" \
    aiida-quantumespresso \
    aiida-vasp \
```

```
    aiida-cp2k \
    aiida-pseudo

register_kernel "kimia-workflow" "Python (kimia-workflow)"

quick_import_test "kimia-workflow" \
    numpy scipy pandas ase pymatgen custodian jobflow fireworks aiida qcelemental qcengine cclib

# ------------------------------------------------------------
# 8. ENVIRONMENT kimia-qcomp
# Quantum computing chemistry
# ------------------------------------------------------------

create_env_if_missing "kimia-qcomp"

install_conda_packages "kimia-qcomp" \
    numpy \
    scipy \
    pandas \
    matplotlib \
    sympy \
    networkx \
    jupyterlab \
    notebook \
    ipywidgets \
    ipykernel \
    pyscf \
    qiskit \
    qiskit-nature \
    pennylane

install_pip_packages "kimia-qcomp" \
    openfermion \
    openfermionpyscf

install_pip_packages_no_fail "kimia-qcomp" \
    tangelo-gc \
    qulacs \
    cirq

register_kernel "kimia-qcomp" "Python (kimia-qcomp)"
```

```
quick_import_test "kimia-qcomp" \
    numpy scipy pandas matplotlib sympy qiskit qiskit_nature pennylane openfermion pyscf

# ------------------------------------------------------------
# 9. ENVIRONMENT kimia-all-lite
# Lingkungan gabungan ringan untuk praktikum harian
# ------------------------------------------------------------

create_env_if_missing "kimia-all-lite"

install_conda_packages "kimia-all-lite" \
    numpy \
    scipy \
    pandas \
    matplotlib \
    plotly \
    sympy \
    pint \
    periodictable \
    mendeleev \
    networkx \
    tqdm \
    jupyterlab \
    notebook \
    ipywidgets \
    ipykernel \
    rdkit \
    openbabel \
    py3dmol \
    nglview \
    selfies \
    mordred \
    pyscf \
    psi4 \
    geometric \
    basis_set_exchange \
    cclib \
    iodata \
    ase \
```

xtb \

openmm \

mdanalysis \

mdtraj \

parmed \

pymatgen \

spglib \

phonopy \

scikit-learn \

xgboost \

dscribe

register_kernel "kimia-all-lite" "Python (kimia-all-lite)"

quick_import_test "kimia-all-lite" \

numpy scipy pandas matplotlib rdkit openbabel py3Dmol nglview \

pyscf psi4 geometric basis_set_exchange cclib iodata ase openmm MDAnalysis mdtraj parmed pymatgen spglib phonopy sklearn xgboost dscribe

```
# ------------------------------------------------------------
```

 # 10. MEMBUAT FILE AKTIVASI CEPAT

Subbab ini menguraikan # 10. MEMBUAT FILE AKTIVASI CEPAT. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ------------------------------------------------------------
```

cat > "$HOME/aktifkan-kimia-env.sh" <<'EOF'

```
#!/usr/bin/env bash
# Shortcut aktivasi environment kimia komputasi
# author: Kasmui
# Telp.: 0818294312
```

source "$HOME/miniforge3/etc/profile.d/conda.sh" 2>/dev/null || source "$(conda info --base)/etc/profile.d/conda.sh"

```
echo "Environment tersedia:"
echo "1. kimia-qc          : kimia kuantum + cheminformatics"
echo "2. kimia-md          : molecular dynamics"
echo "3. kimia-material    : material science"
echo "4. kimia-ml          : AI/ML kimia"
echo "5. kimia-workflow    : workflow high-throughput"
echo "6. kimia-qcomp       : quantum computing chemistry"
echo "7. kimia-all-lite    : praktikum umum gabungan"
```

echo

read -rp "Pilih nomor environment [1-7]: " pilih

case "$pilih" in

1) conda activate kimia-qc ;;

2) conda activate kimia-md ;;

3) conda activate kimia-material ;;

4) conda activate kimia-ml ;;

5) conda activate kimia-workflow ;;

6) conda activate kimia-qcomp ;;

7) conda activate kimia-all-lite ;;

*) echo "Pilihan tidak valid." ;;

esac

EOF

```
chmod +x "$HOME/aktifkan-kimia-env.sh"

# ------------------------------------------------------------
```

11. MEMBUAT SCRIPT TES SEMUA ENVIRONMENT

Subbab ini menguraikan # 11. MEMBUAT SCRIPT TES SEMUA ENVIRONMENT. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ------------------------------------------------------------
```

cat > "$HOME/test-kimia-env.sh" <<'EOF'

```
#!/usr/bin/env bash
# Tes singkat semua environment kimia komputasi
# author: Kasmui
# Telp.: 0818294312
```

set -e

source "$HOME/miniforge3/etc/profile.d/conda.sh" 2>/dev/null || source "$(conda info --base)/etc/profile.d/conda.sh"

```
echo "============================================================"
echo "TES ENVIRONMENT KIMIA KOMPUTASI"
echo "author: Kasmui"
echo "Telp.: 0818294312"
echo "============================================================"

for ENV in kimia-qc kimia-md kimia-material kimia-ml kimia-workflow kimia-qcomp kimia-all-lite; do
```

echo

```
    echo "------------------------------------------------------------"
    echo "Environment: $ENV"
    echo "------------------------------------------------------------"
    conda run -n "$ENV" python - <<'PY'
import sys
print("Python:", sys.version)
print("Environment OK")
```

PY

done

echo

```
echo "Semua environment dasar berhasil dipanggil."
```

EOF

```
chmod +x "$HOME/test-kimia-env.sh"
```

```
# ------------------------------------------------------------
# 12. CATATAN CUDA / GPU UNTUK PyTorch, MACE, NequIP, SchNetPack
# ------------------------------------------------------------
```

cat > "$HOME/catatan-cuda-kimia-ml.txt" <<'EOF'

CATATAN CUDA / GPU UNTUK ENVIRONMENT kimia-ml

author: Kasmui

Telp.: 0818294312

Script utama memasang PyTorch versi CPU agar aman di semua mesin.

Jika ingin memakai GPU NVIDIA CUDA, lakukan manual:

1. Aktifkan environment:

Subbab ini menguraikan 1. Aktifkan environment:. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
   conda activate kimia-ml
```

2. Hapus PyTorch CPU:

Subbab ini menguraikan 2. Hapus PyTorch CPU:. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
   pip uninstall -y torch torchvision torchaudio
```

3. Pasang PyTorch CUDA sesuai versi driver.

Contoh CUDA 12.1:

```
   pip install torch torchvision torchaudio --index-url
https://download.pytorch.org/whl/cu121
```

4. Cek CUDA:

Subbab ini menguraikan 4. Cek CUDA:. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

python - <<'PY'

```
import torch
print("CUDA tersedia:", torch.cuda.is_available())
print("GPU:", torch.cuda.get_device_name(0) if torch.cuda.is_available() else "Tidak ada")
```

PY

5. Untuk MACE CUDA acceleration, cek dokumentasi MACE terbaru.

Paket utama MACE adalah:

```
   pip install mace-torch
```

6. Untuk NequIP / Allegro, kecocokan versi PyTorch, e3nn, dan CUDA sangat penting.

Jika error, buat environment khusus:

```
   conda create -n kimia-nequip python=3.11 -y
   conda activate kimia-nequip
   pip install torch torchvision torchaudio --index-url
https://download.pytorch.org/whl/cu121
   pip install nequip e3nn
   pip install git+https://github.com/mir-group/allegro.git
```

EOF

```
# ------------------------------------------------------------
```

13. RINGKASAN AKHIR

Subbab ini menguraikan # 13. RINGKASAN AKHIR. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ------------------------------------------------------------
```

echo

```
echo "============================================================"
echo "INSTALASI SELESAI"
echo "============================================================"
```

echo

```
echo "Environment yang dibuat:"
echo "  - kimia-qc"
echo "  - kimia-md"
echo "  - kimia-material"
echo "  - kimia-ml"
echo "  - kimia-workflow"
echo "  - kimia-qcomp"
echo "  - kimia-all-lite"
```

echo

```
echo "Shortcut aktivasi:"
echo "  $HOME/aktifkan-kimia-env.sh"
```

echo

```
echo "Script tes:"
echo "  $HOME/test-kimia-env.sh"
```

echo

```
echo "Catatan CUDA:"
echo "  $HOME/catatan-cuda-kimia-ml.txt"
```

echo

```
echo "Contoh menjalankan JupyterLab:"
echo "  conda activate kimia-all-lite"
echo "  jupyter lab"
```

echo

```
echo "Contoh aktivasi cepat:"
echo "  ~/aktifkan-kimia-env.sh"
```

echo

```
echo "Selesai."
```

Catatan penting sebelum dijalankan

Jangan dipaksa semuanya ke satu environment tunggal kecuali benar-benar perlu. Paket seperti **Psi4, GPAW, AiiDA, atomate2, DeepChem, PyTorch, NequIP, Allegro, MACE, OpenMM**

sering punya dependency berat dan bisa saling bertabrakan. Pembagian environment di atas adalah cara yang lebih stabil.

Untuk pemakaian harian/praktikum, cukup aktifkan:

```
conda activate kimia-all-lite
```

Untuk hitungan DFT/PySCF/Psi4:

```
conda activate kimia-qc
```

Untuk MD/OpenMM:

```
conda activate kimia-md
```

Untuk material/kristal/fonon:

```
conda activate kimia-material
```

Untuk AI kimia:

```
conda activate kimia-ml
```

BAB 2

SENYAWA KECIL ANTI-KANKER PAYUDARA DAN REPRESENTASI MOLEKUL

Pengantar Bab

Bab ini memperkenalkan senyawa kecil yang relevan untuk kajian anti-kanker payudara serta cara merepresentasikannya secara komputasional. Pembahasan diarahkan agar pembaca memahami hubungan antara nama senyawa, struktur, SMILES, format file, dan kesiapan molekul untuk optimasi geometri, docking, atau analisis descriptor.

Senyawa dalam bab ini tidak diposisikan sebagai klaim terapi langsung, tetapi sebagai objek latihan ilmiah untuk membangun model, membandingkan sifat molekul, dan merancang kandidat hibrida secara hati-hati. Setiap representasi struktur perlu dibaca sebagai input awal yang harus diverifikasi sebelum dipakai dalam perhitungan lanjut.

Setelah pengantar ini, pembahasan dilanjutkan dengan orientasi bab, peta kompetensi, uraian materi inti, contoh kerja, dan catatan metodologis yang diperlukan untuk menuntun pembaca dari konsep menuju praktik.

SMILES menjadi pintu masuk penting karena dapat diolah dengan RDKit, Open Babel, DeepChem, dan berbagai library cheminformatics. Akan tetapi, representasi SMILES perlu diperiksa kembali ketika akan digunakan untuk optimasi 3D, docking, atau DFT. Bentuk garam, tautomer, protonasi, stereokimia, dan muatan dapat memengaruhi hasil.

Bab ini memetakan senyawa-senyawa kecil yang relevan dalam terapi atau kajian kanker payudara. Fokus buku bukan memberikan klaim klinis, melainkan menjadikan senyawa sebagai model pembelajaran untuk membaca struktur, SMILES, nama IUPAC, target biologis, dan peluang modifikasi komputasi.

Orientasi Bab

Subbab ini menguraikan Orientasi Bab. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

RISET KIMIA KOMPUTASI SENYAWA HIBRID ANTI KANKER PAYUDARA

Berikut daftar **molekul kecil** yang banyak dipakai/sering muncul dalam terapi atau pencegahan kanker payudara. Daftar obat dasarnya mengikuti obat yang tercantum untuk kanker payudara di NCI/FDA; saya **tidak memasukkan antibodi/biologik** seperti trastuzumab, pertuzumab, pembrolizumab, atau ADC karena tidak cocok direpresentasikan dengan **SMILES sederhana** seperti molekul kecil. (Informasi Kanker Komprehensif)

Catatan: SMILES di bawah umumnya bentuk **parent/free base**, bukan selalu bentuk garam klinis seperti citrate, succinate, ditosylate, maleate, dan sejenisnya.

No	Nama trivial/generik	Nama IUPAC	SMILES
1	**Tamoxifen**	(Z)-2-[4-(1,2-diphenylbut-1-enyl)phenoxy]-N,N-dimethylethanamine	**CC/C(=C(/C1=CC=C(C=C1)OCCN(C)C)\C2=CC=CC=C2)/C3=CC=CC=C3**
2	**Raloxifene**	[6-hydroxy-2-(4-hydroxyphenyl)-benzothiophen-3-yl]-[4-[2-(1-piperidyl)ethoxy]phenyl]-methanone	**O=C(c1c3ccc(O)cc3sc1c2ccc(O)cc2)c5ccc(OCCN4CCCCC4)cc5 (Wikipedia)**
3	**Anastrozole**	2,2'-[5-(1H-1,2,4-triazol-1-ylmethyl)-1,3-phenylene]bis(2-methylpropanenitrile)	**N#CC(C)(C)c1cc(Cn2cncn2)cc(c1)C(C)(C)C#N (Wikipedia)**
4	**Letrozole**	4,4'-((1H-1,2,4-triazol-1-yl)methylene)dibenzonitrile	**N#Cc1ccc(cc1)C(c2ccc(C#N)cc2)n3ncnc3 (Wikipedia)**
5	**Exemestane**	6-methylideneandrosta-1,4-diene-3,17-dione	**O=C\1\C=C/[C@]3(C(=C/1)/C(=C)C[C@H]4[C@@H]2CCC(=O)[C@]2(CC[C@H]34)C)C (Wikipedia)**
6	**Fulvestrant**	(7R,8R,9S,13S,14S,17S)-13-methyl-7-[9-(4,4,5,5,5-pentafluoropentylsulfinyl)nonyl]-6,7,8,9,11,12,14,15,16,17-decahydrocyclopenta[a]phenanthrene-3,17-diol	**C[C@]12CC[C@@H]3c4ccc(O)cc4CC(CCCCCCCCCS(=O)CCCC(F)(F)C(F)(F)F)[C@H]3[C@@H]1CC[C@@H]2O (Wikipedia)**
7	**Elacestrant**	(6R)-6-{2-[ethyl({4-[2-(ethylamino)ethyl]phenyl}methyl)amino]-4-methoxyphenyl}-5,6,7,8-tetrahydronaphthalen-2-ol	**CCNCCC1=CC=C(C=C1)CN(CC)C2=C(C=CC(=C2)OC)C3CCC4=C(C3)C=CC(=C4)O (Wikipedia)**
8	**Palbociclib**	6-acetyl-8-cyclopentyl-5-methyl-2-{[5-(1-piperazinyl)-2-pyridinyl]amino}pyrido[2,3-d]pyrimidin-7(8H)-one	**O=C2N(c1nc(ncc1/C(=C2/C(=O)C)C)Nc3ncc(cc3)N4CCNCC4)C5CCCC5 (Wikipedia)**
9	**Ribociclib**	7-cyclopentyl-N,N-dimethyl-2-{[5-(1-piperazinyl)-2-pyridinyl]amino}-7H-pyrrolo[2,3-d]pyrimidine-6-carboxamide	**CN(C)C(=O)c1cc2cnc(nc2n1C3CCCC3)Nc4ccc(cn4)N5CCNCC5 (Wikipedia)**
10	**Abemaciclib**	N-[5-[(4-ethyl-1-piperazinyl)methyl]-2-pyridinyl]-5-fluoro-4-[4-fluoro-2-methyl-1-(1-methylethyl)-1H-benzimidazol-6-yl]-2-pyrimidinamine	**CCN1CCN(CC1)Cc2ccc(nc2)Nc3ncc(c(n3)c4cc5c(c(c4)F)nc(n5C(C)C)C)F (Wikipedia)**

11	**Capivasertib**	4-amino-N-[(1S)-1-(4-chlorophenyl)-3-hydroxypropyl]-1-(7H-pyrrolo[2,3-d]pyrimidin-4-yl)piperidine-4-carboxamide	**NC1(CCN(CC1)C1=C2C=CNC2=NC=N1)C(=O)N[C@@H](CCO)C1=CC=C(Cl)C=C1 (Wikipedia)**
12	**Alpelisib**	(2S)-1-N-[4-methyl-5-[2-(1,1,1-trifluoro-2-methylpropan-2-yl)pyridin-4-yl]-1,3-thiazol-2-yl]pyrrolidine-1,2-dicarboxamide	**CC1=C(SC(=N1)NC(=O)N2CCCC2C(=O)N)C3=CC(=NC=C3)C(C)(C)C(F)(F)F (Wikipedia)**
13	**Inavolisib**	(2S)-2-[[2-[(4S)-4-(difluoromethyl)-2-oxo-1,3-oxazolidin-3-yl]-5,6-dihydroimidazo[1,2-d][1,4]benzoxazepin-9-yl]amino]propanamide	**C[C@@H](C(=O)N)NC1=CC2=C(C=C1)C3=NC(=CN3CCO2)N4[C@@H](COC4=O)C(F)F (Wikipedia)**
14	**Lapatinib**	N-[3-chloro-4-[(3-fluorophenyl)methoxy]phenyl]-6-[5-[(2-methylsulfonylethylamino)methyl]-2-furyl]quinazolin-4-amine	**CS(=O)(=O)CCNCc1ccc(o1)c2ccc3c(c2)c(ncn3)Nc4ccc(c(c4)Cl)OCc5cccc(c5)F (Wikipedia)**
15	**Neratinib**	(2E)-N-[4-[[3-chloro-4-[(pyridin-2-yl)methoxy]phenyl]amino]-3-cyano-7-ethoxyquinolin-6-yl]-4-(dimethylamino)but-2-enamide	**CCOc1cc2ncc(C#N)c(Nc3ccc(OCc4cccn4)c(Cl)c3)c2cc1NC(=O)C=CCN(C)C (Wikipedia)**
16	**Tucatinib**	N4-[4-([1,2,4]triazolo[1,5-a]pyridin-7-yloxy)-3-methylphenyl]-N6-(4,4-dimethyl-4,5-dihydrooxazol-2-yl)quinazoline-4,6-diamine	**CC1=C(C=CC(=C1)NC2=NC=NC3=C2C=C(C=C3)NC4=NC(CO4)(C)C)OC5=CC6=NC=NN6C=C5 (Wikipedia)**
17	**Olaparib**	4-[(3-[(4-cyclopropylcarbonyl)piperazin-1-yl]carbonyl)-4-fluorophenyl]methyl(2H)phthalazin-1-one	**c4cccc2c4c(n[nH]c2=O)Cc(ccc1F)cc1C(=O)N3CCN(CC3)C(=O)C5CC5 (Wikipedia)**
18	**Talazoparib**	(8S,9R)-5-fluoro-8-(4-fluorophenyl)-9-(1-methyl-1H-1,2,4-triazol-5-yl)-2,7,8,9-tetrahydro-3H-pyrido[4,3,2-de]phthalazin-3-one	**Cn1c(ncn1)[C@H]2c3c4c(cc(cc4N[C@@H]2c5ccc(cc5)F)F)c(=O)[nH]n3 (Wikipedia)**
19	**Capecitabine**	pentyl [1-(3,4-dihydroxy-5-methyltetrahydrofuran-2-yl)-5-fluoro-2-oxo-1H-pyrimidin-4-yl]carbamate	**FC=1\C(=N/C(=O)N(C=1)[C@@H]2O[C@@H]([C@@H](O)[C@H]2O)C)\NC(=O)OCCCCC (Wikipedia)**
20	**5-Fluorouracil / Fluorouracil**	5-fluoro-1H,3H-pyrimidine-2,4-dione	**O=C1NC(=O)NC=C1F (Wikipedia)**
21	**Cyclophosphamide**	(RS)-N,N-bis(2-chloroethyl)-1,3,2-oxazaphosphinan-2-amine 2-oxide	**O=P1(OCCCN1)N(CCCl)CCCl (Wikipedia)**
22	**Gemcitabine**	4-amino-1-(2-deoxy-2,2-difluoro-β-D-erythro-pentofuranosyl)pyrimidin-2(1H)-one	**c1cn(c(=O)nc1N)[C@H]2C([C@@H]([C@H](O2)CO)O)(F)F (Wikipedia)**
23	**Paclitaxel**	(2α,4α,5β,7β,10β,13α)-4,10-bis(acetyloxy)-13-{[(2R,3S)-3-(benzoylamino)-2-hydroxy-3-phenylpropanoyl]oxy}-1,7-dihydroxy-9-oxo-5,20-epoxytax-11-en-2-yl benzoate	**CC1=C2[C@@]([C@]([C@H]([C@@H]3[C@]4([C@H](OC4)C[C@@H]([C@]3(C(=O)[C@@H]2OC(=O)C)C)O)OC(=O)C)OC(=O)c5ccccc5)(C[C@@H]1OC(=O)[C@H](O)[C@@H](NC(=O)c6ccccc6)c7ccccc7)O)(C)C (Wikipedia)**
24	**Docetaxel**	(1S,2S,3R,4S,7R,9S,10S,12R,15S)-4-acetyloxy-15-{[(2R,3S)-3-[(tert-butoxycarbonyl)amino]-2-hydroxy-3-phenylpropanoyl]oxy}-1,9,12-trihydroxy-10,14,17,17-tetramethyl-11-oxo-6-	**CC1=C2[C@@H]([C@]3([C@H]([C@@H]([C@]([C@@]([C@@H]1OC(=O)[C@@H]([C@H](C4=CC=CC=C4)NC(=O)OC(C)(C)C)O)O)OC(=O)C)(C)C2=O)O)OC(=O)C5=CC=CC=C5)O (Wikipedia)**

		oxatetracyclo[11.3.1.0³,¹⁰.0⁴,⁷]heptadec-13-en-2-yl benzoate	
25	**Doxorubicin**	(7S,9S)-7-[(2R,4S,5S,6S)-4-amino-5-hydroxy-6-methyloxan-2-yl]oxy-6,9,11-trihydroxy-9-(2-hydroxyacetyl)-4-methoxy-8,10-dihydro-7H-tetracene-5,12-dione	**C[C@H]1[C@H]([C@H](C[C@@H](O1)O[C@H]2C[C@@](CC3=C2C(=C4C(=C3O)C(=O)C5=C(C4=O)C(=CC=C5)OC)O)(C(=O)CO)O)N)O (Wikipedia)**
26	**Methotrexate**	(2S)-2-[(4-{[(2,4-diaminopteridin-6-yl)methyl](methyl)amino}benzoyl)amino]pentanedioic acid	**O=C([C@H](CCC(O)=O)NC(C1=CC=C(N(CC2=CN=C(N=C(N)N=C3N)C3=N2)C)C=C1)=O)O (Wikipedia)**

Prioritas praktis untuk komputasi molekul/docking kanker payudara: tamoxifen, anastrozole, letrozole, exemestane, fulvestrant, palbociclib, ribociclib, abemaciclib, lapatinib, tucatinib, olaparib, talazoparib, capivasertib, dan alpelisib. Obat biologik tetap penting secara klinis, tetapi tidak ideal dimasukkan ke daftar SMILES molekul kecil.

Perintah: python tamoxifen.py –job optimize --method B3LYP --basis def2-SVP

Berikut saya susun **20 judul riset skripsi S1 Kimia Komputasi** berbasis senyawa kanker payudara yang ada dalam dokumen Bapak, terutama senyawa prioritas seperti **tamoxifen, anastrozole, letrozole, exemestane, fulvestrant, palbociclib, ribociclib, abemaciclib, lapatinib, tucatinib, olaparib, talazoparib, capivasertib, dan alpelisib**. Dokumen Bapak juga memuat daftar lengkap 26 molekul kecil beserta IUPAC dan SMILES, sehingga semua topik di bawah dapat langsung dikembangkan menjadi struktur 3D, docking, DFT, ADMET, atau QSAR.

Catatan penting: agar **realistis S1 selesai** sekitar **4–8 bulan** per mahasiswa, selama data protein, ligan, dan perangkat lunak sudah tersedia.

---

1. Studi Komputasi Interaksi Tamoxifen terhadap Estrogen Receptor Alpha sebagai Kandidat Terapi Kanker Payudara Hormonal

Riset ini berfokus pada **Tamoxifen** sebagai antiestrogen klasik pada kanker payudara ER-positif. Mahasiswa dapat melakukan optimasi geometri Tamoxifen, analisis HOMO-LUMO, MEP, momen dipol, serta docking ke **estrogen receptor alpha/ERα**, misalnya struktur PDB 3ERT atau 5TOA. Hasil docking dibandingkan dengan posisi ligan asli dan dianalisis residu pentingnya, seperti ikatan hidrogen, interaksi hidrofobik, dan π-π stacking.

Software yang digunakan: **Avogadro, RDKit, PySCF atau ORCA, AutoDock Vina/PyRx, AutoDockTools, Discovery Studio Visualizer, PyMOL, SwissADME, pkCSM**.

Metode komputasi: **optimasi geometri DFT, analisis HOMO-LUMO, molecular electrostatic potential, molecular docking, validasi docking redocking, ADMET sederhana**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP** untuk optimasi awal; **B3LYP/6-31G(d,p)** atau **M06-2X/6-31G(d,p)** untuk validasi energi elektronik ligan.

Referensi jurnal: studi analog Tamoxifen menggunakan kombinasi **molecular docking dan DFT** terhadap reseptor hormon dapat dijadikan rujukan metodologis utama. (Frontiers)

---

2. Perbandingan Aktivitas Komputasi Tamoxifen, Raloxifene, Fulvestrant, dan Elacestrant sebagai Modulator Reseptor Estrogen pada Kanker Payudara

Judul ini cocok untuk mahasiswa S1 karena memakai beberapa ligan yang masih dalam satu jalur mekanisme, yaitu terapi berbasis estrogen receptor. Fokus riset adalah membandingkan kekuatan afinitas, pola interaksi, sifat elektronik, dan prediksi ADMET dari **Tamoxifen, Raloxifene, Fulvestrant, dan Elacestrant**. Karena Fulvestrant cukup besar dan fleksibel, mahasiswa dapat membatasi analisis DFT pada ligan saja, lalu docking dilakukan pada ERα.

Software yang digunakan: **RDKit, Open Babel, Avogadro, PySCF/ORCA, AutoDock Vina, PyRx, Discovery Studio Visualizer, LigPlot+, SwissADME**.

Metode komputasi: **DFT ligan, docking komparatif, analisis interaksi residu, ADMET, drug-likeness, bioavailability radar**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP** untuk semua ligan; untuk Tamoxifen dan Raloxifene dapat ditambah **6-31G(d,p)** sebagai pembanding.

Referensi jurnal: artikel QSAR, docking, dan molecular dynamics untuk ERα binders serta studi analog Fulvestrant mendukung arah riset ini. (PubMed)

---

3. Desain Turunan Sederhana Tamoxifen Berbasis Substituen Hidroksi, Metoksi, dan Halogen melalui DFT, Docking, dan Prediksi ADMET

Mahasiswa membuat 8–12 turunan sederhana Tamoxifen, misalnya substitusi **–OH, –OCH_3, –F, –Cl, –CN** pada cincin aromatik. Tujuannya adalah melihat apakah substituen tertentu meningkatkan interaksi dengan ERα dan memperbaiki sifat elektronik. Judul ini sangat cocok untuk S1 karena tidak perlu sintesis, hanya desain molekul, optimasi, docking, dan ADMET.

Software yang digunakan: **Avogadro, RDKit, Open Babel, ORCA/PySCF, AutoDock Vina, SwissADME, pkCSM, DataWarrior**.

Metode komputasi: **desain analog in silico, optimasi geometri DFT, descriptor elektronik, docking, ADMET, analisis hubungan substituen-aktivitas secara kualitatif**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; jika komputer kuat, validasi beberapa kandidat terbaik dengan **B3LYP/6-311G(d,p)**.

Referensi jurnal: riset desain analog Tamoxifen terbaru memakai integrasi **QSAR, ADME, docking, dan MD**, sedangkan studi lain memakai docking dan DFT untuk analog Tamoxifen. (ScienceDirect)

---

4. Studi Molecular Docking dan DFT Inhibitor Aromatase: Anastrozole, Letrozole, dan Exemestane pada Target CYP19A1

Topik ini sangat sesuai untuk skripsi karena hanya membandingkan tiga molekul utama inhibitor aromatase. Mahasiswa dapat mengkaji perbedaan inhibitor nonsteroid (**Anastrozole dan Letrozole**) dengan inhibitor steroidal (**Exemestane**) terhadap aromatase. Target protein yang umum dipakai adalah **aromatase/CYP19A1**.

Software yang digunakan: **Avogadro, RDKit, AutoDock Vina/PyRx, AutoDockTools, Discovery Studio, PyMOL, ORCA/PySCF**.

Metode komputasi: **optimasi geometri DFT, docking, analisis residu aktif, interaksi dengan gugus heme aromatase, ADMET sederhana**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP** untuk semua ligan; untuk ligan kecil seperti Anastrozole dan Letrozole dapat digunakan **B3LYP/6-31G(d,p)**.

Referensi jurnal: studi docking inhibitor aromatase pada struktur kristal aromatase menunjukkan pendekatan ini relevan dan sudah mapan untuk inhibitor generasi berbeda. (PMC)

---

5. Perancangan Analog Letrozole sebagai Inhibitor Aromatase melalui Quantum Docking, DFT, dan Molecular Dynamics Pendek

Mahasiswa dapat mendesain analog Letrozole dengan modifikasi cincin aromatik atau gugus triazol, lalu memilih 3–5 kandidat terbaik dari docking untuk dianalisis DFT dan MD pendek. Topik ini dapat dibuat sederhana: MD cukup **10 ns** untuk dua kompleks terbaik agar sesuai kemampuan S1.

Software yang digunakan: **Avogadro, RDKit, AutoDock Vina, GROMACS, ACPYPE/GAFF, ORCA/PySCF, SwissADME**.

Metode komputasi: **docking, DFT ligan, MD pendek 10 ns, RMSD, RMSF, radius of gyration, hydrogen bond occupancy, ADMET**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; untuk kandidat terbaik bisa ditambah **ωB97X-D/def2-SVP** bila memakai ORCA.

Referensi jurnal: desain analog Letrozole berbasis **quantum docking, molecular dynamics, dan studi teoretis** sudah dilaporkan dan dapat dijadikan model metodologi. (MDPI)

---

6. Analisis QSAR Sederhana Inhibitor Aromatase Steroidal dan Nonsteroidal Menggunakan Descriptor Molekul dan Docking Score

Riset ini menggabungkan **QSAR sederhana** dan docking. Mahasiswa mengumpulkan data inhibitor aromatase dari literatur atau ChEMBL, lalu membangun model regresi sederhana menggunakan descriptor seperti logP, TPSA, molecular weight, jumlah donor/akseptor H, HOMO-LUMO gap, dan docking score. Anastrozole, Letrozole, dan Exemestane dijadikan pembanding obat standar.

Software yang digunakan: **PaDEL-Descriptor, RDKit, DataWarrior, KNIME, Orange, Python scikit-learn, AutoDock Vina**.

Metode komputasi: **2D-QSAR sederhana, multiple linear regression/random forest, docking, validasi internal, analisis descriptor penting**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP** untuk descriptor elektronik subset molekul; untuk QSAR murni 2D tidak wajib memakai basis set.

Referensi jurnal: studi QSAR dan docking inhibitor aromatase telah digunakan untuk menjelaskan pengaruh descriptor terhadap aktivitas inhibitor aromatase. (ScienceDirect)

7. Perbandingan Palbociclib, Ribociclib, dan Abemaciclib sebagai Inhibitor CDK4/6 melalui Molecular Docking dan Analisis DFT

Topik ini membandingkan tiga inhibitor CDK4/6 yang sangat penting pada kanker payudara HR+/HER2−. Mahasiswa dapat melakukan docking ke **CDK4** dan **CDK6**, kemudian membandingkan residu kunci, energi ikat, dan sifat elektronik ketiga obat. Riset ini dapat dibuat sangat rapi karena hanya memakai tiga ligan utama.

Software yang digunakan: **RDKit, Avogadro, AutoDock Vina/PyRx, PyMOL, Discovery Studio, ORCA/PySCF, SwissADME**.

Metode komputasi: **DFT, docking CDK4/CDK6, analisis interaksi ATP-binding pocket, ADMET, perbandingan descriptor elektronik**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP** karena ukuran Palbociclib, Ribociclib, dan Abemaciclib cukup besar; validasi single-point dapat memakai **B3LYP-D3BJ/def2-TZVP** pada kandidat tertentu jika komputer memadai.

Referensi jurnal: CDK4/6 inhibitor seperti Palbociclib, Ribociclib, dan Abemaciclib telah banyak dibahas dalam terapi kanker payudara, dan review farmakologi/komputasi dapat menjadi landasan pemilihan target. (PMC)

---

8. Kajian ADMET dan Drug-Likeness Palbociclib, Ribociclib, dan Abemaciclib Berbasis Descriptor RDKit dan Prediksi Farmakokinetik

Topik ini lebih ringan daripada docking protein besar. Fokusnya pada perbandingan sifat molekuler dan farmakokinetik: berat molekul, logP, TPSA, rotatable bonds, HIA, BBB, CYP inhibition, hepatotoxicity, dan toxicological alerts. Cocok untuk mahasiswa S1 yang baru belajar cheminformatics.

Software yang digunakan: **RDKit, DataWarrior, SwissADME, pkCSM, ProTox-II, Excel/LibreOffice Calc, Python**.

Metode komputasi: **cheminformatics, descriptor molekul, ADMET prediction, radar bioavailability, BOILED-Egg, analisis Lipinski/Ghose/Veber**.

Basis set yang disarankan: tidak wajib untuk ADMET murni; bila ditambah DFT, gunakan **B3LYP-D3BJ/def2-SVP** untuk HOMO-LUMO dan momen dipol.

Referensi jurnal: perbedaan metabolisme, transport, resistensi, efektivitas, dan keamanan tiga CDK4/6 inhibitor telah dibahas dalam kajian farmakologis komprehensif. (Frontiers)

---

9. Studi Komputasi Olaparib dan Talazoparib sebagai Inhibitor PARP-1 pada Kanker Payudara BRCA-Mutated

Mahasiswa meneliti **Olaparib dan Talazoparib** sebagai inhibitor PARP-1. Analisis diarahkan pada perbandingan binding pocket PARP-1, interaksi dengan residu donor/akseptor, perbedaan bentuk molekul, dan stabilitas kompleks. Topik ini sangat relevan untuk kanker payudara dengan mutasi BRCA.

Software yang digunakan: **Avogadro, RDKit, AutoDock Vina, Discovery Studio, PyMOL, ORCA/PySCF, SwissADME**.

Metode komputasi: **DFT ligan, docking PARP-1, analisis interaksi residu, ADMET, perbandingan sifat elektronik**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; single-point kandidat terbaik dapat memakai **M06-2X/def2-SVP**.

Referensi jurnal: riset komputasi PARP-1 inhibitor untuk kanker payudara telah memakai workflow docking, molecular dynamics, MM/GBSA, DFT, dan ADMET, dengan Olaparib dan Talazoparib sebagai pembanding klinis. (MDPI)

---

10. Perbandingan Inhibitor HER2: Lapatinib, Neratinib, dan Tucatinib melalui DFT, Docking, dan Prediksi ADMET

Topik ini meneliti tiga inhibitor tirosin kinase HER2 berbasis molekul kecil. Mahasiswa dapat membandingkan docking ke **HER2 kinase domain**, lalu melihat apakah perbedaan substituen memengaruhi afinitas dan interaksi. Karena semua ligan cukup besar, DFT cukup dilakukan untuk optimasi ligan dan analisis orbital.

Software yang digunakan: **AutoDock Vina/PyRx, AutoDockTools, PyMOL, Discovery Studio, RDKit, Avogadro, ORCA/PySCF, SwissADME**.

Metode komputasi: **DFT ligan, molecular docking HER2, analisis interaksi ATP-binding site, ADMET, perbandingan HOMO-LUMO**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; single-point energi dapat memakai **def2-TZVP** untuk satu ligan terbaik.

Referensi jurnal: studi computational profiling melaporkan Lapatinib, Tucatinib, dan Neratinib sebagai molekul dengan afinitas tinggi pada target HER2 dan protein terkait; studi HER2 lain juga menggunakan Lapatinib sebagai ligan pembanding. (RSC Publishing)

---

11. Analisis Molecular Docking Lapatinib sebagai Kontrol Positif untuk Skrining Senyawa Kandidat Anti-HER2 pada Kanker Payudara

Judul ini lebih sederhana: mahasiswa memakai **Lapatinib** sebagai kontrol positif, lalu membandingkan 10–20 kandidat senyawa lain dari database atau bahan alam. Fokus riset bukan menemukan obat baru final, tetapi membuktikan pipeline docking, validasi, dan pemeringkatan kandidat.

Software yang digunakan: **PyRx, AutoDock Vina, AutoDockTools, PubChem, SwissADME, Discovery Studio, PyMOL**.

Metode komputasi: **virtual screening kecil, molecular docking, redocking, RMSD validasi, ADMET, analisis interaksi 2D/3D**.

Basis set yang disarankan: docking tidak memakai basis set; untuk kandidat terbaik, optimasi ligan memakai **B3LYP-D3BJ/def2-SVP** atau **PM7** bila komputer terbatas.

Referensi jurnal: riset in silico HER2 banyak menggunakan Lapatinib sebagai pembanding, termasuk studi natural compounds, lutein, glabrene, dan inhibitor HER2 lainnya. (PMC)

---

12. Studi Komputasi Alpelisib dan Inavolisib sebagai Inhibitor PI3Kα pada Kanker Payudara PIK3CA-Mutated

Topik ini cocok untuk mahasiswa yang ingin fokus pada jalur **PI3K/AKT/mTOR**. Ligan utama adalah **Alpelisib** dan **Inavolisib**. Targetnya adalah PI3Kα, khususnya domain katalitik. Riset dapat diarahkan pada perbandingan docking, profil ADMET, dan descriptor elektronik.

Software yang digunakan: **Avogadro, RDKit, AutoDock Vina, PyMOL, Discovery Studio, ORCA/PySCF, SwissADME, pkCSM**.

Metode komputasi: **DFT, docking PI3Kα, ADMET, analisis interaksi hidrogen dan hidrofobik, perbandingan pose docking**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; untuk gugus yang mengandung F dapat memakai **def2-SVP** karena basis ini cukup baik untuk unsur ringan dan halogen.

Referensi jurnal: Alpelisib dikenal sebagai PI3K inhibitor yang relevan untuk HR+/HER2− metastatic breast cancer dengan mutasi PIK3CA, dan studi docking jalur PI3K/AKT/mTOR pada kanker payudara memberi dasar biologis untuk topik ini. (ScienceDirect)

---

13. Kajian Capivasertib sebagai Inhibitor AKT pada Kanker Payudara Resisten Endokrin: Docking, DFT, dan ADMET

Mahasiswa meneliti **Capivasertib** sebagai inhibitor AKT, terutama terkait resistensi terapi endokrin. Riset dapat memakai protein AKT1/AKT2 sebagai target docking. Analisis diarahkan pada bagaimana gugus fungsional Capivasertib membentuk interaksi pada ATP-binding pocket.

Software yang digunakan: **AutoDock Vina/PyRx, AutoDockTools, Discovery Studio, PyMOL, RDKit, PySCF/ORCA, SwissADME**.

Metode komputasi: **docking AKT, DFT ligan, analisis MEP, HOMO-LUMO, ADMET, drug-likeness**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; untuk validasi kandidat terbaik dapat memakai **B3LYP-D3BJ/def2-TZVP** pada single-point.

Referensi jurnal: Capivasertib telah dibahas sebagai agen terapi kanker payudara, termasuk kombinasi dengan Fulvestrant untuk mengatasi resistensi endokrin pada HR+/HER2− dengan perubahan PIK3CA/AKT1/PTEN. (MDPI)

---

14. Perbandingan Fulvestrant, Alpelisib, dan Capivasertib pada Jalur ER–PI3K–AKT Menggunakan Pendekatan Multi-Target Docking

Topik ini menghubungkan terapi endokrin dan jalur resistensi. Fulvestrant diarahkan ke ERα, Alpelisib ke PI3Kα, dan Capivasertib ke AKT. Mahasiswa tidak perlu melakukan simulasi sistem biologis penuh; cukup membuat **multi-target docking workflow** dan membandingkan masing-masing obat pada target utamanya.

Software yang digunakan: **PyRx, AutoDock Vina, PyMOL, Discovery Studio, RDKit, SwissADME, pkCSM**.

Metode komputasi: **multi-target docking, ADMET, clustering ligan-target, analisis mekanistik kualitatif**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP** untuk optimasi ligan; docking protein tidak memakai basis set.

Referensi jurnal: literatur Capivasertib menegaskan relevansi jalur PI3K/AKT dalam resistensi endokrin, sedangkan studi analog Fulvestrant menunjukkan pendekatan docking, MD, dan ADMET dapat dipakai untuk ER+ breast cancer. (PMC)

---

15. Studi DFT dan Molecular Docking Paclitaxel dan Docetaxel pada β-Tubulin sebagai Agen Antimitotik Kanker Payudara

Judul ini memakai dua obat taxane besar, yaitu **Paclitaxel dan Docetaxel**. Karena ukurannya besar, DFT dapat dibuat terbatas pada optimasi struktur dan analisis orbital. Docking dilakukan pada β-tubulin atau kompleks tubulin-taxane. Mahasiswa menilai residu yang berperan dalam stabilisasi mikrotubulus.

Software yang digunakan: **Avogadro, Open Babel, AutoDock Vina, PyMOL, Discovery Studio, ORCA/PySCF, SwissADME**.

Metode komputasi: **optimasi geometri DFT ringan, docking β-tubulin, analisis interaksi hidrofobik dan H-bond, ADMET deskriptif**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; bila komputasi berat, gunakan **GFN2-xTB** untuk optimasi awal lalu single-point **B3LYP-D3BJ/def2-SVP**.

Referensi jurnal: Paclitaxel merupakan kemoterapi yang digunakan pada berbagai kanker termasuk kanker payudara; referensi farmakologis dan komputasi terkait taxane dapat dipakai sebagai dasar pemilihan target mikrotubulus. (NCBI)

---

16. Analisis Reaktivitas Elektronik Doxorubicin melalui DFT dan Docking terhadap Topoisomerase II sebagai Model Kemoterapi Kanker Payudara

Riset ini mempelajari **Doxorubicin**, salah satu obat kemoterapi penting. Kajian dapat difokuskan pada HOMO-LUMO, MEP, potensi redoks relatif, dan interaksi docking dengan topoisomerase II atau model DNA-binding pocket. Karena struktur Doxorubicin cukup besar, mahasiswa dapat melakukan optimasi xTB dahulu.

Software yang digunakan: **xTB, ORCA/PySCF, Avogadro, AutoDock Vina, PyMOL, Discovery Studio, SwissADME**.

Metode komputasi: **GFN2-xTB pre-optimization, DFT single-point, MEP, HOMO-LUMO, docking, analisis interaksi DNA/protein**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; untuk studi elektronik lanjutan dapat memakai **M06-2X/def2-SVP**.

Referensi jurnal: studi doxorubicin telah banyak memakai simulasi molekuler, termasuk molecular dynamics dan free energy pada sistem pembawa lipid, serta kajian docking doxorubicin dengan ERα sebagai pembanding pendekatan in silico. (PMC)

---

17. Studi Komputasi Capecitabine, 5-Fluorouracil, dan Gemcitabine sebagai Antimetabolit: DFT, Docking, dan ADMET

Mahasiswa membandingkan tiga antimetabolit: **Capecitabine, 5-Fluorouracil, dan Gemcitabine**. Target yang dapat digunakan antara lain thymidylate synthase untuk 5-FU dan target enzim metabolisme nukleosida untuk Gemcitabine. Topik ini baik untuk memahami hubungan struktur nukleosida dengan reaktivitas elektronik.

Software yang digunakan: **Avogadro, RDKit, PySCF/ORCA, AutoDock Vina, Discovery Studio, SwissADME, pkCSM**.

Metode komputasi: **DFT, HOMO-LUMO, MEP, docking enzim target, ADMET, analisis kemiripan struktur nukleobasa**.

Basis set yang disarankan: **B3LYP/6-31G(d,p)** untuk 5-FU dan Gemcitabine; **B3LYP-D3BJ/def2-SVP** untuk Capecitabine.

Referensi jurnal: kajian umum komputasi kanker payudara dan docking multi-target dapat digunakan untuk mendukung pemilihan antimetabolit sebagai ligan pembanding dalam skrining in silico. (arXiv)

---

18. Kajian DFT dan Molecular Docking Methotrexate terhadap Dihydrofolate Reductase sebagai Model Antifolat Antikanker

Topik ini klasik dan cocok untuk S1 karena target **dihydrofolate reductase/DHFR** sangat dikenal. Mahasiswa mengkaji Methotrexate dari sisi struktur, muatan parsial, MEP, donor/akseptor hidrogen, dan docking ke DHFR. Penelitian dapat diperluas dengan membandingkan folat atau analog sederhana.

Software yang digunakan: **Avogadro, RDKit, AutoDock Vina, PyMOL, Discovery Studio, PySCF/ORCA, SwissADME**.

Metode komputasi: **DFT, docking DHFR, analisis interaksi residu, ADMET, perbandingan dengan ligan native**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; untuk fragmen kecil dapat memakai **6-31G(d,p)**.

Referensi jurnal: studi molecular docking, ADMET, dan virtual screening pada target kanker dapat dijadikan rujukan metodologis, terutama untuk penggunaan docking sebagai platform pemahaman mekanisme pengikatan ligan. (Ubaya Repository)

---

19. Virtual Screening Multi-Target 26 Obat Kecil Kanker Payudara terhadap ERα, HER2, Aromatase, CDK6, PARP-1, dan PI3Kα

Ini judul paling komprehensif tetapi tetap bisa untuk S1 jika dibatasi hanya pada docking dan ADMET. Mahasiswa memakai 26 molekul dari dokumen Bapak, lalu melakukan docking ke 5–6 target utama kanker payudara. Outputnya berupa ranking ligan-target dan peta kecocokan: obat mana paling sesuai untuk ERα, HER2, aromatase, CDK6, PARP-1, dan PI3Kα.

Software yang digunakan: **RDKit, Open Babel, PyRx, AutoDock Vina, PyMOL, Discovery Studio, SwissADME, Python pandas/matplotlib**.

Metode komputasi: **multi-target docking, ADMET, clustering sederhana, heatmap docking score, analisis residu interaksi**.

Basis set yang disarankan: tidak wajib untuk semua 26 ligan; optimasi awal bisa memakai **MMFF94 atau UFF**, lalu 5 ligan terbaik dioptimasi dengan **B3LYP-D3BJ/def2-SVP**.

Referensi jurnal: studi skrining ligan terhadap target EGFR, HER2, estrogen receptor, progesterone receptor, dan NF-κB menunjukkan pendekatan multi-target docking relevan untuk subtipe kanker payudara. (arXiv)

---

20. Pengembangan Model QSAR dan Machine Learning Sederhana untuk Mengelompokkan Obat Kanker Payudara Berdasarkan Target Molekulernya

Topik ini cocok untuk mahasiswa yang kuat Python. Dataset awal dapat memakai 26 molekul dalam dokumen Bapak, lalu diperluas dengan 100–300 molekul dari ChEMBL atau BindingDB. Target kelas dapat berupa **ER inhibitor, aromatase inhibitor, HER2 inhibitor, CDK4/6 inhibitor, PARP inhibitor, PI3K/AKT inhibitor**. Output riset berupa model klasifikasi sederhana.

Software yang digunakan: **RDKit, PaDEL-Descriptor, KNIME, Python, scikit-learn, pandas, matplotlib, seaborn jika di luar laporan visual, DataWarrior**.

Metode komputasi: **descriptor calculation, molecular fingerprints, PCA, random forest, SVM, k-nearest neighbors, validasi train-test split, confusion matrix, feature importance**.

Basis set yang disarankan: untuk QSAR/ML tidak wajib memakai basis set; bila ingin menambah descriptor kuantum, hitung HOMO, LUMO, gap, dan momen dipol dengan **B3LYP-D3BJ/def2-SVP** untuk subset molekul.

Referensi jurnal: penelitian virtual screening berbasis machine learning dan docking terhadap target kanker payudara telah memakai klasifikasi aktif/tidak aktif dan target prediction; studi QSAR estrogen receptor juga menunjukkan descriptor dan fingerprint dapat dipakai untuk prediksi aktivitas ER. (arXiv)

Rekomendasi judul paling aman untuk skripsi S1

Untuk mahasiswa S1 yang waktu dan perangkat komputasinya terbatas, saya sarankan memilih salah satu dari lima topik berikut:

Pertama, Tamoxifen terhadap ERα, karena target dan literatur sangat kuat.

Kedua, Anastrozole–Letrozole–Exemestane terhadap aromatase, karena hanya tiga ligan dan targetnya jelas.

Ketiga, Palbociclib–Ribociclib–Abemaciclib terhadap CDK4/6, karena relevan secara klinis dan bagus untuk komparasi.

Keempat, Olaparib–Talazoparib terhadap PARP-1, karena hanya dua ligan tetapi pembahasannya kuat.

Kelima, Lapatinib–Neratinib–Tucatinib terhadap HER2, karena cocok untuk skripsi docking dan DFT obat target terapi.

Untuk standar metode skripsi S1, format yang paling ideal adalah:

BAB 3

DESAIN JUDUL DAN ARAH PENGEMBANGAN RISET

Pengantar Bab

Bab ini mengubah daftar senyawa menjadi gagasan penelitian yang terarah. Fokus utama bab adalah membantu mahasiswa memilih judul yang realistis, memiliki pertanyaan ilmiah yang jelas, dan dapat diselesaikan dengan sumber daya komputasi yang tersedia pada level S1.

Setiap judul dipahami sebagai pintu masuk menuju rancangan riset, bukan sebagai kesimpulan hasil. Karena itu, pembaca perlu memperhatikan keterkaitan antara target biologis, struktur senyawa, metode komputasi, parameter yang diukur, serta batas interpretasi dari data in silico.

Setelah pengantar ini, pembahasan dilanjutkan dengan orientasi bab, peta kompetensi, uraian materi inti, contoh kerja, dan catatan metodologis yang diperlukan untuk menuntun pembaca dari konsep menuju praktik.

Optimasi ligan: Avogadro/RDKit → MMFF94 → DFT B3LYP-D3BJ/def2-SVP.
Docking: AutoDock Vina/PyRx dengan validasi redocking.
Analisis: Discovery Studio, PyMOL, LigPlot+.
ADMET: SwissADME, pkCSM, ProTox-II.
Laporan: energi docking, residu interaksi, HOMO-LUMO gap, MEP, logP, TPSA, HBA/HBD, Lipinski, dan prediksi toksisitas.

Judul-judul pengembangan dalam naskah ini dapat dibaca sebagai bank ide. Pembaca tidak harus mengambil seluruhnya, tetapi dapat memilih satu topik yang paling feasible, lalu menurunkannya menjadi proposal, desain molekul, file input, parameter perhitungan, dan format analisis hasil.

Bab ini mengubah daftar senyawa menjadi peluang penelitian. Judul riset yang baik harus memenuhi tiga syarat: masalahnya jelas, metodenya dapat dijalankan dengan sumber daya yang tersedia, dan keluarannya dapat dianalisis secara kimia. Untuk mahasiswa S1, topik yang terlalu luas perlu dibatasi agar tidak berubah menjadi proyek high-throughput yang sulit diselesaikan.

Orientasi Bab

Subbab ini menguraikan Orientasi Bab. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

20 judul tambahan skripsi S1 Kimia Komputasi yang berfokus pada **modifikasi struktur dan desain senyawa hibrida baru** dari molekul kanker payudara yang Bapak lampirkan, seperti **Tamoxifen, Raloxifene, Anastrozole, Letrozole, Exemestane, Fulvestrant, Palbociclib, Ribociclib, Abemaciclib, Lapatinib, Neratinib, Tucatinib, Olaparib, Talazoparib, Capivasertib, Alpelisib, 5-Fluorouracil, Paclitaxel, Docetaxel, Doxorubicin, dan Methotrexate**. Dokumen Bapak sudah memuat nama trivial, IUPAC, dan SMILES dari molekul-molekul tersebut sehingga dapat langsung dijadikan sumber awal pembuatan ligan 3D.

Arah riset di bawah **bukan klaim menemukan obat kanker yang pasti efektif**, tetapi rancangan **kandidat senyawa baru secara in silico**. Konsepnya sesuai dengan strategi **molecular hybridization**, yaitu menggabungkan dua atau lebih farmakofor aktif menjadi senyawa baru tipe **linked hybrid, merged hybrid, atau fused hybrid** untuk memperoleh aktivitas multi-target atau profil farmakokinetik yang lebih baik. Kajian review tentang molecular hybrids untuk kanker payudara menunjukkan bahwa pendekatan hibridisasi molekul memang menjadi salah satu strategi penting dalam desain kandidat anti-breast cancer modern. (PMC)

---

1. Desain Hibrida Tamoxifen–5-Fluorouracil sebagai Kandidat Anti-Kanker Payudara Multi-Mekanisme Berbasis DFT, Docking ERα, dan ADMET

Riset ini bertujuan merancang senyawa hibrida yang menggabungkan farmakofor **Tamoxifen** sebagai modulator reseptor estrogen dengan fragmen **5-Fluorouracil** sebagai antimetabolit. Ide utamanya adalah menguji apakah penggabungan fragmen antiestrogen dan antimetabolit dapat menghasilkan kandidat yang tetap berinteraksi baik dengan **estrogen receptor alpha/ERα**, sekaligus memiliki karakter elektronik yang mendukung reaktivitas biologis. Mahasiswa cukup mendesain 8–12 variasi linker pendek, misalnya ester, amida, eter, dan alkil pendek, lalu memilih 3 kandidat terbaik berdasarkan docking score, interaksi residu, HOMO-LUMO gap, dan ADMET.

Software yang digunakan: **Avogadro, RDKit, Open Babel, PySCF atau ORCA, AutoDock Vina/PyRx, AutoDockTools, Discovery Studio Visualizer, PyMOL, SwissADME, pkCSM, ProTox-II**.

Metode komputasi: **desain farmakofor hibrida, optimasi geometri DFT, molecular electrostatic potential, HOMO-LUMO, molecular docking ERα, prediksi ADMET dan toksisitas awal**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP** untuk optimasi geometri; **B3LYP/6-31G(d,p)** dapat digunakan untuk pembanding pada fragmen kecil; jika struktur terlalu besar, lakukan pre-optimasi dengan **GFN2-xTB** lalu single-point DFT.

Referensi jurnal yang berkaitan: studi teoretis gabungan **5-Fluorouracil dan Tamoxifen** pernah dianalisis dengan pendekatan DFT dan docking, sehingga cocok menjadi dasar ide riset ini. (Taylor & Francis Online)

---

2. Desain Analog Hibrida Tamoxifen Berbasis Substitusi Hidroksi, Amida, Karboksil, dan Sulfhidril untuk Peningkatan Afinitas terhadap ERα

Judul ini lebih aman untuk S1 karena tidak membuat molekul terlalu besar. Mahasiswa mendesain turunan Tamoxifen dengan modifikasi gugus hidrofilik, misalnya mengganti atau menambah gugus **–OH, –$CONH_2$, –COOH, –SH, –OCH_3, dan –F** pada cincin aromatik atau rantai samping. Tujuannya untuk melihat substituen mana yang memperkuat interaksi dengan kantong aktif ERα dan memperbaiki polaritas molekul tanpa merusak bentuk farmakofor utama Tamoxifen.

Software yang digunakan: **Avogadro, GaussView bila tersedia, RDKit, ORCA/PySCF, AutoDock Vina, Discovery Studio, PyMOL, SwissADME**.

Metode komputasi: **DFT, docking ERα, analisis substituen, MEP, HOMO-LUMO, logP, TPSA, Lipinski, dan analisis residu ikatan**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; untuk kandidat terbaik dapat dilakukan single-point **M06-2X/def2-SVP**.

Referensi jurnal yang berkaitan: analog Tamoxifen dengan modifikasi gugus hidrofilik telah dipelajari memakai **molecular docking dan DFT** untuk melihat pengaruhnya terhadap interaksi pada hormone receptor. (Frontiers)

---

3. Perancangan Hibrida Tamoxifen–Raloxifene sebagai Kandidat Selective Estrogen Receptor Modulator Baru melalui Docking, DFT, dan ADMET

Riset ini menggabungkan dua kerangka SERM, yaitu **Tamoxifen** dan **Raloxifene**. Ide desainnya bukan menggabungkan seluruh struktur secara utuh, tetapi mengambil farmakofor penting: rantai aminoetoksi dari Tamoxifen dan inti benzotiofen/benzofenon dari Raloxifene. Mahasiswa dapat membuat 8–10 struktur **merged hybrid** dan membandingkannya dengan Tamoxifen serta Raloxifene sebagai kontrol.

Software yang digunakan: **MarvinSketch/ChemDraw, Avogadro, RDKit, ORCA/PySCF, AutoDock Vina, Discovery Studio, SwissADME, pkCSM**.

Metode komputasi: **pharmacophore merging, DFT, docking ERα dan ERβ bila memungkinkan, ADMET, prediksi toksisitas, serta analisis selektivitas interaksi residu**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; untuk struktur besar dapat memakai **GFN2-xTB** sebagai tahap awal.

Referensi jurnal yang berkaitan: review molekul hibrida kanker payudara membahas pentingnya penggabungan farmakofor aktif dalam desain senyawa anti-breast cancer, sedangkan kajian ERα antagonists mendukung relevansi target reseptor estrogen dalam kanker payudara. (PMC)

---

4. Desain Hibrida Tamoxifen–Resveratrol sebagai Kandidat Antagonis ERα Berbasis Farmakofor Fenolik

Riset ini menggabungkan farmakofor antiestrogen Tamoxifen dengan fragmen fenolik dari **Resveratrol**. Tujuannya untuk menghasilkan kandidat yang tetap memiliki karakter antiestrogen tetapi lebih kaya interaksi hidrogen melalui gugus fenolik. Mahasiswa dapat membuat variasi linker eter, ester, dan amida, lalu menilai apakah kandidat hibrida mampu berikatan lebih baik pada ERα dibandingkan Tamoxifen.

Software yang digunakan: **Avogadro, RDKit, Open Babel, PySCF/ORCA, AutoDock Vina, PyMOL, Discovery Studio, SwissADME, ProTox-II**.

Metode komputasi: **DFT, MEP, docking ERα, ADMET, analisis donor/akseptor hidrogen, dan komparasi docking dengan Tamoxifen**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; untuk validasi kandidat terbaik dapat digunakan **ωB97X-D/def2-SVP** bila memakai ORCA.

Referensi jurnal yang berkaitan: turunan Resveratrol berbasis imino dilaporkan memiliki potensi sebagai antagonis ERα untuk kanker payudara, sehingga fragmen fenolik Resveratrol dapat dijadikan inspirasi hibridisasi dengan Tamoxifen. (Springer Nature Link)

---

5. Perancangan Hibrida Letrozole–Anastrozole sebagai Inhibitor Aromatase Baru Berbasis Gugus Triazol dan Nitril

Judul ini sangat cocok untuk S1 karena kedua molekul relatif kecil. Mahasiswa dapat membuat hibrida yang mempertahankan gugus **triazol** dan **nitril** dari Letrozole/Anastrozole, lalu memodifikasi spacer aromatik dan substituen alkil. Target docking adalah **aromatase/CYP19A1**. Fokus analisis adalah kemampuan gugus triazol berinteraksi dengan area heme aromatase dan kontribusi gugus nitril terhadap polaritas serta orientasi ligan.

Software yang digunakan: **Avogadro, RDKit, Open Babel, ORCA/PySCF, AutoDock Vina/PyRx, Discovery Studio, PyMOL, SwissADME**.

Metode komputasi: **desain analog hibrida, DFT, docking aromatase, validasi redocking, ADMET, analisis interaksi dengan residu aktif aromatase**.

Basis set yang disarankan: **B3LYP/6-31G(d,p)** atau **B3LYP-D3BJ/def2-SVP**; karena ukuran molekul tidak terlalu besar, keduanya masih realistis untuk skripsi 6 bulan.

Referensi jurnal yang berkaitan: docking inhibitor aromatase pada generasi inhibitor berbeda telah digunakan untuk menjelaskan mode ikatan obat aromatase, dan desain analog Letrozole juga telah dikembangkan melalui quantum docking, MD, serta kajian teoretis. (PMC)

---

6. Desain Hibrida Exemestane–Letrozole sebagai Kandidat Inhibitor Aromatase Steroidal-Nonsteroidal

Riset ini menggabungkan kelebihan Exemestane sebagai inhibitor aromatase steroidal dengan fragmen triazol/nitril dari Letrozole. Mahasiswa dapat merancang 6–10 kandidat dengan memasang fragmen azol atau nitril pada posisi yang tidak merusak inti steroid. Karena struktur steroid cukup besar, jumlah kandidat sebaiknya dibatasi. Tujuannya adalah mencari kandidat yang memiliki interaksi kuat pada kantong aromatase dan tetap memiliki profil ADMET yang layak.

Software yang digunakan: **Avogadro, RDKit, ORCA/PySCF, AutoDock Vina, Discovery Studio, PyMOL, SwissADME, pkCSM**.

Metode komputasi: **DFT, docking aromatase, analisis mode ikatan steroidal, analisis interaksi H-bond/hidrofobik, ADMET, drug-likeness**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; gunakan **GFN2-xTB** untuk pre-optimasi semua kandidat.

Referensi jurnal yang berkaitan: studi docking dan QSAR steroidal aromatase inhibitors menunjukkan bahwa ikatan hidrogen tertentu dengan residu aktif aromatase penting untuk optimasi afinitas, sehingga ide modifikasi steroidal-nonsteroidal dapat diuji secara komputasi. (ScienceDirect)

---

7. Desain Hibrida Fulvestrant–Tamoxifen sebagai Kandidat Selective Estrogen Receptor Degrader/Modulator Baru

Topik ini mengarah pada desain kandidat yang menggabungkan karakter **SERD** dari Fulvestrant dan karakter **SERM** dari Tamoxifen. Karena Fulvestrant besar dan sangat fleksibel, riset harus dibatasi pada fragmen farmakofor utama, bukan seluruh molekul penuh. Mahasiswa dapat membuat desain **truncated hybrid**, misalnya mempertahankan inti fenolik Fulvestrant dan rantai aminoetoksi Tamoxifen, kemudian mengevaluasi interaksi pada ERα.

Software yang digunakan: **RDKit, Avogadro, Open Babel, AutoDock Vina, PyMOL, Discovery Studio, SwissADME, ORCA/PySCF**.

Metode komputasi: **fragment-based hybrid design, docking ERα, DFT ligan, ADMET, conformer analysis sederhana, analisis fleksibilitas linker**.

Basis set yang disarankan: **GFN2-xTB** untuk screening konformer; **B3LYP-D3BJ/def2-SVP** untuk 3 kandidat terbaik.

Referensi jurnal yang berkaitan: review drug conjugates yang memasukkan ligan ER seperti Tamoxifen, 4-hydroxytamoxifen, Endoxifen, dan Fulvestrant mendukung strategi penggabungan ligan ER dengan agen antikanker lain. (MDPI)

---

8. Perancangan Hibrida Palbociclib–Ribociclib sebagai Kandidat Inhibitor CDK4/6 Baru melalui Scaffold Merging

Riset ini menggabungkan fragmen penting Palbociclib dan Ribociclib yang berperan dalam pengikatan ATP-binding pocket CDK4/6. Mahasiswa dapat membuat 8–12 analog hibrida dengan mempertahankan inti heterosiklik utama, tetapi memvariasikan gugus piperazinil, siklopentil, dan substituen amida. Target docking: **CDK4 dan CDK6**.

Software yang digunakan: **MarvinSketch, RDKit, Avogadro, AutoDock Vina, Discovery Studio, PyMOL, ORCA/PySCF, SwissADME**.

Metode komputasi: **scaffold merging, docking CDK4/CDK6, DFT, ADMET, analisis selectivity CDK4 vs CDK6, molecular fingerprints**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; untuk screening cepat dapat memakai **MMFF94** lalu DFT untuk kandidat terbaik.

Referensi jurnal yang berkaitan: Palbociclib, Ribociclib, dan Abemaciclib merupakan inhibitor CDK4/6 utama pada kanker payudara; literatur juga menekankan perbedaan potensi dan selektivitas ketiganya, sehingga desain hibrida berbasis scaffold CDK4/6 cukup relevan. (PMC)

---

9. Desain Hibrida Abemaciclib–Palbociclib untuk Meningkatkan Selektivitas CDK4 dibanding CDK6 secara In Silico

Judul ini lebih spesifik dibanding judul sebelumnya. Fokusnya bukan hanya binding score, tetapi **selektivitas CDK4/CDK6**. Mahasiswa dapat mengambil fragmen yang diduga memberi kontribusi terhadap potensi CDK4 dari Abemaciclib dan menggabungkannya dengan inti Palbociclib. Output riset adalah kandidat hibrida yang memiliki selisih energi docking lebih baik pada CDK4 dibanding CDK6.

Software yang digunakan: **RDKit, AutoDock Vina, PyRx, PyMOL, Discovery Studio, ORCA/PySCF, SwissADME, Python pandas**.

Metode komputasi: **selective docking CDK4/CDK6, DFT, ADMET, analisis interaksi hinge residue, heatmap docking score, ranking kandidat**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP** untuk kandidat akhir; screening awal cukup **MMFF94/UFF**.

Referensi jurnal yang berkaitan: literatur CDK4/6 menjelaskan perbedaan potensi Abemaciclib, Palbociclib, dan Ribociclib terhadap CDK4/CDK6; kajian prospective CDK4/6 inhibitor juga mendukung desain analog berbasis ATP-competitive kinase inhibitors. (PMC)

---

10. Desain Hibrida CDK4/6–PARP Berbasis Fragmen Palbociclib dan Olaparib sebagai Kandidat Dual-Target Kanker Payudara

Riset ini menggabungkan fragmen **CDK4/6 inhibitor** dan **PARP inhibitor**. Ide dasarnya adalah dual-target: menghambat siklus sel dan perbaikan DNA. Untuk skripsi S1, desain dibuat sederhana: 6–10 hibrida dengan linker amida/urea pendek antara fragmen Palbociclib dan Olaparib yang disederhanakan. Docking dilakukan ke **CDK6 dan PARP-1**, lalu kandidat terbaik dipilih berdasarkan kemampuan mengikat dua target sekaligus.

Software yang digunakan: **Avogadro, RDKit, AutoDock Vina, Discovery Studio, PyMOL, ORCA/PySCF, SwissADME, pkCSM**.

Metode komputasi: **dual-target docking, DFT ligan, ADMET, analisis multi-objective ranking, perbandingan dengan Palbociclib dan Olaparib sebagai kontrol**.

Basis set yang disarankan: **GFN2-xTB** untuk pre-optimasi; **B3LYP-D3BJ/def2-SVP** untuk 3 kandidat terbaik.

Referensi jurnal yang berkaitan: pendekatan dual CDK/PARP telah dikembangkan melalui desain in silico dan struktur-berbasis data; riset CDK-1/PARP-1 menunjukkan bahwa dual-target mechanism dapat menggabungkan penghambatan siklus sel dan mekanisme perbaikan DNA. (MDPI)

---

11. Desain Hibrida PARP1–BRD4 Berbasis Fragmen Olaparib untuk Kandidat Terapi Kanker Payudara Triple-Negative

Topik ini agak lebih maju, tetapi masih dapat dikerjakan S1 bila dibatasi pada desain komputasi. Mahasiswa memakai fragmen phthalazinone dari Olaparib sebagai jangkar PARP1, lalu menggabungkan fragmen sederhana yang meniru farmakofor BRD4 inhibitor. Target docking adalah **PARP1 dan BRD4**. Tujuan riset adalah memperoleh kandidat dual inhibitor yang secara prediksi stabil pada kedua binding pocket.

Software yang digunakan: **RDKit, Avogadro, Open Babel, AutoDock Vina, PyMOL, Discovery Studio, ORCA/PySCF, SwissADME, ProTox-II**.

Metode komputasi: **fragment linking, dual docking PARP1-BRD4, DFT, ADMET, ligand efficiency, drug-likeness, analisis multi-target score**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP** untuk kandidat pendek; **GFN2-xTB** untuk screening konformer.

Referensi jurnal yang berkaitan: desain dual **PARP1–BRD4 inhibitors** untuk terapi kanker payudara telah dikembangkan dengan kombinasi pendekatan data-driven dan structure-based design. (Repository ICR)

---

12. Desain Hibrida Talazoparib–Olaparib sebagai Kandidat Inhibitor PARP-1 Baru melalui Scaffold Merging

Riset ini fokus pada dua PARP inhibitor yang sama-sama relevan pada kanker payudara BRCA-mutated. Mahasiswa dapat menggabungkan fragmen phthalazinone, triazole, dan cincin fluorofenil secara terkontrol untuk membuat 8–10 analog hibrida. Analisis diarahkan pada peningkatan interaksi hidrogen dan stacking di binding pocket PARP-1.

Software yang digunakan: **MarvinSketch, RDKit, Avogadro, AutoDock Vina, Discovery Studio, PyMOL, ORCA/PySCF, SwissADME**.

Metode komputasi: **scaffold merging, docking PARP-1, DFT, ADMET, analisis residu katalitik, ligand efficiency**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; untuk kandidat terbaik bisa dilanjutkan **M06-2X/def2-SVP**.

Referensi jurnal yang berkaitan: review computational chemistry untuk pengembangan PARP1 inhibitors menjelaskan penggunaan docking, QSAR, MD, dan metode komputasi modern dalam desain inhibitor PARP1. (PMC)

---

13. Perancangan Hibrida Lapatinib–Tucatinib sebagai Kandidat Inhibitor HER2 Selektif Berbasis Quinazoline

Judul ini mengambil inti **quinazoline** dari Lapatinib dan fitur selektivitas Tucatinib terhadap HER2. Mahasiswa mendesain 8–12 analog dengan variasi substituen halogen, metoksi, dan heteroaril. Target docking: **HER2 kinase domain**. Tujuannya memperoleh kandidat yang afinitasnya minimal sebanding dengan Lapatinib dan Tucatinib, tetapi dengan ADMET yang lebih baik secara prediksi.

Software yang digunakan: **RDKit, Avogadro, Open Babel, AutoDock Vina/PyRx, PyMOL, Discovery Studio, ORCA/PySCF, SwissADME, pkCSM**.

Metode komputasi: **DFT, docking HER2, analisis ATP-binding pocket, ADMET, prediksi toksisitas, ligand efficiency**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; karena molekul cukup besar, lakukan pre-optimasi dengan **MMFF94 atau GFN2-xTB**.

Referensi jurnal yang berkaitan: kajian HER2 kinase-targeted breast cancer therapy menyoroti inhibitor HER2 dan desain inhibitor untuk mengatasi resistensi Lapatinib; studi lain juga

menunjukkan Lapatinib digunakan sebagai pembanding penting dalam docking HER2. (ACS Publications)

---

14. Desain Hibrida Lapatinib–Neratinib sebagai Kandidat Inhibitor HER2/EGFR Ganda melalui Molecular Docking dan DFT

Riset ini merancang kandidat dual HER2/EGFR dengan menggabungkan fitur Lapatinib dan Neratinib. Lapatinib dikenal sebagai inhibitor HER2/EGFR reversibel, sedangkan Neratinib memiliki karakter inhibitor kuat berbasis kerangka quinoline. Untuk skripsi S1, fokus cukup pada desain non-kovalen, docking ke HER2 dan EGFR, lalu ADMET.

Software yang digunakan: **Avogadro, RDKit, AutoDock Vina, Discovery Studio, PyMOL, ORCA/PySCF, SwissADME, ProTox-II**.

Metode komputasi: **dual-target docking HER2/EGFR, DFT, ADMET, analisis interaksi hinge region, perbandingan dengan Lapatinib dan Neratinib**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; jika waktu terbatas, DFT hanya untuk 3 kandidat tertinggi.

Referensi jurnal yang berkaitan: desain inhibitor dual EGFR/HER2 telah dikembangkan secara in silico dengan molecular dynamics dan binding energy calculation, menggunakan Lapatinib sebagai pembanding. (arXiv)

---

15. Desain Hibrida Lapatinib–Curcumin sebagai Kandidat Anti-HER2 Berbasis Farmakofor Heteroaril dan Polifenol

Riset ini menggabungkan farmakofor HER2 inhibitor Lapatinib dengan karakter polifenolik Curcumin. Karena Curcumin memiliki isu stabilitas dan bioavailabilitas, mahasiswa dapat mendesain fragmen Curcumin yang disederhanakan dan digabungkan dengan fragmen quinazoline/aryl dari Lapatinib. Tujuan riset adalah mencari kandidat yang tetap memiliki afinitas HER2 dan memperbaiki sifat farmakokinetik prediktif.

Software yang digunakan: **RDKit, Avogadro, AutoDock Vina, Discovery Studio, PyMOL, SwissADME, pkCSM, ORCA/PySCF**.

Metode komputasi: **desain analog hibrida, docking HER2 wild-type dan HER2 mutant bila tersedia, DFT, ADMET, prediksi toksisitas**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP**; kandidat awal dapat dioptimasi **MMFF94**.

Referensi jurnal yang berkaitan: skrining turunan Curcumin terhadap HER2 wild-type dan mutan dengan RDKit/KNIME serta docking menunjukkan pendekatan derivatisasi Curcumin relevan untuk pencarian kandidat anti-HER2. (Journal WAOCP)

16. Desain Hibrida Alpelisib–Capivasertib sebagai Kandidat Dual PI3Kα/AKT Inhibitor pada Kanker Payudara Resisten Endokrin

Judul ini mengarah pada jalur **PI3K/AKT**, salah satu jalur penting dalam resistensi kanker payudara. Mahasiswa dapat mendesain hibrida sederhana dari fragmen Alpelisib dan Capivasertib, lalu melakukan docking ke **PI3Kα dan AKT1**. Karena molekul bisa menjadi besar, desain dibatasi pada fragmen kunci: heterosiklik Alpelisib dan bagian aminopiperidin/karboksamida Capivasertib.

Software yang digunakan: **RDKit, Avogadro, AutoDock Vina, PyMOL, Discovery Studio, ORCA/PySCF, SwissADME, ProTox-II**.

Metode komputasi: **dual-target docking PI3Kα/AKT1, DFT, ADMET, multi-objective scoring, analisis interaksi kinase ATP-binding pocket**.

Basis set yang disarankan: **GFN2-xTB** untuk screening; **B3LYP-D3BJ/def2-SVP** untuk kandidat terbaik.

Referensi jurnal yang berkaitan: Capivasertib relevan untuk kanker payudara HR+/HER2− dengan perubahan jalur PI3K/AKT/PTEN, dan literatur resistensi CDK4/6–PI3K juga mendukung pentingnya jalur ini sebagai target desain terapi. (MDPI)

---

17. Perancangan Hibrida Palbociclib–Alpelisib sebagai Kandidat Dual CDK4/6–PI3Kα Inhibitor

Riset ini berangkat dari ide bahwa kanker payudara dapat melibatkan aktivasi siklus sel dan jalur PI3K. Mahasiswa mendesain 6–10 hibrida sederhana dari fragmen Palbociclib dan Alpelisib. Target docking adalah **CDK6 dan PI3Kα**. Kriteria kandidat terbaik adalah docking score baik pada kedua target, ADMET tidak terlalu buruk, dan berat molekul tidak terlalu besar.

Software yang digunakan: **RDKit, AutoDock Vina, Avogadro, Discovery Studio, PyMOL, SwissADME, pkCSM, ORCA/PySCF**.

Metode komputasi: **fragment linking, dual-target docking, DFT, ADMET, drug-likeness, analisis Pareto ranking**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP** untuk kandidat akhir; **GFN2-xTB/MMFF94** untuk screening awal.

Referensi jurnal yang berkaitan: literatur CDK4/6 inhibitor dan jalur PI3K/mTOR menunjukkan kombinasi atau dual inhibition berpotensi mengatasi resistensi pada kanker yang bergantung pada jalur siklus sel dan PI3K. (MDPI)

---

18. Desain Hibrida Doxorubicin–Tamoxifen Tereduksi sebagai Kandidat Targeting ERα dengan Aktivitas Interkalasi DNA Terprediksi

Riset ini harus dilakukan hati-hati karena Doxorubicin sangat besar dan toksik. Untuk skripsi S1, yang dirancang bukan seluruh Doxorubicin penuh, tetapi **fragmen antrasiklin tereduksi** yang digabungkan dengan fragmen Tamoxifen. Tujuannya adalah melihat apakah kandidat hibrida dapat mempertahankan interaksi ERα dan memiliki kemungkinan interaksi dengan DNA/topoisomerase secara prediksi.

Software yang digunakan: **Avogadro, RDKit, ORCA/PySCF, AutoDock Vina, PyMOL, Discovery Studio, SwissADME, ProTox-II**.

Metode komputasi: **fragment simplification, DFT, docking ERα dan topoisomerase II, ADMET, prediksi toksisitas, analisis MEP**.

Basis set yang disarankan: **GFN2-xTB** untuk pre-optimasi; **B3LYP-D3BJ/def2-SVP** untuk kandidat terpilih.

Referensi jurnal yang berkaitan: drug conjugates yang memasukkan ligan ER dan agen antikanker lain telah direview sebagai strategi untuk meningkatkan targetabilitas, sedangkan studi docking Doxorubicin dan Tamoxifen dapat menjadi pembanding mekanistik. (MDPI)

---

19. Desain Hibrida Paclitaxel–Lapatinib Berbasis Fragmen Sederhana sebagai Kandidat Dual β-Tubulin/HER2

Topik ini bertujuan merancang kandidat yang menggabungkan farmakofor antimitotik dan anti-HER2. Karena Paclitaxel sangat besar, mahasiswa harus memakai **fragmen taxane sederhana** atau farmakofor cincin aromatik/ester yang relevan, bukan seluruh struktur Paclitaxel penuh. Docking dilakukan ke **β-tubulin** dan **HER2**. Topik ini menarik, tetapi desain harus dijaga agar berat molekul tidak berlebihan.

Software yang digunakan: **RDKit, Avogadro, AutoDock Vina, PyMOL, Discovery Studio, ORCA/PySCF, SwissADME**.

Metode komputasi: **fragment-based hybrid design, dual docking β-tubulin/HER2, ADMET, DFT ligan, analisis ligand efficiency**.

Basis set yang disarankan: **B3LYP-D3BJ/def2-SVP** hanya untuk kandidat terbaik; screening awal memakai **GFN2-xTB**.

Referensi jurnal yang berkaitan: strategi molecular hybrid untuk kanker payudara dan literatur HER2-targeted therapy mendukung desain senyawa multi-target, sementara Paclitaxel merupakan agen antimitotik penting dalam terapi kanker termasuk kanker payudara. (PMC)

---

20. Desain Pustaka Virtual 100 Senyawa Hibrida dari Obat Kanker Payudara untuk Skrining Multi-Target ERα, Aromatase, HER2, CDK6, PARP1, dan PI3Kα

Judul ini cocok untuk mahasiswa yang kuat Python dan cheminformatics. Mahasiswa tidak mendalami satu hibrida saja, tetapi membuat **virtual library** dari fragmen senyawa yang ada: Tamoxifen, Letrozole, Palbociclib, Lapatinib, Olaparib, dan Alpelisib. Desain dapat dibuat otomatis dengan RDKit melalui kombinasi fragmen-linker sederhana. Kemudian dilakukan filtering Lipinski, PAINS, ADMET, docking multi-target, dan pemeringkatan kandidat.

Software yang digunakan: **RDKit, Python, Open Babel, AutoDock Vina, PyRx, PyMOL, Discovery Studio, SwissADME, pkCSM, ProTox-II, pandas, matplotlib**.

Metode komputasi: **virtual library generation, fragment recombination, ADMET filtering, PAINS filtering, multi-target docking, ligand efficiency, clustering, ranking kandidat final**.

Basis set yang disarankan: tidak semua 100 molekul perlu DFT. Gunakan **MMFF94/UFF** untuk screening awal, lalu **B3LYP-D3BJ/def2-SVP** untuk 5 kandidat terbaik.

Referensi jurnal yang berkaitan: pendekatan machine learning dan molecular docking untuk screening ligan terhadap target EGFR, HER2, ER, PR, dan NF-κB pada kanker payudara menunjukkan bahwa virtual screening multi-target dapat digunakan untuk menemukan prioritas ligan baru. (arXiv)

Rancangan kerja 6 bulan yang realistis untuk semua judul di atas

Pada **bulan 1**, mahasiswa mengumpulkan struktur ligan dari PubChem/ChEMBL atau dari SMILES pada dokumen Bapak, menentukan target protein dari PDB, dan menyusun desain senyawa hibrida 6–20 kandidat. Pada **bulan 2**, semua ligan dibuat 3D, dioptimasi MMFF94/UFF atau GFN2-xTB, lalu kandidat awal difilter berdasarkan berat molekul, logP, TPSA, HBA/HBD, dan Lipinski. Pada **bulan 3**, dilakukan docking ke target utama dan validasi redocking jika tersedia ligan native. Pada **bulan 4**, 3–5 kandidat terbaik dianalisis DFT dengan **B3LYP-D3BJ/def2-SVP**, termasuk HOMO, LUMO, gap, momen dipol, dan MEP. Pada **bulan 5**, dilakukan ADMET, toksisitas awal, dan analisis interaksi 2D/3D. Pada **bulan 6**, mahasiswa menyusun pembahasan, membandingkan kandidat dengan obat induk, dan menulis skripsi.

BAB 4

RANCANGAN PROPOSAL RISET KIMIA KOMPUTASI SENYAWA HIBRIDA

Pengantar Bab

Bab ini menyajikan contoh bagaimana ide riset diterjemahkan menjadi proposal akademik yang lengkap. Penekanan diberikan pada kesinambungan antara latar belakang, rumusan masalah, tujuan, hipotesis, batasan, metode, jenis data, strategi analisis, dan luaran penelitian.

Dalam riset kimia komputasi, proposal harus lebih rinci pada bagian teknis dibandingkan proposal umum. Metode, basis set, functional, file input, parameter docking, validasi, dan cara membaca data harus dijelaskan sejak awal agar penelitian dapat direplikasi dan dipertanggungjawabkan.

Setelah pengantar ini, pembahasan dilanjutkan dengan orientasi bab, peta kompetensi, uraian materi inti, contoh kerja, dan catatan metodologis yang diperlukan untuk menuntun pembaca dari konsep menuju praktik.

Untuk skripsi S1, disarankan judul yang paling aman adalah **nomor 1, 2, 5, 8, 10, 13, 16, dan 20**, karena ruang lingkupnya jelas, targetnya kuat, dan perangkat lunaknya dapat dijalankan tanpa fasilitas superkomputer.

Perhatikan bahwa bagian metode harus lebih rinci daripada proposal umum. Pada riset komputasi, parameter seperti functional DFT, basis set, grid, kriteria konvergensi, metode docking, ukuran grid box, validasi redocking, dan kriteria ADMET perlu dijelaskan agar penelitian dapat direplikasi.

Bab ini menunjukkan bagaimana ide riset diturunkan menjadi proposal. Proposal komputasi yang baik harus memiliki logika yang runtut: mengapa target dipilih, mengapa senyawa induk relevan, bagaimana kandidat hibrida dirancang, metode apa yang digunakan, data apa yang dikumpulkan, dan bagaimana kandidat terbaik dipilih.

Orientasi Bab

Subbab ini menguraikan Orientasi Bab. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

Judul **hibrida Letrozole–Anastrozole** karena paling sesuai untuk S1: molekul induknya relatif kecil, target biologisnya jelas yaitu **aromatase/CYP19A1**, jumlah kandidat hibrida dapat dibatasi, serta metode **DFT–docking–ADMET** dapat selesai dalam **±6 bulan** dengan laptop/komputer laboratorium standar. Dalam dokumen, Letrozole dan Anastrozole sudah tersedia beserta nama IUPAC dan SMILES sehingga dapat langsung dijadikan molekul induk untuk desain struktur 3D.

Judul

Desain In Silico Senyawa Hibrida Letrozole–Anastrozole sebagai Kandidat Inhibitor Aromatase untuk Terapi Kanker Payudara Berbasis DFT, Molecular Docking, dan Prediksi ADMET

---

1. Ringkasan Proposal

Kanker payudara hormon-reseptor positif sangat berkaitan dengan biosintesis estrogen. Salah satu target penting dalam pengendalian produksi estrogen adalah enzim aromatase atau CYP19A1. Aromatase berperan dalam tahap akhir biosintesis estrogen dengan mengubah androgen menjadi estrogen, sehingga penghambatan aromatase menjadi strategi penting dalam terapi kanker payudara yang bergantung pada estrogen. Letrozole dan Anastrozole merupakan inhibitor aromatase nonsteroidal yang banyak digunakan dalam terapi kanker payudara, terutama pada pasien pascamenopause. (Pusat Informasi Bioteknologi Nasional)

Penelitian ini bertujuan merancang sejumlah kandidat senyawa hibrida baru berbasis struktur Letrozole dan Anastrozole. Strategi desain dilakukan dengan mempertahankan farmakofor penting berupa gugus triazol, nitril, dan cincin aromatik, kemudian memodifikasi spacer serta substituen tertentu untuk memperoleh kandidat yang diprediksi memiliki afinitas lebih baik terhadap aromatase. Pendekatan hibridisasi molekul dipilih karena penggabungan dua farmakofor aktif dalam satu molekul telah menjadi strategi penting dalam desain kandidat antikanker modern, termasuk pada riset kanker payudara. (PMC)

Metode penelitian meliputi pembuatan struktur 3D kandidat hibrida, optimasi geometri dengan metode DFT, analisis sifat elektronik, molecular docking terhadap aromatase/CYP19A1, validasi docking, prediksi ADMET, prediksi toksisitas awal, serta pemeringkatan kandidat. Penelitian ini bersifat komputasi murni sehingga dapat dilakukan dalam waktu enam bulan oleh mahasiswa S1 dengan perangkat lunak terbuka atau semi-terbuka.

---

2. Latar Belakang

Kanker payudara merupakan salah satu kanker yang secara biologis heterogen. Salah satu subtipe penting adalah kanker payudara hormon-reseptor positif, yaitu kanker yang pertumbuhannya dapat dipengaruhi oleh keberadaan estrogen. Pada kondisi ini, pengurangan produksi estrogen atau penghambatan sinyal estrogen menjadi strategi terapi yang penting. Aromatase inhibitor seperti Letrozole, Anastrozole, dan Exemestane telah dikenal sebagai kelompok terapi hormonal yang digunakan pada kanker payudara, khususnya pada perempuan pascamenopause. (Pusat Informasi Bioteknologi Nasional)

Aromatase atau CYP19A1 merupakan enzim sitokrom P450 yang mengatalisis tahap akhir biosintesis estrogen. Enzim ini mengubah androgen seperti androstenedione dan testosteron menjadi estrogen seperti estrone dan estradiol melalui reaksi oksidasi dan aromatisasi. Karena estrogen dapat mendorong pertumbuhan beberapa jenis kanker payudara, maka aromatase menjadi salah satu target molekuler yang sangat penting dalam desain obat antikanker payudara. (GeneCards)

Letrozole dan Anastrozole termasuk inhibitor aromatase nonsteroidal. Keduanya memiliki karakter farmakofor yang menarik untuk riset kimia komputasi. Letrozole mengandung gugus triazol dan dua gugus benzonitril, sedangkan Anastrozole mengandung gugus triazol, gugus nitril, dan kerangka aromatik tersubstitusi. Berdasarkan dokumen molekul kanker payudara yang digunakan sebagai data awal, Letrozole memiliki SMILES N#Cc1ccc(cc1)C(c2ccc(C#N)cc2)n3ncnc3, sedangkan Anastrozole memiliki SMILES N#CC(C)(C)c1cc(Cn2cncn2)cc(c1)C(C)(C)C#N.

Molecular docking telah digunakan untuk menjelaskan mode ikatan inhibitor aromatase dari berbagai generasi terhadap struktur kristal aromatase. Pendekatan ini membantu memahami orientasi ligan, interaksi dengan residu aktif, serta peran gugus farmakofor dalam penghambatan enzim. Dengan demikian, docking merupakan metode yang relevan untuk skrining awal kandidat hibrida Letrozole–Anastrozole. (PMC)

Selain docking, pendekatan DFT diperlukan untuk memahami sifat elektronik ligan, seperti energi HOMO, LUMO, gap HOMO–LUMO, momen dipol, dan molecular electrostatic potential. Studi desain analog Letrozole berbasis quantum docking, molecular docking, molecular dynamics, dan kajian teoretis telah dilaporkan dalam riset aromatase inhibitor, sehingga kombinasi DFT dan docking memiliki dasar metodologis yang kuat untuk desain kandidat baru. (PubMed)

Berdasarkan latar belakang tersebut, penelitian ini mengusulkan desain in silico senyawa hibrida Letrozole–Anastrozole. Penelitian diarahkan untuk memperoleh kandidat molekul baru yang secara komputasi diprediksi memiliki interaksi lebih baik terhadap aromatase dibandingkan molekul induknya, serta tetap memiliki profil drug-likeness dan ADMET yang layak untuk pengembangan lebih lanjut.

---

3. Rumusan Masalah

Penelitian ini dirancang untuk menjawab beberapa pertanyaan utama.

Pertama, bagaimana rancangan struktur senyawa hibrida Letrozole–Anastrozole yang secara kimiawi masih rasional, tidak terlalu besar, dan tetap mempertahankan farmakofor penting inhibitor aromatase?

Kedua, bagaimana sifat elektronik kandidat hibrida berdasarkan hasil optimasi geometri DFT, khususnya energi HOMO, LUMO, gap HOMO–LUMO, momen dipol, dan molecular electrostatic potential?

Ketiga, bagaimana afinitas prediktif kandidat hibrida terhadap aromatase/CYP19A1 berdasarkan molecular docking?

Keempat, kandidat mana yang memiliki kombinasi terbaik antara docking score, pola interaksi residu, sifat elektronik, drug-likeness, ADMET, dan toksisitas awal?

Kelima, apakah kandidat hibrida tertentu secara komputasi lebih menjanjikan dibandingkan Letrozole dan Anastrozole sebagai ligan pembanding?

---

4. Tujuan Penelitian

Tujuan umum penelitian ini adalah merancang dan mengevaluasi kandidat senyawa hibrida Letrozole–Anastrozole sebagai calon inhibitor aromatase untuk kanker payudara menggunakan pendekatan kimia komputasi.

Tujuan khusus penelitian ini adalah sebagai berikut.

Pertama, menyusun pustaka kecil kandidat hibrida Letrozole–Anastrozole sebanyak 8–12 senyawa berdasarkan prinsip molecular hybridization.

Kedua, mengoptimasi struktur geometri Letrozole, Anastrozole, dan kandidat hibrida menggunakan metode DFT.

Ketiga, menganalisis sifat elektronik kandidat hibrida melalui parameter HOMO, LUMO, gap HOMO–LUMO, momen dipol, dan molecular electrostatic potential.

Keempat, melakukan molecular docking terhadap target aromatase/CYP19A1 untuk mengetahui afinitas dan pola interaksi molekul.

Kelima, memprediksi drug-likeness, ADMET, dan toksisitas awal kandidat hibrida.

Keenam, menentukan kandidat hibrida terbaik yang layak direkomendasikan untuk riset lanjutan, baik komputasi lanjutan maupun sintesis eksperimental.

---

5. Hipotesis Penelitian

Hipotesis utama penelitian ini adalah bahwa penggabungan farmakofor Letrozole dan Anastrozole melalui desain hibrida yang rasional dapat menghasilkan kandidat molekul baru yang secara komputasi menunjukkan afinitas lebih baik terhadap aromatase/CYP19A1 dibandingkan salah satu atau kedua molekul induknya.

Hipotesis operasionalnya adalah kandidat hibrida yang mempertahankan gugus triazol dan nitril, serta memiliki orientasi aromatik yang sesuai dengan kantong aktif aromatase, akan memberikan docking score lebih negatif, interaksi residu aktif lebih stabil, dan profil ADMET yang masih dapat diterima.

---

6. Kebaruan Penelitian

Kebaruan penelitian ini terletak pada desain pustaka kecil senyawa hibrida Letrozole–Anastrozole yang difokuskan pada kombinasi farmakofor nonsteroidal aromatase inhibitor. Penelitian sebelumnya telah banyak membahas docking inhibitor aromatase dan desain analog Letrozole, tetapi rancangan hibrida sederhana yang menggabungkan ciri struktural Letrozole dan Anastrozole masih dapat dikembangkan sebagai topik skripsi komputasi. Molecular hybridization sendiri telah diakui sebagai strategi desain molekul antikanker karena memungkinkan penggabungan dua motif aktif dalam satu struktur kandidat. (PMC)

Kebaruan praktisnya adalah menghasilkan alur kerja yang dapat direplikasi mahasiswa S1: desain struktur, optimasi DFT, docking, ADMET, dan pemeringkatan kandidat dalam satu workflow yang lengkap tetapi tetap realistis untuk enam bulan.

---

7. Batasan Penelitian

Penelitian ini merupakan penelitian **in silico** dan tidak mencakup sintesis kimia, uji kultur sel, uji enzim, uji hewan, atau uji klinis.

Jumlah kandidat hibrida dibatasi pada 8–12 senyawa agar realistis untuk skripsi S1.

Target protein utama hanya aromatase/CYP19A1. Target lain seperti ERα, HER2, CDK4/6, atau PARP1 tidak dianalisis dalam penelitian utama.

Molecular docking digunakan sebagai metode prediksi awal afinitas, bukan bukti aktivitas biologis final.

Prediksi ADMET dan toksisitas bersifat komputasional dan hanya digunakan untuk penyaringan awal.

DFT dilakukan pada ligan bebas, bukan pada kompleks protein-ligan penuh, karena komputasi kuantum seluruh kompleks protein-ligan terlalu besar untuk riset S1.

8. Manfaat Penelitian

Secara teoretis, penelitian ini memberikan contoh penerapan hibridisasi molekul, DFT, docking, dan ADMET dalam desain kandidat inhibitor aromatase.

Secara praktis, penelitian ini menghasilkan kandidat struktur baru berbasis Letrozole–Anastrozole yang dapat menjadi dasar penelitian lanjutan.

Secara pendidikan, penelitian ini melatih mahasiswa S1 dalam alur kerja standar kimia komputasi modern, mulai dari pembuatan struktur molekul, optimasi geometri, docking, analisis elektronik, sampai evaluasi ADMET.

Secara metodologis, penelitian ini dapat menjadi template skripsi kimia komputasi untuk desain obat antikanker berbasis struktur.

9. Desain Penelitian

Jenis penelitian ini adalah **penelitian kimia komputasi berbasis desain molekul dan virtual screening terbatas**.

Model penelitian yang digunakan adalah:

Desain struktur hibrida → optimasi geometri → validasi struktur → docking protein-ligan → analisis interaksi → prediksi ADMET → pemeringkatan kandidat → rekomendasi kandidat terbaik.

Objek penelitian adalah Letrozole, Anastrozole, dan kandidat hibrida Letrozole–Anastrozole.

Target molekuler adalah aromatase/CYP19A1.

Kontrol positif adalah Letrozole dan Anastrozole.

Parameter utama keberhasilan adalah docking score, kesesuaian orientasi ligan dalam binding pocket aromatase, jumlah dan kualitas interaksi residu, HOMO–LUMO gap, MEP, drug-likeness, ADMET, dan prediksi toksisitas.

10. Data dan Bahan Komputasi

Data ligan utama adalah Letrozole dan Anastrozole. Struktur awal dapat diperoleh dari SMILES yang sudah tersedia dalam dokumen molekul kanker payudara Bapak atau dari basis data PubChem.

Data protein target adalah struktur tiga dimensi aromatase/CYP19A1 dari Protein Data Bank. Struktur yang disarankan adalah struktur kristal aromatase manusia yang memiliki ligan ko-kristal atau

inhibitor referensi. Keberadaan ligan ko-kristal penting untuk validasi redocking dan penentuan grid box docking. Studi docking inhibitor aromatase sebelumnya telah menggunakan struktur kristal aromatase untuk menjelaskan binding mode inhibitor aromatase. (PMC)

Data pembanding berupa docking score dan interaksi Letrozole serta Anastrozole terhadap aromatase.

Data output berupa geometri teroptimasi, energi total, HOMO, LUMO, gap energi, momen dipol, peta MEP, docking score, residu interaksi, parameter ADMET, dan prediksi toksisitas.

---

11. Perangkat Lunak yang Digunakan

Perangkat lunak utama yang digunakan adalah sebagai berikut.

RDKit digunakan untuk membaca SMILES, membangkitkan struktur 3D, menambahkan atom hidrogen, melakukan minimisasi awal MMFF94, menghitung descriptor molekul dasar, serta membantu pembuatan pustaka kandidat hibrida.

Avogadro digunakan untuk menggambar struktur molekul, melakukan pemeriksaan visual geometri, dan melakukan minimisasi awal secara manual bila diperlukan.

Open Babel digunakan untuk konversi format molekul, misalnya SMILES, MOL, SDF, PDB, dan PDBQT.

ORCA atau PySCF digunakan untuk optimasi geometri DFT, perhitungan energi elektronik, HOMO, LUMO, dan momen dipol. ORCA lebih disarankan bila ingin memperoleh output DFT yang lengkap dan mudah dianalisis, sedangkan PySCF dapat digunakan untuk pembelajaran dan otomasi Python.

AutoDockTools digunakan untuk preparasi protein dan ligan ke format PDBQT, penambahan hidrogen polar, dan penentuan grid box.

AutoDock Vina digunakan untuk molecular docking.

PyRx dapat digunakan sebagai antarmuka praktis untuk AutoDock Vina agar workflow lebih mudah bagi mahasiswa S1.

Discovery Studio Visualizer digunakan untuk visualisasi interaksi 2D antara ligan dan residu protein.

PyMOL digunakan untuk visualisasi struktur 3D protein-ligan.

SwissADME digunakan untuk prediksi drug-likeness, Lipinski, TPSA, logP, GI absorption, dan bioavailability radar.

pkCSM atau admetSAR digunakan untuk prediksi ADMET lanjutan.

ProTox-II digunakan untuk prediksi toksisitas awal.

Python, pandas, matplotlib, dan LibreOffice Calc/Excel digunakan untuk rekapitulasi data, pemeringkatan kandidat, grafik, dan analisis statistik deskriptif.

---

12. Metode Komputasi dan Parameter

12.1 Desain Senyawa Hibrida

Kandidat hibrida dirancang berdasarkan penggabungan farmakofor Letrozole dan Anastrozole. Farmakofor yang dipertahankan adalah gugus triazol, gugus nitril, dan cincin aromatik. Variasi struktur dilakukan pada jenis linker, posisi substituen, dan jumlah gugus nitril.

Jenis hibrida yang dirancang meliputi **linked hybrid** dan **merged hybrid**. Linked hybrid dibuat dengan menghubungkan fragmen Letrozole dan Anastrozole melalui linker pendek seperti metilena, eter, amida, atau urea. Merged hybrid dibuat dengan mempertahankan satu inti aromatik bersama yang membawa gugus triazol dan nitril.

Jumlah kandidat hibrida dirancang sebanyak 8–12 senyawa agar sesuai dengan waktu penelitian enam bulan.

12.2 Optimasi Awal Struktur

Semua ligan dibangun dari SMILES atau digambar manual. Struktur 3D dibangkitkan menggunakan RDKit dengan metode embedding ETKDG. Atom hidrogen ditambahkan secara eksplisit. Struktur awal diminimisasi menggunakan force field **MMFF94** atau **UFF**.

Jika struktur terlalu fleksibel, dilakukan conformer search sederhana dengan 20–50 konformer. Konformer dengan energi MMFF94 terendah dipilih untuk tahap DFT.

12.3 Optimasi Geometri DFT

Optimasi geometri ligan dilakukan menggunakan metode:

B3LYP-D3BJ/def2-SVP

Metode B3LYP dipilih karena banyak digunakan dalam studi struktur organik dan DFT ligan obat. Koreksi dispersi D3BJ digunakan karena kandidat mengandung cincin aromatik dan interaksi dispersi dapat memengaruhi geometri. Basis set def2-SVP dipilih karena memberikan keseimbangan antara akurasi dan efisiensi komputasi untuk molekul organik berukuran sedang.

Untuk validasi terbatas pada 2–3 kandidat terbaik, dapat dilakukan single-point energy menggunakan:

B3LYP-D3BJ/def2-TZVP

atau

M06-2X/def2-SVP

Parameter DFT yang dianalisis adalah energi total, energi HOMO, energi LUMO, gap HOMO–LUMO, momen dipol, dan molecular electrostatic potential.

12.4 Preparasi Protein Aromatase

Protein aromatase diunduh dari Protein Data Bank. Protein dipersiapkan dengan menghapus molekul air yang tidak relevan, mempertahankan kofaktor penting bila diperlukan, menambahkan atom hidrogen polar, serta menentukan muatan sesuai kebutuhan docking.

Ligan ko-kristal dipisahkan dan digunakan untuk validasi redocking. Grid box ditentukan berdasarkan posisi ligan ko-kristal atau binding pocket aromatase.

12.5 Validasi Docking

Validasi docking dilakukan dengan metode redocking, yaitu ligan ko-kristal dipisahkan dari protein kemudian didocking kembali ke binding pocket asalnya. Docking dianggap memadai jika pose redocking mendekati posisi kristal dengan RMSD rendah, idealnya kurang dari 2,0 Å.

Jika ligan ko-kristal tidak tersedia atau sulit dipakai, validasi dilakukan dengan docking ulang Letrozole atau Anastrozole sebagai kontrol positif dan membandingkan pola interaksi dengan literatur docking aromatase.

12.6 Molecular Docking

Docking dilakukan menggunakan AutoDock Vina. Semua ligan hasil optimasi DFT dikonversi ke format PDBQT. Protein disiapkan dalam format PDBQT. Grid box difokuskan pada kantong aktif aromatase.

Parameter docking yang digunakan:

exhaustiveness: 8–16
num_modes: 10–20
energy_range: 3 kcal/mol

Pose terbaik dipilih berdasarkan docking score paling negatif dan kesesuaian interaksi dengan residu aktif. Namun, docking score tidak digunakan sebagai satu-satunya dasar keputusan; pola interaksi dan posisi ligan dalam binding pocket juga dianalisis.

12.7 Analisis Interaksi Protein-Ligan

Analisis interaksi dilakukan dengan Discovery Studio Visualizer dan PyMOL. Interaksi yang dianalisis meliputi ikatan hidrogen, interaksi hidrofobik, π-π stacking, π-alkyl, van der Waals, dan interaksi dengan daerah aktif aromatase.

Kandidat terbaik adalah kandidat yang memiliki docking score baik, pose stabil, interaksi residu yang masuk akal, serta tidak menunjukkan benturan sterik besar.

12.8 Prediksi Drug-Likeness dan ADMET

Prediksi drug-likeness dilakukan menggunakan SwissADME. Parameter yang dianalisis meliputi molecular weight, logP, TPSA, hydrogen bond donor, hydrogen bond acceptor, rotatable bonds, Lipinski rule of five, Veber rule, GI absorption, dan bioavailability score.

Prediksi ADMET dilakukan menggunakan pkCSM atau admetSAR. Parameter yang dianalisis meliputi absorpsi intestinal, distribusi, blood-brain barrier permeability, metabolisme CYP450, clearance, hepatotoxicity, AMES toxicity, dan skin sensitization.

Prediksi toksisitas awal dilakukan menggunakan ProTox-II untuk memperkirakan kelas toksisitas dan kemungkinan endpoint toksik tertentu.

---

13. Prosedur Penelitian

Tahap 1: Studi Literatur dan Penentuan Molekul Induk

Mahasiswa mengumpulkan literatur tentang kanker payudara hormon-reseptor positif, aromatase/CYP19A1, Letrozole, Anastrozole, inhibitor aromatase, molecular hybridization, DFT, docking, dan ADMET. Pada tahap ini juga dikumpulkan struktur SMILES Letrozole dan Anastrozole dari dokumen awal dan basis data publik.

Tahap 2: Desain Kandidat Hibrida

Mahasiswa membuat 8–12 kandidat hibrida Letrozole–Anastrozole. Setiap kandidat diberi kode, misalnya LAH-01 sampai LAH-12. Struktur digambar menggunakan Avogadro, MarvinSketch, atau RDKit. Kandidat yang terlalu besar, terlalu fleksibel, atau melanggar terlalu banyak aturan drug-likeness dieliminasi sejak awal.

Tahap 3: Pembuatan Struktur 3D dan Minimisasi Awal

Semua kandidat dikonversi menjadi struktur 3D. Atom hidrogen ditambahkan. Struktur diminimisasi menggunakan MMFF94 atau UFF. Struktur yang gagal diminimisasi diperbaiki secara manual atau diganti dengan kandidat lain.

Tahap 4: Optimasi Geometri DFT

Letrozole, Anastrozole, dan semua kandidat hibrida dioptimasi menggunakan B3LYP-D3BJ/def2-SVP. Output DFT diperiksa untuk memastikan struktur konvergen. Data HOMO, LUMO, gap HOMO–LUMO, energi total, dan momen dipol dikumpulkan.

Tahap 5: Preparasi Protein Aromatase

Struktur aromatase diunduh dari PDB. Protein dibersihkan, hidrogen polar ditambahkan, muatan disiapkan, dan ligan ko-kristal dipisahkan untuk redocking. Grid box ditentukan berdasarkan kantong aktif.

Tahap 6: Validasi Docking

Validasi redocking dilakukan. Jika nilai RMSD pose redocking memenuhi kriteria, parameter docking digunakan untuk seluruh kandidat. Jika tidak memenuhi, parameter grid box dan preparasi protein diperbaiki.

Tahap 7: Docking Ligan

Letrozole, Anastrozole, dan kandidat hibrida didocking ke aromatase. Setiap docking menghasilkan beberapa pose. Pose terbaik dipilih berdasarkan docking score, orientasi di binding pocket, dan interaksi residu.

Tahap 8: Analisis Interaksi

Pose docking divisualisasi. Interaksi 2D dan 3D dianalisis. Kandidat dibandingkan dengan Letrozole dan Anastrozole.

Tahap 9: Prediksi ADMET dan Toksisitas

Kandidat dengan docking score baik dianalisis lebih lanjut menggunakan SwissADME, pkCSM/admetSAR, dan ProTox-II. Kandidat dengan toksisitas prediktif tinggi atau pelanggaran drug-likeness berat diberi peringkat rendah.

Tahap 10: Pemeringkatan Kandidat

Pemeringkatan dilakukan dengan sistem skor gabungan. Parameter yang dipakai adalah docking score, jumlah interaksi penting, HOMO–LUMO gap, logP, TPSA, Lipinski, GI absorption, toksisitas, dan kemudahan sintesis secara teoritis.

Tahap 11: Penyusunan Kesimpulan

Mahasiswa menentukan 1–3 kandidat terbaik dan menjelaskan alasan pemilihannya berdasarkan bukti komputasi.

---

14. Instrumen Analisis Data

Instrumen analisis data dalam penelitian ini adalah sebagai berikut.

Analisis struktur dan energi elektronik menggunakan output DFT dari ORCA atau PySCF. Data yang diambil meliputi energi total, HOMO, LUMO, gap energi, momen dipol, dan distribusi muatan atau MEP.

Analisis docking menggunakan AutoDock Vina output. Parameter utama adalah docking score dalam kcal/mol, pose ligan, dan interaksi dengan residu aromatase.

Analisis visual interaksi menggunakan Discovery Studio Visualizer dan PyMOL. Parameter yang diamati adalah ikatan hidrogen, interaksi hidrofobik, π-π stacking, π-alkyl, dan van der Waals.

Analisis drug-likeness menggunakan SwissADME. Parameter utama adalah Lipinski, TPSA, logP, rotatable bonds, GI absorption, dan bioavailability score.

Analisis ADMET menggunakan pkCSM atau admetSAR. Parameter utama adalah absorpsi, distribusi, metabolisme, ekskresi, hepatotoxicity, AMES toxicity, dan CYP inhibition.

Analisis toksisitas awal menggunakan ProTox-II. Parameter utama adalah LD50 prediktif, kelas toksisitas, dan potensi endpoint toksik.

Analisis pemeringkatan menggunakan Excel, LibreOffice Calc, atau Python pandas. Kandidat diberi skor gabungan untuk menentukan molekul terbaik.

---

15. Teknik Analisis dan Kriteria Pemilihan Kandidat Terbaik

Kandidat terbaik dipilih berdasarkan beberapa kriteria.

Pertama, docking score kandidat harus lebih negatif atau minimal sebanding dengan Letrozole dan Anastrozole.

Kedua, pose docking harus berada di binding pocket aromatase dan tidak menyimpang jauh dari area aktif.

Ketiga, kandidat harus memiliki interaksi penting seperti ikatan hidrogen, interaksi hidrofobik, atau π-π stacking dengan residu aktif.

Keempat, HOMO–LUMO gap tidak boleh menunjukkan ketidakstabilan elektronik ekstrem. Gap yang terlalu kecil dapat menunjukkan reaktivitas tinggi, sedangkan gap yang terlalu besar dapat menunjukkan molekul terlalu inert. Interpretasi dilakukan secara relatif terhadap Letrozole dan Anastrozole.

Kelima, parameter ADMET harus layak. Kandidat yang melanggar banyak aturan Lipinski, memiliki TPSA terlalu tinggi, logP terlalu ekstrem, atau toksisitas prediktif tinggi tidak diprioritaskan.

Keenam, kandidat harus secara struktur masih rasional untuk kemungkinan sintesis lanjutan.

---

16. Rencana Jadwal Penelitian Enam Bulan

Bulan pertama digunakan untuk studi literatur, pengumpulan struktur ligan dan protein, serta penyusunan desain awal hibrida.

Bulan kedua digunakan untuk menggambar struktur, membuat SMILES, membangkitkan struktur 3D, melakukan minimisasi MMFF94/UFF, dan memilih kandidat awal.

Bulan ketiga digunakan untuk optimasi geometri DFT dan analisis sifat elektronik.

Bulan keempat digunakan untuk preparasi protein, validasi docking, dan molecular docking semua kandidat.

Bulan kelima digunakan untuk analisis interaksi, ADMET, toksisitas, dan pemeringkatan kandidat.

Bulan keenam digunakan untuk penyusunan skripsi, pembuatan grafik, pembahasan, revisi, dan persiapan ujian.

---

17. Luaran Penelitian

Luaran utama penelitian adalah:

satu pustaka kecil kandidat hibrida Letrozole–Anastrozole;

struktur 3D teroptimasi dari Letrozole, Anastrozole, dan kandidat hibrida;

data DFT berupa energi total, HOMO, LUMO, gap energi, momen dipol, dan MEP;

data docking terhadap aromatase/CYP19A1;

visualisasi interaksi 2D dan 3D;

data ADMET dan toksisitas awal;

peringkat kandidat terbaik;

naskah skripsi S1;

dan satu draft artikel ilmiah sederhana berbasis hasil skripsi.

---

18. Risiko Penelitian dan Strategi Mitigasi

Risiko pertama adalah struktur hibrida menjadi terlalu besar dan melanggar aturan drug-likeness. Strateginya adalah membatasi panjang linker dan memilih desain merged hybrid yang lebih kompak.

Risiko kedua adalah optimasi DFT memerlukan waktu lama. Strateginya adalah melakukan screening awal dengan MMFF94 atau GFN2-xTB, kemudian DFT hanya dilakukan pada kandidat yang lolos seleksi awal.

Risiko ketiga adalah validasi docking gagal. Strateginya adalah memperbaiki grid box, memeriksa preparasi protein, mempertahankan kofaktor penting, dan membandingkan dengan protokol docking aromatase dalam literatur.

Risiko keempat adalah hasil docking kandidat tidak lebih baik dari molekul induk. Strateginya adalah tetap menjadikan hasil tersebut sebagai temuan ilmiah, karena riset komputasi tidak harus selalu menghasilkan kandidat yang lebih baik; yang penting adalah analisis struktur-aktivitasnya jelas.

---

19. Etika dan Keselamatan Penelitian

Penelitian ini tidak menggunakan manusia, hewan, jaringan biologis, atau data pasien, sehingga tidak memerlukan izin etik biomedis. Namun, penelitian tetap harus mengikuti etika akademik, yaitu mencantumkan sumber data molekul, sumber protein, perangkat lunak, dan referensi ilmiah dengan benar.

Hasil penelitian tidak boleh diklaim sebagai obat kanker baru yang terbukti efektif. Istilah yang tepat adalah “kandidat senyawa hibrida potensial berdasarkan prediksi komputasi” atau “kandidat awal untuk studi lanjutan”.

---

20. Struktur Bab Skripsi yang Disarankan

Bab I Pendahuluan berisi latar belakang, rumusan masalah, tujuan, manfaat, batasan penelitian, dan kebaruan.

Bab II Tinjauan Pustaka berisi kanker payudara hormon-reseptor positif, aromatase/CYP19A1, Letrozole, Anastrozole, molecular hybridization, DFT, molecular docking, dan ADMET.

Bab III Metode Penelitian berisi desain penelitian, alat dan bahan komputasi, software, prosedur desain ligan, DFT, docking, ADMET, dan teknik analisis data.

Bab IV Hasil dan Pembahasan berisi desain kandidat, hasil optimasi geometri, hasil DFT, docking score, interaksi residu, ADMET, toksisitas, dan pemeringkatan kandidat.

Bab V Kesimpulan dan Saran berisi kandidat terbaik, kesimpulan utama, keterbatasan, dan rekomendasi penelitian lanjutan.

---

21. Contoh Kandidat Hibrida yang Dirancang

Kandidat dapat diberi kode sebagai berikut.

LAH-01: hibrida Letrozole–Anastrozole dengan linker metilena.

LAH-02: hibrida dengan linker eter.

LAH-03: hibrida dengan linker amida.

LAH-04: hibrida dengan linker urea.

LAH-05: hibrida dengan satu gugus triazol dan dua gugus nitril.

LAH-06: hibrida dengan dua cincin aromatik dan satu substituen fluoro.

LAH-07: hibrida dengan satu gugus metoksi untuk mengubah polaritas.

LAH-08: hibrida dengan substituen kloro untuk menguji efek halogen.

LAH-09: merged hybrid dengan inti aromatik bersama.

LAH-10: linked hybrid kompak dengan rotatable bond rendah.

Daftar tersebut masih berupa rancangan awal dan dapat disesuaikan setelah analisis drug-likeness awal.

---

22. Format Data Hasil yang Dikumpulkan

Untuk setiap ligan, data yang dikumpulkan meliputi:

kode ligan;

nama ligan;

SMILES;

formula molekul;

berat molekul;

logP;

TPSA;

jumlah donor ikatan hidrogen;

jumlah akseptor ikatan hidrogen;

jumlah rotatable bonds;

energi total DFT;

energi HOMO;

energi LUMO;

gap HOMO–LUMO;

momen dipol;

docking score;

residu interaksi utama;

jumlah ikatan hidrogen;

jenis interaksi hidrofobik;

status Lipinski;

prediksi absorpsi gastrointestinal;

prediksi toksisitas;

dan skor akhir kandidat.

---

23. Daftar Pustaka Inti

Ahmad, I., et al. Review tentang signifikansi molecular hybrids untuk kanker payudara, termasuk linked, merged, dan fused hybrids sebagai strategi pengembangan kandidat anti-breast cancer. (PMC)

Edris, A., et al. Desain analog Letrozole untuk target aromatase pada kanker payudara menggunakan molecular docking, molecular dynamics, dan theoretical studies. (PubMed)

Suvannang, N., et al. Molecular docking inhibitor aromatase generasi berbeda terhadap struktur kristal aromatase untuk menjelaskan binding mode inhibitor. (PMC)

Peters, A., et al. Penjelasan klinis aromatase inhibitors, termasuk Anastrozole, dalam manajemen dan terapi kanker payudara. (Pusat Informasi Bioteknologi Nasional)

GeneCards. Deskripsi CYP19A1 sebagai gen yang mengkode aromatase, enzim sitokrom P450 yang mengatalisis tahap akhir biosintesis estrogen. (GeneCards)

Breast Cancer Now. Ringkasan penggunaan aromatase inhibitors, termasuk Letrozole, Anastrozole, dan Exemestane, sebagai terapi hormonal kanker payudara. (Breast Cancer Now)

Dokumen “Molekul Kanker Payudara”. Daftar molekul kecil kanker payudara beserta nama trivial, IUPAC, dan SMILES, termasuk Letrozole dan Anastrozole.

24. Kesimpulan Rancangan Proposal

Proposal ini layak untuk skripsi S1 karena ruang lingkupnya jelas, targetnya spesifik, molekul induknya kecil, software yang digunakan tersedia luas, dan metode yang dipakai merupakan kombinasi standar dalam riset kimia komputasi modern. Desain hibrida Letrozole–Anastrozole dapat menghasilkan kandidat struktur baru yang secara komputasi dievaluasi melalui DFT, docking, ADMET, dan toksisitas awal.

Penelitian ini tidak bertujuan langsung membuktikan aktivitas klinis, tetapi menghasilkan kandidat molekul prioritas untuk riset lanjutan. Dengan batasan jumlah kandidat 8–12 molekul dan jadwal enam bulan, topik ini sangat realistis untuk mahasiswa S1 kimia komputasi.

Judul ini paling direkomendasikan untuk S1 karena **tidak terlalu berat secara komputasi**, berbeda dari hibrida Paclitaxel, Fulvestrant, Doxorubicin, atau Palbociclib yang ukurannya jauh lebih besar dan lebih berisiko memakan waktu.

File Struktur Instruksi Python lah01-letrozole-anastrozole-hybrid.py

Menyusun file Python turunan dari **template paracetamol.py** dan mengikuti struktur instruksi pada **prompt-genmolekul.txt** untuk kandidat riset proposal: **LAH-01 Letrozole–Anastrozole Hybrid**. File ini mempertahankan pola PySCF, argumen terminal, output otomatis, log running, ekspor XYZ/MOL/SDF/PDB/CML/JSON, serta analisis struktur seperti template asal.

Nama file:

lah01-letrozole-anastrozole-hybrid.py

Molekul default dalam file:

LAH-01 Letrozole-Anastrozole Hybrid

Formula: C24H22N6

SMILES:

CC(C)(C#N)c1cc(C(n2ncnc2)c2ccc(C#N)cc2)cc(C(C)(C)C#N)c1

Struktur XYZ 3D sudah dimasukkan langsung ke dalam file dengan **52 atom**, sesuai formula **C24H22N6**. File juga sudah saya cek sintaksnya dengan py_compile dan lolos. PySCF tidak saya jalankan di sandbox ini karena modul PySCF tidak tersedia di lingkungan eksekusi, tetapi file disusun untuk langsung dijalankan di lingkungan Python/PySCF Bapak.

Download file Python LAH-01

Cara menjalankan:

python lah01-letrozole-anastrozole-hybrid.py

Jika D3BJ belum tersedia:

python lah01-letrozole-anastrozole-hybrid.py --method B3LYP --basis def2-SVP

Untuk optimasi geometri:

python lah01-letrozole-anastrozole-hybrid.py --job optimize --method B3LYP --basis def2-SVP

Untuk riset proposal, file ini menghasilkan struktur teroptimasi dan file **SDF/PDB/XYZ** yang dapat dipakai lanjut untuk **docking aromatase/CYP19A1**, visualisasi, dan ADMET.

Struktur File lah01-letrozole-anastrozole-hybrid.py

```
#!/usr/bin/env python3
# author: Kasmui
# Telp.: 0818294312
# ============================================================
#!/usr/bin/env python3
# author: Kasmui
# Telp.: 0818294312
# ============================================================
# FILE RISET KIMIA KOMPUTASI PySCF UNTUK HIBRIDA LETROZOLE-ANASTROZOLE
# ============================================================
#
# Molekul default : LAH-01 Letrozole-Anastrozole Hybrid
# Rumus molekul   : C24H22N6
# Program         : PySCF
#
# Metode default yang dipilih:
#   B3LYP-D3BJ / def2-SVP
#
# Alasan ringkas:
#   - LAH-01 Letrozole-Anastrozole Hybrid adalah molekul organik
netral tertutup-shell.
#   - B3LYP umum dipakai untuk molekul organik.
#   - Koreksi dispersi D3BJ membantu interaksi non-kovalen dan
konformasi.
#   - def2-SVP cukup seimbang untuk latihan/skrining awal tanpa
terlalu berat.
#
# Catatan penting:
#   - Untuk memakai koreksi dispersi D3BJ, PySCF memerlukan dukungan
#     pyscf-dispersion/simple-dftd3 pada beberapa instalasi.
#   - Jika belum tersedia, jalankan metode B3LYP biasa:
#       python lah01-letrozole-anastrozole-hybrid.py --method B3LYP --
basis def2-SVP
#
# Contoh pemakaian:
#   python lah01-letrozole-anastrozole-hybrid.py
#   python lah01-letrozole-anastrozole-hybrid.py --list
#   python lah01-letrozole-anastrozole-hybrid.py --method B3LYP --
basis def2-SVP
#   python lah01-letrozole-anastrozole-hybrid.py --method PBE0 --basis
def2-TZVP
#   python lah01-letrozole-anastrozole-hybrid.py --method CUSTOM --xc
"m06-2x" --basis def2-TZVP
#   python lah01-letrozole-anastrozole-hybrid.py --xyz-file
molekul.xyz --name "Molekul Baru" --formula "C2H6O"
#   python lah01-letrozole-anastrozole-hybrid.py --coord-file
koordinat.txt --method B3LYP --basis "6-31G**"
```

```
#   python lah01-letrozole-anastrozole-hybrid.py --job optimize --
method B3LYP --basis def2-SVP
#
# ============================================================

import argparse
import json
import math
import sys
from datetime import datetime
from html import escape
from pathlib import Path

from pyscf import dft, gto, scf


# ============================================================
# 1. DAFTAR METODE KOMPUTASI
# ============================================================
# family:
#   HF  = Hartree-Fock
#   DFT = Kohn-Sham DFT
# xc:
#   String functional yang dibaca oleh PySCF.
# disp:
#   None       = tanpa koreksi dispersi eksplisit.
#   "d3bj"    = koreksi dispersi D3 dengan Becke-Johnson damping.
#
# CUSTOM:
#   Dipakai bersama argumen --xc.

AVAILABLE_METHODS = {
    "HF": {
        "family": "HF",
        "xc": None,
        "disp": None,
        "description": "Hartree-Fock; RHF untuk spin 0, UHF untuk spin
bukan 0.",
    },
    "LDA": {
        "family": "DFT",
        "xc": "lda,vwn",
        "disp": None,
        "description": "DFT LDA sederhana.",
    },
    "PBE": {
        "family": "DFT",
        "xc": "pbe,pbe",
        "disp": None,
        "description": "DFT GGA PBE.",
    },
    "BLYP": {
        "family": "DFT",
        "xc": "b88,lyp",
        "disp": None,
        "description": "DFT GGA BLYP.",
    },
```

```
    "B3LYP": {
        "family": "DFT",
        "xc": "b3lyp",
        "disp": None,
        "description": "DFT hybrid B3LYP, umum untuk molekul
organik.",
    },
    "B3LYP-D3BJ": {
        "family": "DFT",
        "xc": "b3lyp",
        "disp": "d3bj",
        "description": "B3LYP dengan koreksi dispersi D3BJ; default
untuk kandidat hibrida LAH-01.",
    },
    "PBE0": {
        "family": "DFT",
        "xc": "pbe0",
        "disp": None,
        "description": "DFT hybrid PBE0.",
    },
    "PBE0-D3BJ": {
        "family": "DFT",
        "xc": "pbe0",
        "disp": "d3bj",
        "description": "PBE0 dengan koreksi dispersi D3BJ.",
    },
    "TPSS": {
        "family": "DFT",
        "xc": "tpss",
        "disp": None,
        "description": "Meta-GGA TPSS.",
    },
    "M06": {
        "family": "DFT",
        "xc": "m06",
        "disp": None,
        "description": "Meta-hybrid Minnesota M06 jika didukung
instalasi PySCF/libxc.",
    },
    "CAM-B3LYP": {
        "family": "DFT",
        "xc": "cam-b3lyp",
        "disp": None,
        "description": "Range-separated hybrid CAM-B3LYP.",
    },
    "WB97X-D": {
        "family": "DFT",
        "xc": "wb97x-d",
        "disp": None,
        "description": "Range-separated functional dengan dispersi
bawaan jika didukung.",
    },
    "CUSTOM": {
        "family": "DFT",
        "xc": None,
        "disp": None,
        "description": "Functional DFT custom dari argumen --xc.",
```

```
    },
}


# ============================================================
# 2. DAFTAR BASIS SET
# ============================================================

AVAILABLE_BASIS_SETS = {
    "STO-3G": "sto-3g",
    "3-21G": "3-21g",
    "6-31G": "6-31g",
    "6-31G*": "6-31g*",
    "6-31G**": "6-31g**",
    "6-31+G*": "6-31+g*",
    "6-31+G**": "6-31+g**",
    "def2-SVP": "def2-svp",
    "def2-TZVP": "def2-tzvp",
    "def2-QZVP": "def2-qzvp",
    "cc-pVDZ": "cc-pvdz",
    "cc-pVTZ": "cc-pvtz",
    "aug-cc-pVDZ": "aug-cc-pvdz",
    "CUSTOM": None,
}


# ============================================================
# 3. KONFIGURASI DEFAULT
# ============================================================

MOLECULE_NAME_DEFAULT = "LAH-01 Letrozole-Anastrozole Hybrid"
FORMULA_DEFAULT = "C24H22N6"

METHOD_DEFAULT = "B3LYP-D3BJ"
BASIS_DEFAULT = "def2-SVP"

CHARGE_DEFAULT = 0
SPIN_DEFAULT = 0

DFT_GRID_LEVEL_DEFAULT = 3
CONV_TOL_DEFAULT = 1e-9
MAX_CYCLE_DEFAULT = 150
VERBOSE_DEFAULT = 4

USE_DENSITY_FITTING_DEFAULT = False
JOB_DEFAULT = "singlepoint"

OUTPUT_ROOT_DEFAULT = "hasil_pyscf"
BOND_SCALE_DEFAULT = 1.25
BOND_TOLERANCE_DEFAULT = 0.15

AUTHOR_DEFAULT = "Kasmui"
AUTHOR_PHONE_DEFAULT = "0818294312"
SMILES_DEFAULT =
"CC(C)(C#N)c1cc(C(n2ncnc2)c2ccc(C#N)cc2)cc(C(C)(C)C#N)c1"
IUPAC_NAME_DEFAULT = "2,2'-[5-[(4-cyanophenyl)(1H-1,2,4-triazol-1-
yl)methyl]-1,3-phenylene]bis(2-methylpropanenitrile)"
```

```
RESEARCH_NOTE_DEFAULT = (
    "Kandidat hibrida in silico Letrozole-Anastrozole untuk riset
inhibitor aromatase/CYP19A1. "
    "Struktur ini adalah kandidat awal komputasi, bukan obat
tervalidasi klinis."
)


# ============================================================
# 4. KOORDINAT DEFAULT LAH-01 LETROZOLE-ANASTROZOLE HYBRID
# ============================================================
# Format XYZ standar juga diterima:
# baris 1 = jumlah atom, baris 2 = judul/komentar.
# Program otomatis membuang dua baris tersebut saat membaca koordinat.

DEFAULT_MOLECULE_XYZ = """
52
LAH-01 Letrozole-Anastrozole Hybrid | author: Kasmui | Telp.:
0818294312 | generated from RDKit MMFF94 conformer
C     -1.16285703     3.56822713    -0.22330159
C     -2.10886641     2.42635441     0.21010982
C     -2.42522992     2.61860541     1.70817752
C     -3.36730827     2.58805761    -0.57001762
N     -4.36170938     2.70832474    -1.15811404
C     -1.52061565     1.01310072     0.01359150
C     -0.16177500     0.79711191    -0.26307620
C      0.36083566    -0.49433572    -0.44446414
C      1.84415274    -0.72427333    -0.73431941
N      2.21687154    -0.34619052    -2.10776252
N      1.88420491     0.84024878    -2.67261278
C      2.41256212     0.71166107    -3.89214986
N      3.07029776    -0.46129526    -4.12225021
C      2.93529928    -1.10493537    -2.98666047
C      2.77829088    -0.15791737     0.33889448
C      2.94139876    -0.87761295     1.53890093
C      3.76447128    -0.39683201     2.56186082
C      4.43607054     0.81766175     2.40875175
C      5.27804309     1.31422662     3.45733495
N      5.95802625     1.71648576     4.30739764
C      4.27611227     1.54895333     1.22977636
C      3.45151123     1.06822825     0.20670028
C     -0.50965902    -1.59189117    -0.34206007
C     -1.87658900    -1.43343693    -0.03884708
C     -2.78921915    -2.67347886     0.06518197
C     -2.99157426    -3.25446587    -1.34972668
C     -4.18388871    -2.38246002     0.66108956
C     -2.15751444    -3.71224190     0.92580880
N     -1.64482863    -4.52252547     1.58135224
C     -2.35484059    -0.12214995     0.13346884
H     -1.64830649     4.54708731    -0.11998457
H     -0.86569105     3.46746507    -1.27451724
H     -0.25269457     3.59834878     0.38676234
H     -2.83771040     3.61604361     1.90373693
H     -3.16374377     1.89237691     2.06909984
H     -1.52393428     2.50343340     2.32215728
H      0.52156415     1.63862681    -0.35861219
H      2.01387757    -1.81191010    -0.72187977
```

```
H       2.31970608     1.49530682    -4.63352787
H       3.32315327    -2.08732752    -2.75533808
H       2.41908504    -1.82191848     1.69219282
H       3.87167098    -0.97532108     3.47823944
H       4.78809360     2.50100836     1.09705711
H       3.34385306     1.67665471    -0.69033051
H      -0.10930452    -2.59494931    -0.49412927
H      -3.64380714    -4.13617990    -1.32896814
H      -2.04473039    -3.56995205    -1.80464799
H      -3.44832555    -2.51645779    -2.01997791
H      -4.76977016    -3.30415567     0.76927345
H      -4.76681698    -1.70711806     0.02427887
H      -4.10957946    -1.93157114     1.65839348
H      -3.40826183     0.03930455     0.35768718
"""


# ============================================================
# 5. DATA ATOM DAN JARI-JARI KOVALEN
# ============================================================

COVALENT_RADII_ANGSTROM = {
    "H": 0.31,
    "He": 0.28,
    "Li": 1.28,
    "Be": 0.96,
    "B": 0.84,
    "C": 0.76,
    "N": 0.71,
    "O": 0.66,
    "F": 0.57,
    "Ne": 0.58,
    "Na": 1.66,
    "Mg": 1.41,
    "Al": 1.21,
    "Si": 1.11,
    "P": 1.07,
    "S": 1.05,
    "Cl": 1.02,
    "Ar": 1.06,
    "K": 2.03,
    "Ca": 1.76,
    "Fe": 1.24,
    "Cu": 1.32,
    "Zn": 1.22,
    "Br": 1.20,
    "I": 1.39,
}

ATOMIC_NUMBERS = {
    "H": 1,
    "He": 2,
    "Li": 3,
    "Be": 4,
    "B": 5,
    "C": 6,
    "N": 7,
```

```
    "O": 8,
    "F": 9,
    "Ne": 10,
    "Na": 11,
    "Mg": 12,
    "Al": 13,
    "Si": 14,
    "P": 15,
    "S": 16,
    "Cl": 17,
    "Ar": 18,
    "K": 19,
    "Ca": 20,
    "Fe": 26,
    "Cu": 29,
    "Zn": 30,
    "Br": 35,
    "I": 53,
}


# ============================================================
# 6. FUNGSI BANTUAN UMUM
# ============================================================

def clean_element_symbol(symbol):
    letters = "".join(ch for ch in str(symbol).strip() if
ch.isalpha())
    if not letters:
        return ""
    if len(letters) == 1:
        return letters.upper()
    return letters[0].upper() + letters[1:].lower()


def safe_filename(text):
    safe_chars = []
    for char in str(text):
        if char.isalnum() or char in ["-", "_"]:
            safe_chars.append(char)
        elif char in ["*", "+"]:
            safe_chars.append("x")
        else:
            safe_chars.append("_")
    value = "".join(safe_chars).strip("_")
    while "__" in value:
        value = value.replace("__", "_")
    return value or "molecule"


def normalize_choice(user_value, allowed_dict, label):
    if user_value is None:
        return None
    user_compact = str(user_value).strip().lower().replace(" ", "")
    for key in allowed_dict:
        key_compact = key.strip().lower().replace(" ", "")
        if user_compact == key_compact:
```

```
            return key
    print(f"\nERROR: {label} '{user_value}' tidak tersedia.")
    print(f"Pilihan {label} yang tersedia:")
    for key in allowed_dict:
        print(f"  - {key}")
    sys.exit(1)


def validate_positive_float(value, label):
    if value <= 0:
        print(f"ERROR: {label} harus bernilai positif.")
        sys.exit(1)


def validate_nonnegative_int(value, label):
    if value < 0:
        print(f"ERROR: {label} harus integer >= 0.")
        sys.exit(1)


def validate_positive_int(value, label):
    if value <= 0:
        print(f"ERROR: {label} harus integer positif.")
        sys.exit(1)


class TeeOutput:
    """
    Menyalin semua teks yang ditulis ke terminal sekaligus ke file
log.

    Tujuan:
    - Teks proses running tetap tampil normal di layar.
    - Teks yang sama otomatis tersimpan ke file .log.
    - stdout dan stderr ikut ditangkap, termasuk pesan error PySCF.
    """

    def __init__(self, terminal_stream, log_stream):
        self.terminal_stream = terminal_stream
        self.log_stream = log_stream

    def write(self, message):
        self.terminal_stream.write(message)
        self.log_stream.write(message)
        self.flush()

    def flush(self):
        self.terminal_stream.flush()
        self.log_stream.flush()

    def isatty(self):
        return getattr(self.terminal_stream, "isatty", lambda:
False)()

    def fileno(self):
        return self.terminal_stream.fileno()
```

```
def write_running_log_header(log_path, args, method_name, basis_name,
pyscf_basis):
    print(identity_header_text())

print("\n============================================================"
)
    print("LOG RUNNING OTOMATIS AKTIF")

print("============================================================")
    print(f"File log running  : {log_path}")
    print(f"Waktu mulai       : {datetime.now().strftime('%Y-%m-%d
%H:%M:%S')}")
    print(f"Nama molekul      : {args.name}")
    print(f"Rumus molekul     : {args.formula}")
    print(f"Metode            : {method_name}")
    print(f"Basis set         : {basis_name} ({pyscf_basis})")
    print(f"Jenis perhitungan : {args.job}")
    print("Catatan           : Semua teks terminal stdout/stderr
disalin ke file ini.")

print("============================================================")


def write_running_log_footer(log_path, status="SELESAI"):

print("\n============================================================"
)
    print("LOG RUNNING OTOMATIS DITUTUP")

print("============================================================")
    print(f"Status akhir      : {status}")
    print(f"Waktu selesai     : {datetime.now().strftime('%Y-%m-%d
%H:%M:%S')}")
    print(f"File log running  : {log_path}")

print("============================================================")


# ============================================================
# 7. PEMBACAAN DAN VALIDASI KOORDINAT
# ============================================================

def parse_xyz_like_text(text):
    """
    Membaca koordinat dari:
    1. XYZ standar:
       jumlah_atom
       komentar
       Atom x y z
    2. Blok koordinat langsung:
       Atom x y z
    3. File teks dengan komentar diawali #.
    """

    if not text or not str(text).strip():
        raise ValueError("Blok koordinat kosong.")
```

```
    raw_lines = []
    for line in str(text).splitlines():
        stripped = line.strip()
        if not stripped:
            continue
        if stripped.startswith("#"):
            continue
        raw_lines.append(stripped)

    if not raw_lines:
        raise ValueError("Tidak ada baris koordinat yang dapat
dibaca.")

    expected_count = None
    coordinate_lines = raw_lines

    first_parts = raw_lines[0].split()
    if len(first_parts) == 1:
        try:
            expected_count = int(first_parts[0])
            coordinate_lines = raw_lines[2:2 + expected_count]
        except ValueError:
            expected_count = None

    atoms = []
    for number, line in enumerate(coordinate_lines, start=1):
        parts = line.split()
        if len(parts) < 4:
            raise ValueError(
                f"Format koordinat salah pada baris koordinat ke-
{number}: '{line}'. "
                "Format benar: SimbolAtom X Y Z"
            )

        symbol = clean_element_symbol(parts[0])
        if symbol not in ATOMIC_NUMBERS:
            raise ValueError(
                f"Simbol atom '{parts[0]}' pada baris ke-{number}
tidak dikenali."
            )

        try:
            x = float(parts[1])
            y = float(parts[2])
            z = float(parts[3])
        except ValueError as exc:
            raise ValueError(
                f"Koordinat X/Y/Z harus numerik pada baris ke-
{number}: '{line}'."
            ) from exc

        atoms.append({
            "symbol": symbol,
            "x": x,
            "y": y,
            "z": z,
```

```
        })

    if expected_count is not None and len(atoms) != expected_count:
        raise ValueError(
            f"Jumlah atom di header XYZ = {expected_count}, tetapi
yang terbaca = {len(atoms)}."
        )

    if not atoms:
        raise ValueError("Tidak ada atom yang berhasil dibaca.")

    return atoms


def read_text_file(path):
    file_path = Path(path)
    if not file_path.exists():
        print(f"ERROR: File koordinat tidak ditemukan: {file_path}")
        sys.exit(1)
    return file_path.read_text(encoding="utf-8")


def format_atom_block(atoms):
    return "\n".join(
        f"{atom['symbol']:2s}  {atom['x']:14.8f}  {atom['y']:14.8f}
{atom['z']:14.8f}"
        for atom in atoms
    )


def get_coordinate_atoms(args):
    if args.xyz_file:
        source_text = read_text_file(args.xyz_file)
        source_label = f"file XYZ: {args.xyz_file}"
    elif args.coord_file:
        source_text = read_text_file(args.coord_file)
        source_label = f"file koordinat: {args.coord_file}"
    else:
        source_text = DEFAULT_MOLECULE_XYZ
        source_label = "koordinat default di dalam kode"

    try:
        atoms = parse_xyz_like_text(source_text)
    except ValueError as exc:
        print("\nERROR: Koordinat molekul tidak valid.")
        print(str(exc))
        print("\nContoh format benar:")
        print("C   0.000000   0.000000   0.000000")
        print("H   0.000000   0.000000   1.089000")
        sys.exit(1)

    return atoms, source_label


# ============================================================
# 8. RESOLUSI METODE DAN BASIS SET
# ============================================================
```

```
def resolve_method(args):
    method_name = normalize_choice(args.method, AVAILABLE_METHODS,
"metode")
    info = dict(AVAILABLE_METHODS[method_name])

    if method_name == "CUSTOM":
        if not args.xc or not args.xc.strip():
            print("ERROR: --method CUSTOM wajib disertai --xc, contoh:
--xc 'm06-2x'")
            sys.exit(1)
        info["xc"] = args.xc.strip()
        info["description"] = f"DFT custom dengan functional PySCF:
{args.xc.strip()}"

    if args.disp:
        info["disp"] = args.disp.strip().lower()

    return method_name, info


def resolve_basis(args):
    basis_name = normalize_choice(args.basis, AVAILABLE_BASIS_SETS,
"basis set")

    if basis_name == "CUSTOM":
        if not args.basis_custom or not args.basis_custom.strip():
            print("ERROR: --basis CUSTOM wajib disertai --basis-
custom, contoh: --basis-custom 'def2-tzvpp'")
            sys.exit(1)
        pyscf_basis = args.basis_custom.strip()
    else:
        pyscf_basis = AVAILABLE_BASIS_SETS[basis_name]

    return basis_name, pyscf_basis


# ============================================================
# 9. MEMBANGUN OBJEK MOLEKUL DAN KALKULATOR
# ============================================================

def build_molecule(atom_block, pyscf_basis, charge, spin, verbose):
    mol = gto.M(
        atom=atom_block,
        basis=pyscf_basis,
        unit="Angstrom",
        charge=charge,
        spin=spin,
        verbose=verbose,
    )
    return mol


def build_calculator(mol, method_name, method_info, grid_level,
conv_tol, max_cycle, use_density_fitting):
    family = method_info["family"]
```

```
    if family == "HF":
        mf = scf.RHF(mol) if mol.spin == 0 else scf.UHF(mol)
    elif family == "DFT":
        mf = dft.RKS(mol) if mol.spin == 0 else dft.UKS(mol)
        mf.xc = method_info["xc"]
        mf.grids.level = grid_level
    else:
        raise ValueError(f"Family metode tidak dikenal: {family}")

    if method_info.get("disp"):
        # PySCF modern mendukung atribut mf.disp untuk D3/D4 jika
ekstensi tersedia.
        mf.disp = method_info["disp"]

    if use_density_fitting:
        mf = mf.density_fit()

    mf.conv_tol = conv_tol
    mf.max_cycle = max_cycle
    mf.chkfile =
safe_filename(f"{MOLECULE_NAME}_{method_name}_{BASIS_LABEL}.chk")

    return mf


# ============================================================
# 10. OPTIMASI GEOMETRI OPSIONAL
# ============================================================

def optimize_geometry(mf):
    try:
        from pyscf.geomopt.geometric_solver import optimize
    except ImportError:
        print("\nERROR: Optimasi geometri memerlukan paket
geomeTRIC.")
        print("Instal terlebih dahulu:")
        print("  pip install geometric")
        sys.exit(1)


print("\n============================================================"
)
    print("OPTIMASI GEOMETRI DIMULAI")

print("============================================================")
    mol_opt = optimize(mf)

print("\n============================================================"
)
    print("OPTIMASI GEOMETRI SELESAI")

print("============================================================")
    return mol_opt


# ============================================================
# 11. INFORMASI DAN PELAPORAN DASAR
```

```
# ============================================================

def print_available_options():

print("\n============================================================"
)
    print("DAFTAR METODE KOMPUTASI")

print("============================================================")
    for method_name, info in AVAILABLE_METHODS.items():
        xc_text = info["xc"] if info["xc"] else "-"
        disp_text = info["disp"] if info["disp"] else "-"
        print(f"{method_name:12s} | family={info['family']:3s} |
xc={xc_text:12s} | disp={disp_text:5s} | {info['description']}")


print("\n============================================================"
)
    print("DAFTAR BASIS SET")

print("============================================================")
    for basis_name, pyscf_name in AVAILABLE_BASIS_SETS.items():
        print(f"{basis_name:12s} : {pyscf_name if pyscf_name else 'isi
melalui --basis-custom'}")


print("\n============================================================"
)
    print("CONTOH PERINTAH")

print("============================================================")
    print("python lah01-letrozole-anastrozole-hybrid.py")
    print("python lah01-letrozole-anastrozole-hybrid.py --method B3LYP
--basis def2-SVP")
    print("python lah01-letrozole-anastrozole-hybrid.py --xyz-file
molekul.xyz --name 'Molekul Baru' --method PBE0 --basis def2-TZVP")
    print("python lah01-letrozole-anastrozole-hybrid.py --method
CUSTOM --xc 'm06-2x' --basis CUSTOM --basis-custom 'def2-tzvpp'")
    print("python lah01-letrozole-anastrozole-hybrid.py --job optimize
--method B3LYP --basis def2-SVP")


print("\n============================================================"
)
    print("CONTOH FORMAT KOORDINAT")

print("============================================================")
    print("3")
    print("Water")
    print("O   0.000000   0.000000   0.000000")
    print("H   0.000000  -0.757000   0.587000")
    print("H   0.000000   0.757000   0.587000")
    print()


def print_molecule_info(mol, method_name, method_info, basis_name,
pyscf_basis, charge, spin, job, coordinate_source):
```

```
print("\n============================================================"
)
    print("INFORMASI SISTEM")

print("============================================================")
    print(f"Nama molekul        : {MOLECULE_NAME}")
    print(f"Rumus molekul       : {MOLECULE_FORMULA}")
    print(f"Sumber koordinat    : {coordinate_source}")
    print(f"Jumlah atom         : {mol.natm}")
    print(f"Jumlah elektron     : {mol.nelectron}")
    print(f"Jumlah fungsi basis : {mol.nao_nr()}")
    print(f"Muatan total        : {charge}")
    print(f"Spin PySCF          : {spin}")
    print(f"Metode komputasi    : {method_name}")
    print(f"Family metode       : {method_info['family']}")
    print(f"XC functional       : {method_info.get('xc')}")
    print(f"Dispersion          : {method_info.get('disp')}")
    print(f"Basis set           : {basis_name} ({pyscf_basis})")
    print(f"Jenis perhitungan   : {job}")
    print("Satuan koordinat    : Angstrom")

print("============================================================")


def print_coordinates(mol, title):

print("\n============================================================"
)
    print(title)

print("============================================================")
    coords = mol.atom_coords(unit="Angstrom")
    for i in range(mol.natm):
        symbol = clean_element_symbol(mol.atom_symbol(i))
        x, y, z = coords[i]
        print(f"{symbol:2s}  {x:14.8f}  {y:14.8f}  {z:14.8f}")


# ============================================================
# 12. ANALISIS STRUKTUR AKHIR
# ============================================================

def get_atom_records(mol):
    coords = mol.atom_coords(unit="Angstrom")
    atoms = []
    for i in range(mol.natm):
        symbol = clean_element_symbol(mol.atom_symbol(i))
        atomic_number = ATOMIC_NUMBERS.get(symbol)
        try:
            atomic_number = int(mol.atom_charge(i))
        except Exception:
            pass
        x, y, z = coords[i]
        atoms.append({
            "index": i + 1,
            "symbol": symbol,
```

```
            "atomic_number": atomic_number,
            "x": float(x),
            "y": float(y),
            "z": float(z),
        })
    return atoms


def distance_between_atoms(atom_a, atom_b):
    dx = atom_a["x"] - atom_b["x"]
    dy = atom_a["y"] - atom_b["y"]
    dz = atom_a["z"] - atom_b["z"]
    return math.sqrt(dx * dx + dy * dy + dz * dz)


def vector_between_atoms(atom_a, atom_b):
    return [
        atom_b["x"] - atom_a["x"],
        atom_b["y"] - atom_a["y"],
        atom_b["z"] - atom_a["z"],
    ]


def dot_product(vec_a, vec_b):
    return vec_a[0] * vec_b[0] + vec_a[1] * vec_b[1] + vec_a[2] *
vec_b[2]


def cross_product(vec_a, vec_b):
    return [
        vec_a[1] * vec_b[2] - vec_a[2] * vec_b[1],
        vec_a[2] * vec_b[0] - vec_a[0] * vec_b[2],
        vec_a[0] * vec_b[1] - vec_a[1] * vec_b[0],
    ]


def vector_norm(vec):
    return math.sqrt(dot_product(vec, vec))


def normalize_vector(vec):
    norm = vector_norm(vec)
    if norm < 1e-15:
        return [0.0, 0.0, 0.0]
    return [value / norm for value in vec]


def calculate_angle_degrees(atom_i, atom_j, atom_k):
    vec_ji = vector_between_atoms(atom_j, atom_i)
    vec_jk = vector_between_atoms(atom_j, atom_k)
    norm_ji = vector_norm(vec_ji)
    norm_jk = vector_norm(vec_jk)
    if norm_ji < 1e-15 or norm_jk < 1e-15:
        return None
    cos_theta = dot_product(vec_ji, vec_jk) / (norm_ji * norm_jk)
    cos_theta = max(-1.0, min(1.0, cos_theta))
    return math.degrees(math.acos(cos_theta))
```

```
def calculate_dihedral_degrees(atom_i, atom_j, atom_k, atom_l):
    p0 = [atom_i["x"], atom_i["y"], atom_i["z"]]
    p1 = [atom_j["x"], atom_j["y"], atom_j["z"]]
    p2 = [atom_k["x"], atom_k["y"], atom_k["z"]]
    p3 = [atom_l["x"], atom_l["y"], atom_l["z"]]

    b0 = [p0[n] - p1[n] for n in range(3)]
    b1 = [p2[n] - p1[n] for n in range(3)]
    b2 = [p3[n] - p2[n] for n in range(3)]
    b1_unit = normalize_vector(b1)

    v = [b0[n] - dot_product(b0, b1_unit) * b1_unit[n] for n in
range(3)]
    w = [b2[n] - dot_product(b2, b1_unit) * b1_unit[n] for n in
range(3)]

    if vector_norm(v) < 1e-15 or vector_norm(w) < 1e-15:
        return None

    x_value = dot_product(v, w)
    y_value = dot_product(cross_product(b1_unit, v), w)
    return math.degrees(math.atan2(y_value, x_value))


def infer_bonds(atoms, bond_scale=BOND_SCALE_DEFAULT,
bond_tolerance=BOND_TOLERANCE_DEFAULT):
    bonds = []
    for i in range(len(atoms)):
        for j in range(i + 1, len(atoms)):
            atom_i = atoms[i]
            atom_j = atoms[j]
            symbol_i = atom_i["symbol"]
            symbol_j = atom_j["symbol"]
            radius_i = COVALENT_RADII_ANGSTROM.get(symbol_i, 0.77)
            radius_j = COVALENT_RADII_ANGSTROM.get(symbol_j, 0.77)
            distance = distance_between_atoms(atom_i, atom_j)
            threshold = bond_scale * (radius_i + radius_j) +
bond_tolerance
            if symbol_i == "H" and symbol_j == "H" and distance >
0.90:
                continue
            if distance <= threshold:
                bonds.append({
                    "atom1": atom_i["index"],
                    "atom2": atom_j["index"],
                    "symbol1": symbol_i,
                    "symbol2": symbol_j,
                    "distance": float(distance),
                    "order": 1,
                })
    return bonds


def build_adjacency(atoms, bonds):
    adjacency = {atom["index"]: [] for atom in atoms}
```

```
    for bond in bonds:
        atom1 = bond["atom1"]
        atom2 = bond["atom2"]
        adjacency[atom1].append(atom2)
        adjacency[atom2].append(atom1)
    for atom_index in adjacency:
        adjacency[atom_index] = sorted(adjacency[atom_index])
    return adjacency


def infer_angles(atoms, bonds):
    atom_by_index = {atom["index"]: atom for atom in atoms}
    adjacency = build_adjacency(atoms, bonds)
    angles = []
    for center in sorted(adjacency):
        neighbors = adjacency[center]
        for a in range(len(neighbors)):
            for b in range(a + 1, len(neighbors)):
                atom_i_index = neighbors[a]
                atom_k_index = neighbors[b]
                angle_value = calculate_angle_degrees(
                    atom_by_index[atom_i_index],
                    atom_by_index[center],
                    atom_by_index[atom_k_index],
                )
                if angle_value is None:
                    continue
                angles.append({
                    "atom1": atom_i_index,
                    "atom2": center,
                    "atom3": atom_k_index,
                    "symbol1": atom_by_index[atom_i_index]["symbol"],
                    "symbol2": atom_by_index[center]["symbol"],
                    "symbol3": atom_by_index[atom_k_index]["symbol"],
                    "angle_degrees": float(angle_value),
                })
    return angles


def infer_dihedrals(atoms, bonds):
    atom_by_index = {atom["index"]: atom for atom in atoms}
    adjacency = build_adjacency(atoms, bonds)
    seen = set()
    dihedrals = []
    for bond in bonds:
        j = bond["atom1"]
        k = bond["atom2"]
        for i in adjacency[j]:
            if i == k:
                continue
            for l in adjacency[k]:
                if l == j:
                    continue
                dihedral_tuple = (i, j, k, l)
                reverse_tuple = tuple(reversed(dihedral_tuple))
                canonical = min(dihedral_tuple, reverse_tuple)
                if canonical in seen:
```

```
                    continue
                seen.add(canonical)
                dihedral_value = calculate_dihedral_degrees(
                    atom_by_index[i],
                    atom_by_index[j],
                    atom_by_index[k],
                    atom_by_index[l],
                )
                if dihedral_value is None:
                    continue
                dihedrals.append({
                    "atom1": i,
                    "atom2": j,
                    "atom3": k,
                    "atom4": l,
                    "symbol1": atom_by_index[i]["symbol"],
                    "symbol2": atom_by_index[j]["symbol"],
                    "symbol3": atom_by_index[k]["symbol"],
                    "symbol4": atom_by_index[l]["symbol"],
                    "dihedral_degrees": float(dihedral_value),
                })
    return dihedrals


def calculate_distance_matrix(atoms):
    return [
        [float(distance_between_atoms(atom_i, atom_j)) for atom_j in
atoms]
        for atom_i in atoms
    ]


# ============================================================
# 13. PENULISAN FILE HASIL
# ============================================================

def make_output_dir(args, method_name, basis_name):
    timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
    folder_name = safe_filename(

f"{MOLECULE_NAME}_{MOLECULE_FORMULA}_{method_name}_{basis_name}_{args.
job}_{timestamp}"
    )
    if args.output_dir:
        output_path = Path(args.output_dir)
    else:
        output_path = Path(OUTPUT_ROOT_DEFAULT) / folder_name
    output_path.mkdir(parents=True, exist_ok=True)
    return output_path


def make_output_prefix(method_name, basis_name, job):
    timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
    return
safe_filename(f"{MOLECULE_NAME}_{MOLECULE_FORMULA}_{method_name}_{basi
s_name}_{job}_{timestamp}")
```

```
def format_xyz_coordinate_block(atoms):
    return "\n".join(
        f"{atom['symbol']:2s}  {atom['x']:14.8f}  {atom['y']:14.8f}
{atom['z']:14.8f}"
        for atom in atoms
    )


def identity_header_lines():
    return [
        f"author: {AUTHOR_DEFAULT}",
        f"Telp.: {AUTHOR_PHONE_DEFAULT}",
    ]


def identity_header_text():
    return "\n".join(identity_header_lines())


def write_txt_coordinates(path, atoms, metadata):
    lines = identity_header_lines() + [""]
    lines.append("KOORDINAT AKHIR MOLEKUL")
    lines.append("=" * 60)
    for key, value in metadata.items():
        lines.append(f"{key}: {value}")
    lines.append("=" * 60)
    lines.append("Format: SimbolAtom        X              Y
Z")
    lines.append("Satuan: Angstrom")
    lines.append("")
    lines.append(format_xyz_coordinate_block(atoms))
    lines.append("")
    lines.append("BLOK KOORDINAT SIAP SALIN UNTUK PySCF/XYZ")
    lines.append("=" * 60)
    lines.append('\"\"\"')
    lines.append(format_xyz_coordinate_block(atoms))
    lines.append('\"\"\"')
    lines.append("")
    Path(path).write_text("\n".join(lines), encoding="utf-8")


def write_xyz_file(path, atoms, metadata):
    title = (
        f"{metadata['Nama molekul']} | author: {AUTHOR_DEFAULT} |
Telp.: {AUTHOR_PHONE_DEFAULT} | "
        f"{metadata['Metode']} / {metadata['Basis set']} | "
        f"E = {metadata['Energi total Hartree']} Hartree | "
        f"SCF converged = {metadata['SCF konvergen']}"
    )
    lines = [str(len(atoms)), title,
format_xyz_coordinate_block(atoms)]
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_mol_file(path, atoms, bonds, title=None):
    title = title or MOLECULE_NAME
```

```
    lines = []
    lines.append(title[:80])
    lines.append(f"  author: {AUTHOR_DEFAULT} | Telp.:
{AUTHOR_PHONE_DEFAULT}")
    lines.append("  Generated final 3D coordinates and inferred bonds
by PySCF Python Script")
    lines.append(f"{len(atoms):3d}{len(bonds):3d}  0  0  0  0
999 V2000")

    for atom in atoms:
        lines.append(
            f"{atom['x']:10.4f}"
            f"{atom['y']:10.4f}"
            f"{atom['z']:10.4f} "
            f"{atom['symbol']:<3s}"
            " 0  0  0  0  0  0  0  0  0  0  0  0"
        )

    for bond in bonds:
        lines.append(
            f"{bond['atom1']:3d}"
            f"{bond['atom2']:3d}"
            f"{int(bond.get('order', 1)):3d}"
            "  0  0  0  0"
        )

    lines.append("M  END")
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_sdf_file(path, atoms, bonds, metadata):
    mol_text_path = Path(path).with_suffix(".tmp_mol")
    write_mol_file(mol_text_path, atoms, bonds, title=metadata["Nama
molekul"])
    mol_text = mol_text_path.read_text(encoding="utf-8")
    mol_text_path.unlink(missing_ok=True)

    lines = [mol_text.rstrip("\n")]
    for key, value in metadata.items():
        lines.append(f">  <{key}>")
        lines.append(str(value))
        lines.append("")
    lines.append("$$$$")
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_pdb_file(path, atoms, bonds, metadata):
    lines = []
    lines.append(f"HEADER    {metadata['Nama molekul'][:50]}")
    lines.append(f"REMARK    author: {AUTHOR_DEFAULT}")
    lines.append(f"REMARK    Telp.: {AUTHOR_PHONE_DEFAULT}")
    lines.append(f"TITLE     {metadata['Metode']} / {metadata['Basis
set']}")
    lines.append(f"REMARK    Energy Hartree: {metadata['Energi total
Hartree']}")
    lines.append(f"REMARK    SCF converged: {metadata['SCF
konvergen']}")
```

```
    for atom in atoms:
        atom_index = atom["index"]
        symbol = atom["symbol"]
        lines.append(
            f"HETATM{atom_index:5d} "
            f"{symbol:<4s}"
            f" MOL     1    "
            f"{atom['x']:8.3f}"
            f"{atom['y']:8.3f}"
            f"{atom['z']:8.3f}"
            f"  1.00  0.00          "
            f"{symbol:>2s}"
        )

    adjacency = build_adjacency(atoms, bonds)
    for atom_index in sorted(adjacency):
        if not adjacency[atom_index]:
            continue
        conect_targets = "".join(f"{target:5d}" for target in
adjacency[atom_index])
        lines.append(f"CONECT{atom_index:5d}{conect_targets}")

    lines.append("END")
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_cml_file(path, atoms, bonds, metadata):
    lines = []
    lines.append('<?xml version="1.0" encoding="UTF-8"?>')
    lines.append('<cml xmlns="http://www.xml-cml.org/schema">')
    lines.append(f'  <molecule
id="{escape(safe_filename(metadata["Nama molekul"]))}">')
    lines.append("    <metadataList>")
    for key, value in metadata.items():
        lines.append(f'      <metadata name="{escape(str(key))}"
content="{escape(str(value))}"/>')
    lines.append("    </metadataList>")
    lines.append("    <atomArray>")
    for atom in atoms:
        lines.append(
            f'      <atom id="a{atom["index"]}" '
            f'elementType="{escape(atom["symbol"])}" '
            f'x3="{atom["x"]:.8f}" '
            f'y3="{atom["y"]:.8f}" '
            f'z3="{atom["z"]:.8f}"/>'
        )
    lines.append("    </atomArray>")
    lines.append("    <bondArray>")
    for bond in bonds:
        lines.append(
            f'      <bond atomRefs2="a{bond["atom1"]}
a{bond["atom2"]}" '
            f'order="{int(bond.get("order", 1))}"/>'
        )
    lines.append("    </bondArray>")
    lines.append("  </molecule>")
```

```
    lines.append("</cml>")
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_json_structure_file(path, atoms, bonds, angles, dihedrals,
distance_matrix, metadata):
    structure_data = {
        "metadata": metadata,
        "atoms": atoms,
        "bonds": bonds,
        "angles": angles,
        "dihedrals": dihedrals,
        "distance_matrix_angstrom": distance_matrix,
        "notes": {
            "bond_inference": (
                "Ikatan diinferensi dari jarak antaratom dan jari-jari
kovalen. "
                "Orde ikatan ditulis 1 untuk tujuan visualisasi
struktur 3D."
            ),
            "indexing": "Semua indeks atom memakai sistem 1-based.",
        },
    }
    Path(path).write_text(
        json.dumps(structure_data, indent=2, ensure_ascii=False),
        encoding="utf-8",
    )


def write_log_file(path, atoms, bonds, angles, dihedrals,
distance_matrix, metadata, output_files):
    lines = identity_header_lines() + [""]
    lines.append("=" * 80)
    lines.append("LAPORAN AKHIR PERHITUNGAN KIMIA KOMPUTASI PySCF")
    lines.append("=" * 80)
    for key, value in metadata.items():
        lines.append(f"{key:28s}: {value}")

    lines.append("")
    lines.append("=" * 80)
    lines.append("KOORDINAT AKHIR MOLEKUL")
    lines.append("=" * 80)
    lines.append("No.  Atom            X             Y
Z       Satuan")
    lines.append("-" * 80)
    for atom in atoms:
        lines.append(
            f"{atom['index']:3d}  {atom['symbol']:2s}  "
            f"{atom['x']:14.8f}  {atom['y']:14.8f}  {atom['z']:14.8f}
Angstrom"
        )

    lines.append("")
    lines.append("=" * 80)
    lines.append("DAFTAR IKATAN HASIL INFERENSI")
    lines.append("=" * 80)
```

```
    lines.append("No.  Atom-1       Atom-2       Orde    Jarak
(Angstrom)")
    lines.append("-" * 80)
    if bonds:
        for number, bond in enumerate(bonds, start=1):
            lines.append(
                f"{number:3d}  "
                f"{bond['atom1']:3d}-{bond['symbol1']:<2s}      "
                f"{bond['atom2']:3d}-{bond['symbol2']:<2s}      "
                f"{int(bond.get('order', 1)):3d}     "
                f"{bond['distance']:12.6f}"
            )
    else:
        lines.append("Tidak ada ikatan terdeteksi. Periksa --bond-
scale atau --bond-tolerance.")

    lines.append("")
    lines.append("=" * 80)
    lines.append("DAFTAR SUDUT IKATAN")
    lines.append("=" * 80)
    lines.append("No.  Atom-1       Atom-2/Pusat Atom-3       Sudut
(derajat)")
    lines.append("-" * 80)
    if angles:
        for number, angle in enumerate(angles, start=1):
            lines.append(
                f"{number:3d}  "
                f"{angle['atom1']:3d}-{angle['symbol1']:<2s}      "
                f"{angle['atom2']:3d}-{angle['symbol2']:<2s}        "
                f"{angle['atom3']:3d}-{angle['symbol3']:<2s}      "
                f"{angle['angle_degrees']:12.6f}"
            )
    else:
        lines.append("Tidak ada sudut ikatan terdeteksi.")

    lines.append("")
    lines.append("=" * 80)
    lines.append("DAFTAR SUDUT DIHEDRAL")
    lines.append("=" * 80)
    lines.append("No.  Atom-1       Atom-2       Atom-3       Atom-4
Dihedral (derajat)")
    lines.append("-" * 80)
    if dihedrals:
        for number, dihedral in enumerate(dihedrals, start=1):
            lines.append(
                f"{number:3d}  "
                f"{dihedral['atom1']:3d}-{dihedral['symbol1']:<2s}
"
                f"{dihedral['atom2']:3d}-{dihedral['symbol2']:<2s}
"
                f"{dihedral['atom3']:3d}-{dihedral['symbol3']:<2s}
"
                f"{dihedral['atom4']:3d}-{dihedral['symbol4']:<2s}
"
                f"{dihedral['dihedral_degrees']:12.6f}"
            )
    else:
```

```
        lines.append("Tidak ada sudut dihedral terdeteksi.")

    lines.append("")
    lines.append("=" * 80)
    lines.append("MATRIKS JARAK ANTARATOM (Angstrom)")
    lines.append("=" * 80)
    header = "Atom ".ljust(8) + "".join(f"{atom['index']:>10d}" for
atom in atoms)
    lines.append(header)
    lines.append("-" * max(80, len(header)))
    for atom, row in zip(atoms, distance_matrix):
        row_text = f"{atom['index']:3d}-{atom['symbol']:<2s}
".ljust(8)
        row_text += "".join(f"{value:10.4f}" for value in row)
        lines.append(row_text)

    lines.append("")
    lines.append("=" * 80)
    lines.append("FILE HASIL YANG DIBUAT")
    lines.append("=" * 80)
    for label, filename in output_files.items():
        lines.append(f"{label:18s}: {filename}")

    lines.append("")
    lines.append("CATATAN")
    lines.append("- Format .xyz hanya menyimpan koordinat, bukan
konektivitas ikatan.")
    lines.append("- Format .mol, .sdf, .pdb, dan .cml lebih tepat
untuk visualisasi struktur 3D.")
    lines.append("- Ikatan diinferensi dari jarak antaratom dan jari-
jari kovalen.")
    lines.append("- Jika konektivitas visual belum sesuai, ubah --
bond-scale atau --bond-tolerance.")
    lines.append("")

    Path(path).write_text("\n".join(lines), encoding="utf-8")


def save_final_structure_outputs(mol, mf, energy, method_name,
method_info, basis_name, pyscf_basis, args, output_dir,
running_output_log_path=None):
    atoms = get_atom_records(mol)
    bonds = infer_bonds(atoms, bond_scale=args.bond_scale,
bond_tolerance=args.bond_tolerance)
    angles = infer_angles(atoms, bonds)
    dihedrals = infer_dihedrals(atoms, bonds)
    distance_matrix = calculate_distance_matrix(atoms)

    prefix = make_output_prefix(method_name, basis_name, args.job)

    metadata = {
        "author": AUTHOR_DEFAULT,
        "Telp.": AUTHOR_PHONE_DEFAULT,
        "Nama molekul": MOLECULE_NAME,
        "Rumus molekul": MOLECULE_FORMULA,
        "SMILES": SMILES_DEFAULT,
        "IUPAC-like name": IUPAC_NAME_DEFAULT,
```

```
        "Catatan riset": RESEARCH_NOTE_DEFAULT,
        "Jumlah atom": mol.natm,
        "Jumlah elektron": mol.nelectron,
        "Muatan total": args.charge,
        "Spin PySCF": args.spin,
        "Metode": method_name,
        "Family metode": method_info["family"],
        "XC functional": method_info.get("xc"),
        "Dispersion": method_info.get("disp"),
        "Basis set": basis_name,
        "Basis PySCF": pyscf_basis,
        "Jenis perhitungan": args.job,
        "Energi total Hartree": f"{energy:.12f}",
        "SCF konvergen": bool(getattr(mf, "converged", False)),
        "File checkpoint": getattr(mf, "chkfile", ""),
        "Satuan koordinat": "Angstrom",
        "Bond scale": args.bond_scale,
        "Bond tolerance": args.bond_tolerance,
        "Waktu simpan": datetime.now().strftime("%Y-%m-%d %H:%M:%S"),
    }

    output_files = {
        "txt_coordinates": str(output_dir /
f"{prefix}_koordinat_akhir.txt"),
        "xyz_structure": str(output_dir / f"{prefix}.xyz"),
        "mol_structure": str(output_dir / f"{prefix}.mol"),
        "sdf_structure": str(output_dir / f"{prefix}.sdf"),
        "pdb_structure": str(output_dir / f"{prefix}.pdb"),
        "cml_structure": str(output_dir / f"{prefix}.cml"),
        "json_structure": str(output_dir /
f"{prefix}_struktur_lengkap.json"),
        "log_report": str(output_dir / f"{prefix}.log"),
    }

    if running_output_log_path:
        output_files["running_output_log"] =
str(running_output_log_path)

    write_txt_coordinates(output_files["txt_coordinates"], atoms,
metadata)
    write_xyz_file(output_files["xyz_structure"], atoms, metadata)
    write_mol_file(output_files["mol_structure"], atoms, bonds,
title=MOLECULE_NAME)
    write_sdf_file(output_files["sdf_structure"], atoms, bonds,
metadata)
    write_pdb_file(output_files["pdb_structure"], atoms, bonds,
metadata)
    write_cml_file(output_files["cml_structure"], atoms, bonds,
metadata)
    write_json_structure_file(output_files["json_structure"], atoms,
bonds, angles, dihedrals, distance_matrix, metadata)
    write_log_file(output_files["log_report"], atoms, bonds, angles,
dihedrals, distance_matrix, metadata, output_files)

    return output_files, {
        "atoms": atoms,
        "bonds": bonds,
```

```
        "angles": angles,
        "dihedrals": dihedrals,
    }


# ============================================================
# 14. INFORMASI ORBITAL
# ============================================================

def print_orbital_info(mf):
    mo_energy = getattr(mf, "mo_energy", None)
    mo_occ = getattr(mf, "mo_occ", None)

    if mo_energy is None or mo_occ is None:
        print("\nInformasi orbital tidak tersedia.")
        return

    if isinstance(mo_energy, tuple) or isinstance(mo_occ, tuple):
        print("\nSistem terbuka-shell terdeteksi.")
        print("Pelaporan HOMO-LUMO sederhana dilewati.")
        return

    occupied = [i for i, occ in enumerate(mo_occ) if occ > 0]
    virtual = [i for i, occ in enumerate(mo_occ) if occ == 0]

    if not occupied or not virtual:
        print("\nHOMO/LUMO tidak dapat ditentukan dari okupansi
orbital.")
        return

    homo_index = max(occupied)
    lumo_index = min(virtual)

    homo_hartree = mo_energy[homo_index]
    lumo_hartree = mo_energy[lumo_index]
    gap_hartree = lumo_hartree - homo_hartree

    hartree_to_ev = 27.211386245988
    homo_ev = homo_hartree * hartree_to_ev
    lumo_ev = lumo_hartree * hartree_to_ev
    gap_ev = gap_hartree * hartree_to_ev


print("\n============================================================"
)
    print("INFORMASI ORBITAL SEDERHANA")

print("============================================================")
    print(f"HOMO index          : {homo_index}")
    print(f"LUMO index          : {lumo_index}")
    print(f"Energi HOMO         : {homo_hartree: .10f} Hartree")
    print(f"Energi LUMO         : {lumo_hartree: .10f} Hartree")
    print(f"Gap HOMO-LUMO       : {gap_hartree: .10f} Hartree")
    print(f"Energi HOMO         : {homo_ev: .6f} eV")
    print(f"Energi LUMO         : {lumo_ev: .6f} eV")
    print(f"Gap HOMO-LUMO       : {gap_ev: .6f} eV")
```

```
print("============================================================")


# ============================================================
# 15. ARGUMEN TERMINAL
# ============================================================

def parse_arguments():
    parser = argparse.ArgumentParser(
        description="Template generik perhitungan kimia komputasi
molekul dengan PySCF. Default: LAH-01 Letrozole-Anastrozole Hybrid."
    )

    parser.add_argument("--name", default=MOLECULE_NAME_DEFAULT,
help="Nama molekul.")
    parser.add_argument("--formula", default=FORMULA_DEFAULT,
help="Rumus molekul opsional.")

    parser.add_argument("--xyz-file", default=None, help="File XYZ
standar atau blok koordinat XYZ.")
    parser.add_argument("--coord-file", default=None, help="File teks
koordinat: SimbolAtom X Y Z.")

    parser.add_argument("--method", default=METHOD_DEFAULT,
help=f"Metode komputasi. Default: {METHOD_DEFAULT}")
    parser.add_argument("--xc", default=None, help="Functional PySCF
untuk --method CUSTOM, contoh: 'm06-2x'.")
    parser.add_argument("--disp", default=None, help="Koreksi dispersi
manual, contoh: d3bj, d3zero, d4.")

    parser.add_argument("--basis", default=BASIS_DEFAULT, help=f"Basis
set. Default: {BASIS_DEFAULT}")
    parser.add_argument("--basis-custom", default=None, help="Nama
basis PySCF untuk --basis CUSTOM.")

    parser.add_argument("--charge", type=int, default=CHARGE_DEFAULT,
help=f"Muatan total molekul. Default: {CHARGE_DEFAULT}")
    parser.add_argument(
        "--spin",
        type=int,
        default=SPIN_DEFAULT,
        help=f"Spin PySCF = jumlah elektron alfa - beta. Default:
{SPIN_DEFAULT}",
    )

    parser.add_argument(
        "--job",
        choices=["singlepoint", "optimize"],
        default=JOB_DEFAULT,
        help=f"Jenis perhitungan: singlepoint atau optimize. Default:
{JOB_DEFAULT}",
    )
    parser.add_argument("--optimize", action="store_true", help="Alias
lama untuk --job optimize.")
```

```
    parser.add_argument("--grid", type=int,
default=DFT_GRID_LEVEL_DEFAULT, help=f"Level grid DFT. Default:
{DFT_GRID_LEVEL_DEFAULT}")
    parser.add_argument("--conv-tol", type=float,
default=CONV_TOL_DEFAULT, help=f"Toleransi konvergensi SCF. Default:
{CONV_TOL_DEFAULT}")
    parser.add_argument("--max-cycle", type=int,
default=MAX_CYCLE_DEFAULT, help=f"Maksimum iterasi SCF. Default:
{MAX_CYCLE_DEFAULT}")
    parser.add_argument("--verbose", type=int,
default=VERBOSE_DEFAULT, help=f"Level output PySCF. Default:
{VERBOSE_DEFAULT}")
    parser.add_argument("--density-fit", action="store_true",
default=USE_DENSITY_FITTING_DEFAULT, help="Aktifkan density fitting.")

    parser.add_argument("--output-dir", default=None, help="Direktori
output. Jika kosong, dibuat otomatis di hasil_pyscf/.")
    parser.add_argument("--bond-scale", type=float,
default=BOND_SCALE_DEFAULT, help=f"Faktor skala inferensi ikatan.
Default: {BOND_SCALE_DEFAULT}")
    parser.add_argument("--bond-tolerance", type=float,
default=BOND_TOLERANCE_DEFAULT, help=f"Toleransi inferensi ikatan
dalam Angstrom. Default: {BOND_TOLERANCE_DEFAULT}")

    parser.add_argument("--list", action="store_true", help="Tampilkan
daftar metode, basis set, dan contoh pemakaian.")

    return parser.parse_args()


# ============================================================
# 16. PROGRAM UTAMA
# ============================================================

def main():
    global MOLECULE_NAME, MOLECULE_FORMULA, BASIS_LABEL

    args = parse_arguments()

    if args.list:
        print_available_options()
        return

    if args.optimize:
        args.job = "optimize"

    validate_nonnegative_int(args.grid, "grid")
    validate_positive_float(args.conv_tol, "conv_tol")
    validate_positive_int(args.max_cycle, "max_cycle")
    validate_positive_float(args.bond_scale, "bond_scale")
    validate_nonnegative_int(args.verbose, "verbose")

    MOLECULE_NAME = args.name.strip() if args.name else
MOLECULE_NAME_DEFAULT
    MOLECULE_FORMULA = args.formula.strip() if args.formula else "-"

    method_name, method_info = resolve_method(args)
```

```
    basis_name, pyscf_basis = resolve_basis(args)
    BASIS_LABEL = basis_name

    # Direktori output dan file running log dibuat sebelum PySCF mulai
bekerja,
    # sehingga pesan proses SCF/DFT dan pesan error tetap tersimpan
walaupun perhitungan gagal.
    output_dir = make_output_dir(args, method_name, basis_name)
    running_log_prefix = make_output_prefix(method_name, basis_name,
args.job)
    running_output_log_path = output_dir /
f"{running_log_prefix}_running_output.log"

    original_stdout = sys.stdout
    original_stderr = sys.stderr
    status_akhir = "SELESAI"

    with open(running_output_log_path, "w", encoding="utf-8",
buffering=1) as running_log_file:
        sys.stdout = TeeOutput(original_stdout, running_log_file)
        sys.stderr = TeeOutput(original_stderr, running_log_file)
        try:
            write_running_log_header(
                log_path=running_output_log_path,
                args=args,
                method_name=method_name,
                basis_name=basis_name,
                pyscf_basis=pyscf_basis,
            )

            coordinate_atoms, coordinate_source =
get_coordinate_atoms(args)
            atom_block = format_atom_block(coordinate_atoms)

            mol = build_molecule(
                atom_block=atom_block,
                pyscf_basis=pyscf_basis,
                charge=args.charge,
                spin=args.spin,
                verbose=args.verbose,
            )

            print_molecule_info(
                mol=mol,
                method_name=method_name,
                method_info=method_info,
                basis_name=basis_name,
                pyscf_basis=pyscf_basis,
                charge=args.charge,
                spin=args.spin,
                job=args.job,
                coordinate_source=coordinate_source,
            )

            print_coordinates(mol, "KOORDINAT AWAL MOLEKUL")

            mf = build_calculator(
```

```
                mol=mol,
                method_name=method_name,
                method_info=method_info,
                grid_level=args.grid,
                conv_tol=args.conv_tol,
                max_cycle=args.max_cycle,
                use_density_fitting=args.density_fit,
            )

            if args.job == "optimize":
                mol_opt = optimize_geometry(mf)
                print_coordinates(mol_opt, "KOORDINAT HASIL OPTIMASI
GEOMETRI")
                mf = build_calculator(
                    mol=mol_opt,
                    method_name=method_name,
                    method_info=method_info,
                    grid_level=args.grid,
                    conv_tol=args.conv_tol,
                    max_cycle=args.max_cycle,
                    use_density_fitting=args.density_fit,
                )


print("\n============================================================="
)
            print("PERHITUNGAN ENERGI DIMULAI")

print("=============================================================")
            print(f"Nama molekul : {MOLECULE_NAME}")
            print(f"Metode       : {method_name}")
            print(f"XC           : {method_info.get('xc')}")
            print(f"Dispersion   : {method_info.get('disp')}")
            print(f"Basis set    : {basis_name} ({pyscf_basis})")
            print(f"Grid DFT     : {args.grid}")
            print(f"Conv tol     : {args.conv_tol}")
            print(f"Max cycle    : {args.max_cycle}")
            print(f"Density fit  : {args.density_fit}")

print("=============================================================\n"
)

            try:
                energy = mf.kernel()
            except Exception as exc:
                status_akhir = "GAGAL"
                print("\nERROR: Perhitungan PySCF gagal.")
                print(str(exc))
                if method_info.get("disp"):
                    print("\nKemungkinan penyebab: koreksi dispersi
belum tersedia pada instalasi PySCF Anda.")
                    print("Coba salah satu opsi berikut:")
                    print("1. Jalankan tanpa dispersi:")
                    print("   python lah01-letrozole-anastrozole-
hybrid.py --method B3LYP --basis def2-SVP")
                    print("2. Instal dukungan dispersi sesuai
lingkungan Python Anda, misalnya:")
```

```
                    print("   pip install pyscf-dispersion simple-
dftd3")
                print("\nPENTING:")
                print(f"Teks running/error sudah otomatis tersimpan
di: {running_output_log_path}")
                sys.exit(1)


print("\n============================================================"
)
            print("HASIL PERHITUNGAN")

print("============================================================")
            print(f"Nama molekul       : {MOLECULE_NAME}")
            print(f"Rumus molekul      : {MOLECULE_FORMULA}")
            print(f"Metode             : {method_name}")
            print(f"Basis set          : {basis_name}")
            print(f"Energi total       : {energy: .12f} Hartree")
            print(f"SCF konvergen      : {mf.converged}")
            print(f"File checkpoint    : {mf.chkfile}")

            if not mf.converged:
                print("\nPERINGATAN:")
                print("SCF belum konvergen.")
                print("Coba salah satu langkah berikut:")
                print("1. Naikkan --max-cycle, misalnya --max-cycle
250")
                print("2. Longgarkan sementara --conv-tol, misalnya --
conv-tol 1e-7")
                print("3. Gunakan basis lebih kecil untuk uji awal,
misalnya STO-3G")
                print("4. Optimasi geometri awal di Avogadro atau
jalankan --job optimize")


print("============================================================")

            output_files, structure_summary =
save_final_structure_outputs(
                mol=mf.mol,
                mf=mf,
                energy=energy,
                method_name=method_name,
                method_info=method_info,
                basis_name=basis_name,
                pyscf_basis=pyscf_basis,
                args=args,
                output_dir=output_dir,
                running_output_log_path=running_output_log_path,
            )


print("\n============================================================"
)
            print("FILE HASIL AKHIR BERHASIL DISIMPAN")

print("============================================================")
```

```
            print(f"Direktori output            : {output_dir}")
            print(f"Jumlah ikatan terdeteksi    :
{len(structure_summary['bonds'])}")
            print(f"Jumlah sudut terdeteksi     :
{len(structure_summary['angles'])}")
            print(f"Jumlah dihedral terdeteksi  :
{len(structure_summary['dihedrals'])}")
            print("")
            for label, filename in output_files.items():
                print(f"{label:18s}: {filename}")

print("============================================================")
            print("Catatan:")
            print("- .txt berisi koordinat akhir molekul.")
            print("- .xyz berisi koordinat 3D standar.")
            print("- .mol/.sdf/.pdb/.cml dapat dipakai untuk
visualisasi struktur 3D.")
            print("- .json dan .log menyimpan data ikatan, sudut,
dihedral, jarak, dan teks running.")

print("============================================================")

            print_orbital_info(mf)
            print("\nSELESAI.")

        except SystemExit:
            if status_akhir != "GAGAL":
                status_akhir = "DIHENTIKAN"
            raise
        except Exception:
            status_akhir = "GAGAL"
            print("\nERROR TAK TERDUGA:", file=sys.stderr)
            import traceback
            traceback.print_exc()
            print("\nPENTING:")
            print(f"Teks running/error sudah otomatis tersimpan di:
{running_output_log_path}")
            raise
        finally:
            try:
                write_running_log_footer(running_output_log_path,
status=status_akhir)
            finally:
                sys.stdout = original_stdout
                sys.stderr = original_stderr


if __name__ == "__main__":
    main()
```

BAB 5

IMPLEMENTASI PySCF UNTUK HIBRIDA LETROZOLE-ANASTROZOLE

Pengantar Bab

Bab ini memuat implementasi program PySCF untuk contoh senyawa hibrida Letrozole-Anastrozole. Kode program dipertahankan sebagai bagian utama buku karena menjadi jembatan langsung antara rancangan proposal dan pelaksanaan perhitungan di komputer.

Pembaca dianjurkan membaca kode secara bertahap: identitas molekul, koordinat XYZ, metode, basis set, fungsi perhitungan, analisis keluaran, hingga penulisan hasil. Dengan cara ini, skrip tidak hanya dijalankan, tetapi dipahami sebagai dokumen riset yang dapat diaudit.

Setelah pengantar ini, pembahasan dilanjutkan dengan orientasi bab, peta kompetensi, uraian materi inti, contoh kerja, dan catatan metodologis yang diperlukan untuk menuntun pembaca dari konsep menuju praktik.

Dalam pembelajaran, kode seperti ini dapat dijadikan template. Mahasiswa dapat mengganti nama molekul, rumus, SMILES, koordinat XYZ, metode, dan basis set secara terkontrol. Setiap perubahan harus dicatat dalam logbook agar hasil komputasi dapat ditelusuri.

Bab ini memuat contoh file program riset. Kode program di dalam buku dipertahankan utuh karena menjadi bagian penting dari naskah sumber. Pembaca sebaiknya membaca kode seperti membaca metode penelitian: mulai dari metadata, daftar metode, basis set, konfigurasi default, koordinat molekul, fungsi validasi, kalkulator, analisis struktur, hingga penulisan laporan akhir.

Orientasi Bab

Subbab ini menguraikan Orientasi Bab. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# FILE RISET KIMIA KOMPUTASI PySCF UNTUK HIBRIDA LETROZOLE-ANASTROZOLE
# ============================================================
#
# Molekul default : LAH-01 Letrozole-Anastrozole Hybrid
# Rumus molekul   : C24H22N6
# Program         : PySCF
#
# Metode default yang dipilih:
#   B3LYP-D3BJ / def2-SVP
#
# Alasan ringkas:
#   - LAH-01 Letrozole-Anastrozole Hybrid adalah molekul organik netral tertutup-shell.
#   - B3LYP umum dipakai untuk molekul organik.
#   - Koreksi dispersi D3BJ membantu interaksi non-kovalen dan konformasi.
#   - def2-SVP cukup seimbang untuk latihan/skrining awal tanpa terlalu berat.
#
# Catatan penting:
#   - Untuk memakai koreksi dispersi D3BJ, PySCF memerlukan dukungan
#     pyscf-dispersion/simple-dftd3 pada beberapa instalasi.
#   - Jika belum tersedia, jalankan metode B3LYP biasa:
#       python lah01-letrozole-anastrozole-hybrid.py --method B3LYP --basis def2-SVP
#
# Contoh pemakaian:
#   python lah01-letrozole-anastrozole-hybrid.py
#   python lah01-letrozole-anastrozole-hybrid.py --list
#   python lah01-letrozole-anastrozole-hybrid.py --method B3LYP --basis def2-SVP
#   python lah01-letrozole-anastrozole-hybrid.py --method PBE0 --basis def2-TZVP
#   python lah01-letrozole-anastrozole-hybrid.py --method CUSTOM --xc "m06-2x" --basis
def2-TZVP
#   python lah01-letrozole-anastrozole-hybrid.py --xyz-file molekul.xyz --name "Molekul
Baru" --formula "C2H6O"
#   python lah01-letrozole-anastrozole-hybrid.py --coord-file koordinat.txt --method B3LYP
--basis "6-31G**"
#   python lah01-letrozole-anastrozole-hybrid.py --job optimize --method B3LYP --basis
def2-SVP
#
# ============================================================
```

```
import argparse
import json
import math
import sys
from datetime import datetime
from html import escape
from pathlib import Path
```

```
from pyscf import dft, gto, scf
```

```
# ============================================================
```

1. DAFTAR METODE KOMPUTASI

Subbab ini menguraikan # 1. DAFTAR METODE KOMPUTASI. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================
# family:
#   HF  = Hartree-Fock
#   DFT = Kohn-Sham DFT
# xc:
#   String functional yang dibaca oleh PySCF.
# disp:
#   None       = tanpa koreksi dispersi eksplisit.
#   "d3bj"    = koreksi dispersi D3 dengan Becke-Johnson damping.
#
# CUSTOM:
#   Dipakai bersama argumen --xc.
```

```
AVAILABLE_METHODS = {
  "HF": {
    "family": "HF",
    "xc": None,
    "disp": None,
    "description": "Hartree-Fock; RHF untuk spin 0, UHF untuk spin bukan 0.",
  },
  "LDA": {
    "family": "DFT",
    "xc": "lda,vwn",
    "disp": None,
    "description": "DFT LDA sederhana.",
  },
  "PBE": {
    "family": "DFT",
    "xc": "pbe,pbe",
    "disp": None,
    "description": "DFT GGA PBE.",
  },
  "BLYP": {
    "family": "DFT",
    "xc": "b88,lyp",
    "disp": None,
    "description": "DFT GGA BLYP.",
  },
  "B3LYP": {
    "family": "DFT",
    "xc": "b3lyp",
    "disp": None,
    "description": "DFT hybrid B3LYP, umum untuk molekul organik.",
  },
  "B3LYP-D3BJ": {
    "family": "DFT",
    "xc": "b3lyp",
```

```
    "disp": "d3bj",
    "description": "B3LYP dengan koreksi dispersi D3BJ; default untuk kandidat hibrida LAH-01.",
  },
  "PBE0": {
    "family": "DFT",
    "xc": "pbe0",
    "disp": None,
    "description": "DFT hybrid PBE0.",
  },
  "PBE0-D3BJ": {
    "family": "DFT",
    "xc": "pbe0",
    "disp": "d3bj",
    "description": "PBE0 dengan koreksi dispersi D3BJ.",
  },
  "TPSS": {
    "family": "DFT",
    "xc": "tpss",
    "disp": None,
    "description": "Meta-GGA TPSS.",
  },
  "M06": {
    "family": "DFT",
    "xc": "m06",
    "disp": None,
    "description": "Meta-hybrid Minnesota M06 jika didukung instalasi PySCF/libxc.",
  },
  "CAM-B3LYP": {
    "family": "DFT",
    "xc": "cam-b3lyp",
    "disp": None,
    "description": "Range-separated hybrid CAM-B3LYP.",
  },
  "WB97X-D": {
```

```
        "family": "DFT",
        "xc": "wb97x-d",
        "disp": None,
        "description": "Range-separated functional dengan dispersi bawaan jika didukung.",
    },
    "CUSTOM": {
        "family": "DFT",
        "xc": None,
        "disp": None,
        "description": "Functional DFT custom dari argumen --xc.",
    },
}

# ============================================================
```

2. DAFTAR BASIS SET

Subbab ini menguraikan # 2. DAFTAR BASIS SET. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

AVAILABLE_BASIS_SETS = {
    "STO-3G": "sto-3g",
    "3-21G": "3-21g",
    "6-31G": "6-31g",
    "6-31G*": "6-31g*",
    "6-31G**": "6-31g**",
    "6-31+G*": "6-31+g*",
    "6-31+G**": "6-31+g**",
    "def2-SVP": "def2-svp",
    "def2-TZVP": "def2-tzvp",
    "def2-QZVP": "def2-qzvp",
    "cc-pVDZ": "cc-pvdz",
    "cc-pVTZ": "cc-pvtz",
    "aug-cc-pVDZ": "aug-cc-pvdz",
    "CUSTOM": None,
}

# ============================================================
```

3. KONFIGURASI DEFAULT

Subbab ini menguraikan # 3. KONFIGURASI DEFAULT. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================
```

```
MOLECULE_NAME_DEFAULT = "LAH-01 Letrozole-Anastrozole Hybrid"
```

FORMULA_DEFAULT = "C24H22N6"

Subbab ini menguraikan FORMULA_DEFAULT = "C24H22N6". Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

METHOD_DEFAULT = "B3LYP-D3BJ"

Subbab ini menguraikan METHOD_DEFAULT = "B3LYP-D3BJ". Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
BASIS_DEFAULT = "def2-SVP"
```

CHARGE_DEFAULT = 0

Subbab ini menguraikan CHARGE_DEFAULT = 0. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

SPIN_DEFAULT = 0

Subbab ini menguraikan SPIN_DEFAULT = 0. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

DFT_GRID_LEVEL_DEFAULT = 3

Subbab ini menguraikan DFT_GRID_LEVEL_DEFAULT = 3. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
CONV_TOL_DEFAULT = 1e-9
```

MAX_CYCLE_DEFAULT = 150

Subbab ini menguraikan MAX_CYCLE_DEFAULT = 150. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

VERBOSE_DEFAULT = 4

Subbab ini menguraikan VERBOSE_DEFAULT = 4. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
USE_DENSITY_FITTING_DEFAULT = False
JOB_DEFAULT = "singlepoint"
```

```
OUTPUT_ROOT_DEFAULT = "hasil_pyscf"
```

BOND_SCALE_DEFAULT = 1.25

Subbab ini menguraikan BOND_SCALE_DEFAULT = 1.25. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

BOND_TOLERANCE_DEFAULT = 0.15
Subbab ini menguraikan BOND_TOLERANCE_DEFAULT = 0.15. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
AUTHOR_DEFAULT = "Kasmui"
```

AUTHOR_PHONE_DEFAULT = "0818294312"
Subbab ini menguraikan AUTHOR_PHONE_DEFAULT = "0818294312". Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
SMILES_DEFAULT = "CC(C)(C#N)c1cc(C(n2ncnc2)c2ccc(C#N)cc2)cc(C(C)(C)C#N)c1"
IUPAC_NAME_DEFAULT = "2,2'-[5-[(4-cyanophenyl)(1H-1,2,4-triazol-1-yl)methyl]-1,3-
phenylene]bis(2-methylpropanenitrile)"
```

RESEARCH_NOTE_DEFAULT = (
"Kandidat hibrida in silico Letrozole-Anastrozole untuk riset inhibitor aromatase/CYP19A1. "

"Struktur ini adalah kandidat awal komputasi, bukan obat tervalidasi klinis."

)

```
# ============================================================
```

4. KOORDINAT DEFAULT LAH-01 LETROZOLE-ANASTROZOLE HYBRID
Subbab ini menguraikan # 4. KOORDINAT DEFAULT LAH-01 LETROZOLE-ANASTROZOLE HYBRID. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
==
Format XYZ standar juga diterima:
baris 1 = jumlah atom, baris 2 = judul/komentar.
Program otomatis membuang dua baris tersebut saat membaca koordinat.

DEFAULT_MOLECULE_XYZ = """
52
LAH-01 Letrozole-Anastrozole Hybrid | author: Kasmui | Telp.: 0818294312 | generated from RDKit MMFF94 conformer
C -1.16285703 3.56822713 -0.22330159
Subbab ini menguraikan C -1.16285703 3.56822713 -0.22330159. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C -2.10886641 2.42635441 0.21010982
Subbab ini menguraikan C -2.10886641 2.42635441 0.21010982. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C -2.42522992 2.61860541 1.70817752
Subbab ini menguraikan C -2.42522992 2.61860541 1.70817752. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C -3.36730827 2.58805761 -0.57001762
Subbab ini menguraikan C -3.36730827 2.58805761 -0.57001762. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

N -4.36170938 2.70832474 -1.15811404
Subbab ini menguraikan N -4.36170938 2.70832474 -1.15811404. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C -1.52061565 1.01310072 0.01359150
Subbab ini menguraikan C -1.52061565 1.01310072 0.01359150. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C -0.16177500 0.79711191 -0.26307620
Subbab ini menguraikan C -0.16177500 0.79711191 -0.26307620. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 0.36083566 -0.49433572 -0.44446414
Subbab ini menguraikan C 0.36083566 -0.49433572 -0.44446414. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 1.84415274 -0.72427333 -0.73431941
Subbab ini menguraikan C 1.84415274 -0.72427333 -0.73431941. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
N 2.21687154 -0.34619052 -2.10776252
Subbab ini menguraikan N 2.21687154 -0.34619052 -2.10776252. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
N 1.88420491 0.84024878 -2.67261278
Subbab ini menguraikan N 1.88420491 0.84024878 -2.67261278. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 2.41256212 0.71166107 -3.89214986
Subbab ini menguraikan C 2.41256212 0.71166107 -3.89214986. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
N 3.07029776 -0.46129526 -4.12225021
Subbab ini menguraikan N 3.07029776 -0.46129526 -4.12225021. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 2.93529928 -1.10493537 -2.98666047
Subbab ini menguraikan C 2.93529928 -1.10493537 -2.98666047. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 2.77829088 -0.15791737 0.33889448
Subbab ini menguraikan C 2.77829088 -0.15791737 0.33889448. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 2.94139876 -0.87761295 1.53890093
Subbab ini menguraikan C 2.94139876 -0.87761295 1.53890093. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 3.76447128 -0.39683201 2.56186082

Subbab ini menguraikan C 3.76447128 -0.39683201 2.56186082. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 4.43607054 0.81766175 2.40875175
Subbab ini menguraikan C 4.43607054 0.81766175 2.40875175. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 5.27804309 1.31422662 3.45733495
Subbab ini menguraikan C 5.27804309 1.31422662 3.45733495. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
N 5.95802625 1.71648576 4.30739764
Subbab ini menguraikan N 5.95802625 1.71648576 4.30739764. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 4.27611227 1.54895333 1.22977636
Subbab ini menguraikan C 4.27611227 1.54895333 1.22977636. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 3.45151123 1.06822825 0.20670028
Subbab ini menguraikan C 3.45151123 1.06822825 0.20670028. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C -0.50965902 -1.59189117 -0.34206007
Subbab ini menguraikan C -0.50965902 -1.59189117 -0.34206007. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C -1.87658900 -1.43343693 -0.03884708
Subbab ini menguraikan C -1.87658900 -1.43343693 -0.03884708. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C -2.78921915 -2.67347886 0.06518197
Subbab ini menguraikan C -2.78921915 -2.67347886 0.06518197. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C -2.99157426 -3.25446587 -1.34972668
Subbab ini menguraikan C -2.99157426 -3.25446587 -1.34972668. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C -4.18388871 -2.38246002 0.66108956
Subbab ini menguraikan C -4.18388871 -2.38246002 0.66108956. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C -2.15751444 -3.71224190 0.92580880
Subbab ini menguraikan C -2.15751444 -3.71224190 0.92580880. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
N -1.64482863 -4.52252547 1.58135224
Subbab ini menguraikan N -1.64482863 -4.52252547 1.58135224. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -2.35484059 -0.12214995 0.13346884
Subbab ini menguraikan C -2.35484059 -0.12214995 0.13346884. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H -1.64830649 4.54708731 -0.11998457
Subbab ini menguraikan H -1.64830649 4.54708731 -0.11998457. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H -0.86569105 3.46746507 -1.27451724
Subbab ini menguraikan H -0.86569105 3.46746507 -1.27451724. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H -0.25269457 3.59834878 0.38676234
Subbab ini menguraikan H -0.25269457 3.59834878 0.38676234. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H -2.83771040 3.61604361 1.90373693
Subbab ini menguraikan H -2.83771040 3.61604361 1.90373693. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H -3.16374377 1.89237691 2.06909984
Subbab ini menguraikan H -3.16374377 1.89237691 2.06909984. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H -1.52393428 2.50343340 2.32215728
Subbab ini menguraikan H -1.52393428 2.50343340 2.32215728. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 0.52156415 1.63862681 -0.35861219
Subbab ini menguraikan H 0.52156415 1.63862681 -0.35861219. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 2.01387757 -1.81191010 -0.72187977
Subbab ini menguraikan H 2.01387757 -1.81191010 -0.72187977. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 2.31970608 1.49530682 -4.63352787
Subbab ini menguraikan H 2.31970608 1.49530682 -4.63352787. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 3.32315327 -2.08732752 -2.75533808
Subbab ini menguraikan H 3.32315327 -2.08732752 -2.75533808. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 2.41908504 -1.82191848 1.69219282
Subbab ini menguraikan H 2.41908504 -1.82191848 1.69219282. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 3.87167098 -0.97532108 3.47823944

Subbab ini menguraikan H 3.87167098 -0.97532108 3.47823944. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 4.78809360 2.50100836 1.09705711
Subbab ini menguraikan H 4.78809360 2.50100836 1.09705711. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 3.34385306 1.67665471 -0.69033051
Subbab ini menguraikan H 3.34385306 1.67665471 -0.69033051. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H -0.10930452 -2.59494931 -0.49412927
Subbab ini menguraikan H -0.10930452 -2.59494931 -0.49412927. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H -3.64380714 -4.13617990 -1.32896814
Subbab ini menguraikan H -3.64380714 -4.13617990 -1.32896814. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H -2.04473039 -3.56995205 -1.80464799
Subbab ini menguraikan H -2.04473039 -3.56995205 -1.80464799. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H -3.44832555 -2.51645779 -2.01997791
Subbab ini menguraikan H -3.44832555 -2.51645779 -2.01997791. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H -4.76977016 -3.30415567 0.76927345
Subbab ini menguraikan H -4.76977016 -3.30415567 0.76927345. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H -4.76681698 -1.70711806 0.02427887
Subbab ini menguraikan H -4.76681698 -1.70711806 0.02427887. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H -4.10957946 -1.93157114 1.65839348
Subbab ini menguraikan H -4.10957946 -1.93157114 1.65839348. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H -3.40826183 0.03930455 0.35768718
"""

```
# ============================================================
```

5. DATA ATOM DAN JARI-JARI KOVALEN
Subbab ini menguraikan # 5. DATA ATOM DAN JARI-JARI KOVALEN. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

COVALENT_RADII_ANGSTROM = {
    "H": 0.31,
```

```
    "He": 0.28,
    "Li": 1.28,
    "Be": 0.96,
    "B": 0.84,
    "C": 0.76,
    "N": 0.71,
    "O": 0.66,
    "F": 0.57,
    "Ne": 0.58,
    "Na": 1.66,
    "Mg": 1.41,
    "Al": 1.21,
    "Si": 1.11,
    "P": 1.07,
    "S": 1.05,
    "Cl": 1.02,
    "Ar": 1.06,
    "K": 2.03,
    "Ca": 1.76,
    "Fe": 1.24,
    "Cu": 1.32,
    "Zn": 1.22,
    "Br": 1.20,
    "I": 1.39,
}

ATOMIC_NUMBERS = {
    "H": 1,
    "He": 2,
    "Li": 3,
    "Be": 4,
    "B": 5,
    "C": 6,
    "N": 7,
    "O": 8,
    "F": 9,
    "Ne": 10,
    "Na": 11,
```

```
    "Mg": 12,
    "Al": 13,
    "Si": 14,
    "P": 15,
    "S": 16,
    "Cl": 17,
    "Ar": 18,
    "K": 19,
    "Ca": 20,
    "Fe": 26,
    "Cu": 29,
    "Zn": 30,
    "Br": 35,
    "I": 53,
}

# ============================================================
```

6. FUNGSI BANTUAN UMUM

Subbab ini menguraikan # 6. FUNGSI BANTUAN UMUM. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def clean_element_symbol(symbol):
    letters = "".join(ch for ch in str(symbol).strip() if ch.isalpha())
    if not letters:
        return ""
    if len(letters) == 1:
        return letters.upper()
    return letters[0].upper() + letters[1:].lower()

def safe_filename(text):
    safe_chars = []
    for char in str(text):
        if char.isalnum() or char in ["-", "_"]:
            safe_chars.append(char)
        elif char in ["*", "+"]:
            safe_chars.append("x")
        else:
            safe_chars.append("_")
    value = "".join(safe_chars).strip("_")
    while "__" in value:
        value = value.replace("__", "_")
    return value or "molecule"


def normalize_choice(user_value, allowed_dict, label):
```

```
    if user_value is None:
        return None
    user_compact = str(user_value).strip().lower().replace(" ", "")
    for key in allowed_dict:
        key_compact = key.strip().lower().replace(" ", "")
        if user_compact == key_compact:
            return key
    print(f"\nERROR: {label} '{user_value}' tidak tersedia.")
    print(f"Pilihan {label} yang tersedia:")
    for key in allowed_dict:
        print(f"  - {key}")
    sys.exit(1)

def validate_positive_float(value, label):
    if value <= 0:
        print(f"ERROR: {label} harus bernilai positif.")
        sys.exit(1)

def validate_nonnegative_int(value, label):
    if value < 0:
        print(f"ERROR: {label} harus integer >= 0.")
        sys.exit(1)

def validate_positive_int(value, label):
    if value <= 0:
        print(f"ERROR: {label} harus integer positif.")
        sys.exit(1)

class TeeOutput:
    """
    Menyalin semua teks yang ditulis ke terminal sekaligus ke file log.

    Tujuan:
    - Teks proses running tetap tampil normal di layar.
    - Teks yang sama otomatis tersimpan ke file .log.
    - stdout dan stderr ikut ditangkap, termasuk pesan error PySCF.
    """

    def __init__(self, terminal_stream, log_stream):
        self.terminal_stream = terminal_stream
        self.log_stream = log_stream

    def write(self, message):
        self.terminal_stream.write(message)
        self.log_stream.write(message)
        self.flush()

    def flush(self):
        self.terminal_stream.flush()
        self.log_stream.flush()
```

```
    def isatty(self):
        return getattr(self.terminal_stream, "isatty", lambda: False)()

    def fileno(self):
        return self.terminal_stream.fileno()

def write_running_log_header(log_path, args, method_name, basis_name, pyscf_basis):
    print(identity_header_text())
    print("\n============================================================")
    print("LOG RUNNING OTOMATIS AKTIF")
    print("============================================================")
    print(f"File log running  : {log_path}")
    print(f"Waktu mulai       : {datetime.now().strftime('%Y-%m-%d %H:%M:%S')}")
    print(f"Nama molekul      : {args.name}")
    print(f"Rumus molekul     : {args.formula}")
    print(f"Metode            : {method_name}")
    print(f"Basis set         : {basis_name} ({pyscf_basis})")
    print(f"Jenis perhitungan : {args.job}")
    print("Catatan           : Semua teks terminal stdout/stderr disalin ke file ini.")
    print("============================================================")

def write_running_log_footer(log_path, status="SELESAI"):
    print("\n============================================================")
    print("LOG RUNNING OTOMATIS DITUTUP")
    print("============================================================")
    print(f"Status akhir      : {status}")
    print(f"Waktu selesai     : {datetime.now().strftime('%Y-%m-%d %H:%M:%S')}")
    print(f"File log running  : {log_path}")
    print("============================================================")

# ============================================================
```

7. PEMBACAAN DAN VALIDASI KOORDINAT

Subbab ini menguraikan # 7. PEMBACAAN DAN VALIDASI KOORDINAT. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def parse_xyz_like_text(text):
    """
```

Membaca koordinat dari:

1. XYZ standar:
jumlah_atom

komentar

Atom x y z

2. Blok koordinat langsung:
Atom x y z

3. File teks dengan komentar diawali #.

```
    """

    if not text or not str(text).strip():
        raise ValueError("Blok koordinat kosong.")
```

raw_lines = []

```
    for line in str(text).splitlines():
```

stripped = line.strip()

```
        if not stripped:
```

```
            continue
        if stripped.startswith("#"):
            continue
        raw_lines.append(stripped)

    if not raw_lines:
        raise ValueError("Tidak ada baris koordinat yang dapat dibaca.")

    expected_count = None
    coordinate_lines = raw_lines

    first_parts = raw_lines[0].split()
    if len(first_parts) == 1:
        try:
            expected_count = int(first_parts[0])
            coordinate_lines = raw_lines[2:2 + expected_count]
        except ValueError:
            expected_count = None

    atoms = []
    for number, line in enumerate(coordinate_lines, start=1):
        parts = line.split()
        if len(parts) < 4:
            raise ValueError(
                f"Format koordinat salah pada baris koordinat ke-{number}: '{line}'. "
                "Format benar: SimbolAtom X Y Z"
            )

        symbol = clean_element_symbol(parts[0])
        if symbol not in ATOMIC_NUMBERS:
            raise ValueError(
                f"Simbol atom '{parts[0]}' pada baris ke-{number} tidak dikenali."
            )

        try:
            x = float(parts[1])
            y = float(parts[2])
            z = float(parts[3])
        except ValueError as exc:
            raise ValueError(
                f"Koordinat X/Y/Z harus numerik pada baris ke-{number}: '{line}'."
            ) from exc

        atoms.append({
            "symbol": symbol,
            "x": x,
```

```
            "y": y,
            "z": z,
        })

    if expected_count is not None and len(atoms) != expected_count:
        raise ValueError(
            f"Jumlah atom di header XYZ = {expected_count}, tetapi yang terbaca = {len(atoms)}."
        )

    if not atoms:
        raise ValueError("Tidak ada atom yang berhasil dibaca.")

    return atoms

def read_text_file(path):
    file_path = Path(path)
    if not file_path.exists():
        print(f"ERROR: File koordinat tidak ditemukan: {file_path}")
        sys.exit(1)
    return file_path.read_text(encoding="utf-8")

def format_atom_block(atoms):
    return "\n".join(
        f"{atom['symbol']:2s}  {atom['x']:14.8f}  {atom['y']:14.8f}  {atom['z']:14.8f}"
        for atom in atoms
    )

def get_coordinate_atoms(args):
    if args.xyz_file:
        source_text = read_text_file(args.xyz_file)
        source_label = f"file XYZ: {args.xyz_file}"
    elif args.coord_file:
        source_text = read_text_file(args.coord_file)
        source_label = f"file koordinat: {args.coord_file}"
    else:
        source_text = DEFAULT_MOLECULE_XYZ
        source_label = "koordinat default di dalam kode"

    try:
        atoms = parse_xyz_like_text(source_text)
    except ValueError as exc:
        print("\nERROR: Koordinat molekul tidak valid.")
        print(str(exc))
        print("\nContoh format benar:")
        print("C   0.000000   0.000000   0.000000")
        print("H   0.000000   0.000000   1.089000")
        sys.exit(1)

    return atoms, source_label

# ============================================================
# 8. RESOLUSI METODE DAN BASIS SET
```

Subbab ini menguraikan # 8. RESOLUSI METODE DAN BASIS SET. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def resolve_method(args):
    method_name = normalize_choice(args.method, AVAILABLE_METHODS, "metode")
    info = dict(AVAILABLE_METHODS[method_name])

    if method_name == "CUSTOM":
        if not args.xc or not args.xc.strip():
            print("ERROR: --method CUSTOM wajib disertai --xc, contoh: --xc 'm06-2x'")
            sys.exit(1)
        info["xc"] = args.xc.strip()
        info["description"] = f"DFT custom dengan functional PySCF: {args.xc.strip()}"

    if args.disp:
        info["disp"] = args.disp.strip().lower()

    return method_name, info

def resolve_basis(args):
    basis_name = normalize_choice(args.basis, AVAILABLE_BASIS_SETS, "basis set")

    if basis_name == "CUSTOM":
        if not args.basis_custom or not args.basis_custom.strip():
            print("ERROR: --basis CUSTOM wajib disertai --basis-custom, contoh: --basis-
custom 'def2-tzvpp'")
            sys.exit(1)
        pyscf_basis = args.basis_custom.strip()
    else:
        pyscf_basis = AVAILABLE_BASIS_SETS[basis_name]

    return basis_name, pyscf_basis


# ============================================================
```

9. MEMBANGUN OBJEK MOLEKUL DAN KALKULATOR

Subbab ini menguraikan # 9. MEMBANGUN OBJEK MOLEKUL DAN KALKULATOR. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def build_molecule(atom_block, pyscf_basis, charge, spin, verbose):
    mol = gto.M(
        atom=atom_block,
        basis=pyscf_basis,
        unit="Angstrom",
```

```
        charge=charge,
        spin=spin,
        verbose=verbose,
    )
    return mol


def build_calculator(mol, method_name, method_info, grid_level, conv_tol, max_cycle,
use_density_fitting):
    family = method_info["family"]

    if family == "HF":
        mf = scf.RHF(mol) if mol.spin == 0 else scf.UHF(mol)
    elif family == "DFT":
        mf = dft.RKS(mol) if mol.spin == 0 else dft.UKS(mol)
        mf.xc = method_info["xc"]
        mf.grids.level = grid_level
    else:
        raise ValueError(f"Family metode tidak dikenal: {family}")

    if method_info.get("disp"):
        # PySCF modern mendukung atribut mf.disp untuk D3/D4 jika ekstensi tersedia.
        mf.disp = method_info["disp"]

    if use_density_fitting:
        mf = mf.density_fit()

    mf.conv_tol = conv_tol
    mf.max_cycle = max_cycle
    mf.chkfile = safe_filename(f"{MOLECULE_NAME}_{method_name}_{BASIS_LABEL}.chk")

    return mf
# ============================================================
```

10. OPTIMASI GEOMETRI OPSIONAL

Subbab ini menguraikan # 10. OPTIMASI GEOMETRI OPSIONAL. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def optimize_geometry(mf):
    try:
        from pyscf.geomopt.geometric_solver import optimize
    except ImportError:
        print("\nERROR: Optimasi geometri memerlukan paket geomeTRIC.")
        print("Instal terlebih dahulu:")
        print("  pip install geometric")
        sys.exit(1)

    print("\n============================================================")
    print("OPTIMASI GEOMETRI DIMULAI")
    print("============================================================")
```

mol_opt = optimize(mf)

```
    print("\n============================================================")
    print("OPTIMASI GEOMETRI SELESAI")
    print("============================================================")
    return mol_opt

# ============================================================
```

11. INFORMASI DAN PELAPORAN DASAR

Subbab ini menguraikan # 11. INFORMASI DAN PELAPORAN DASAR. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def print_available_options():
    print("\n============================================================")
    print("DAFTAR METODE KOMPUTASI")
    print("============================================================")
    for method_name, info in AVAILABLE_METHODS.items():
```

xc_text = info["xc"] if info["xc"] else "-"

disp_text = info["disp"] if info["disp"] else "-"

```
        print(f"{method_name:12s} | family={info['family']:3s} | xc={xc_text:12s} |
disp={disp_text:5s} | {info['description']}")

    print("\n============================================================")
    print("DAFTAR BASIS SET")
    print("============================================================")
    for basis_name, pyscf_name in AVAILABLE_BASIS_SETS.items():
        print(f"{basis_name:12s} : {pyscf_name if pyscf_name else 'isi melalui --basis-
custom'}")

    print("\n============================================================")
    print("CONTOH PERINTAH")
    print("============================================================")
    print("python lah01-letrozole-anastrozole-hybrid.py")
    print("python lah01-letrozole-anastrozole-hybrid.py --method B3LYP --basis def2-SVP")
    print("python lah01-letrozole-anastrozole-hybrid.py --xyz-file molekul.xyz --name
'Molekul Baru' --method PBE0 --basis def2-TZVP")
    print("python lah01-letrozole-anastrozole-hybrid.py --method CUSTOM --xc 'm06-2x' --
basis CUSTOM --basis-custom 'def2-tzvpp'")
    print("python lah01-letrozole-anastrozole-hybrid.py --job optimize --method B3LYP --
basis def2-SVP")

    print("\n============================================================")
    print("CONTOH FORMAT KOORDINAT")
    print("============================================================")
    print("3")
    print("Water")
    print("O   0.000000   0.000000   0.000000")
    print("H   0.000000  -0.757000   0.587000")
    print("H   0.000000   0.757000   0.587000")
    print()

def print_molecule_info(mol, method_name, method_info, basis_name, pyscf_basis, charge,
spin, job, coordinate_source):
    print("\n============================================================")
    print("INFORMASI SISTEM")
    print("============================================================")
    print(f"Nama molekul        : {MOLECULE_NAME}")
    print(f"Rumus molekul       : {MOLECULE_FORMULA}")
    print(f"Sumber koordinat    : {coordinate_source}")
    print(f"Jumlah atom         : {mol.natm}")
    print(f"Jumlah elektron     : {mol.nelectron}")
    print(f"Jumlah fungsi basis : {mol.nao_nr()}")
    print(f"Muatan total        : {charge}")
    print(f"Spin PySCF          : {spin}")
```

```
    print(f"Metode komputasi    : {method_name}")
    print(f"Family metode       : {method_info['family']}")
    print(f"XC functional       : {method_info.get('xc')}")
    print(f"Dispersion          : {method_info.get('disp')}")
    print(f"Basis set           : {basis_name} ({pyscf_basis})")
    print(f"Jenis perhitungan   : {job}")
    print("Satuan koordinat    : Angstrom")
    print("============================================================")

def print_coordinates(mol, title):
    print("\n============================================================")
    print(title)
    print("============================================================")
```

coords = mol.atom_coords(unit="Angstrom")

```
    for i in range(mol.natm):
```

symbol = clean_element_symbol(mol.atom_symbol(i))

x, y, z = coords[i]

```
        print(f"{symbol:2s}  {x:14.8f}  {y:14.8f}  {z:14.8f}")

# ============================================================
```

12. ANALISIS STRUKTUR AKHIR

Subbab ini menguraikan # 12. ANALISIS STRUKTUR AKHIR. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def get_atom_records(mol):
```

coords = mol.atom_coords(unit="Angstrom")
atoms = []

```
    for i in range(mol.natm):
```

symbol = clean_element_symbol(mol.atom_symbol(i))
atomic_number = ATOMIC_NUMBERS.get(symbol)

```
        try:
```

atomic_number = int(mol.atom_charge(i))

```
        except Exception:
```

pass
x, y, z = coords[i]
atoms.append({
"index": i + 1,
"symbol": symbol,
"atomic_number": atomic_number,
"x": float(x),
"y": float(y),
"z": float(z),
})

```
    return atoms


def distance_between_atoms(atom_a, atom_b):
```

dx = atom_a["x"] - atom_b["x"]
dy = atom_a["y"] - atom_b["y"]
dz = atom_a["z"] - atom_b["z"]

```
    return math.sqrt(dx * dx + dy * dy + dz * dz)


def vector_between_atoms(atom_a, atom_b):
    return [
```

```
        atom_b["x"] - atom_a["x"],
        atom_b["y"] - atom_a["y"],
        atom_b["z"] - atom_a["z"],
    ]


def dot_product(vec_a, vec_b):
    return vec_a[0] * vec_b[0] + vec_a[1] * vec_b[1] + vec_a[2] * vec_b[2]


def cross_product(vec_a, vec_b):
    return [
        vec_a[1] * vec_b[2] - vec_a[2] * vec_b[1],
        vec_a[2] * vec_b[0] - vec_a[0] * vec_b[2],
        vec_a[0] * vec_b[1] - vec_a[1] * vec_b[0],
    ]


def vector_norm(vec):
    return math.sqrt(dot_product(vec, vec))


def normalize_vector(vec):
    norm = vector_norm(vec)
    if norm < 1e-15:
        return [0.0, 0.0, 0.0]
    return [value / norm for value in vec]


def calculate_angle_degrees(atom_i, atom_j, atom_k):
    vec_ji = vector_between_atoms(atom_j, atom_i)
    vec_jk = vector_between_atoms(atom_j, atom_k)
    norm_ji = vector_norm(vec_ji)
    norm_jk = vector_norm(vec_jk)
    if norm_ji < 1e-15 or norm_jk < 1e-15:
        return None
    cos_theta = dot_product(vec_ji, vec_jk) / (norm_ji * norm_jk)
    cos_theta = max(-1.0, min(1.0, cos_theta))
    return math.degrees(math.acos(cos_theta))


def calculate_dihedral_degrees(atom_i, atom_j, atom_k, atom_l):
    p0 = [atom_i["x"], atom_i["y"], atom_i["z"]]
    p1 = [atom_j["x"], atom_j["y"], atom_j["z"]]
    p2 = [atom_k["x"], atom_k["y"], atom_k["z"]]
    p3 = [atom_l["x"], atom_l["y"], atom_l["z"]]

    b0 = [p0[n] - p1[n] for n in range(3)]
    b1 = [p2[n] - p1[n] for n in range(3)]
    b2 = [p3[n] - p2[n] for n in range(3)]
    b1_unit = normalize_vector(b1)

    v = [b0[n] - dot_product(b0, b1_unit) * b1_unit[n] for n in range(3)]
    w = [b2[n] - dot_product(b2, b1_unit) * b1_unit[n] for n in range(3)]

    if vector_norm(v) < 1e-15 or vector_norm(w) < 1e-15:
        return None
```

```
    x_value = dot_product(v, w)
    y_value = dot_product(cross_product(b1_unit, v), w)
    return math.degrees(math.atan2(y_value, x_value))


def infer_bonds(atoms, bond_scale=BOND_SCALE_DEFAULT,
bond_tolerance=BOND_TOLERANCE_DEFAULT):
    bonds = []
    for i in range(len(atoms)):
        for j in range(i + 1, len(atoms)):
            atom_i = atoms[i]
            atom_j = atoms[j]
            symbol_i = atom_i["symbol"]
            symbol_j = atom_j["symbol"]
            radius_i = COVALENT_RADII_ANGSTROM.get(symbol_i, 0.77)
            radius_j = COVALENT_RADII_ANGSTROM.get(symbol_j, 0.77)
            distance = distance_between_atoms(atom_i, atom_j)
            threshold = bond_scale * (radius_i + radius_j) + bond_tolerance
            if symbol_i == "H" and symbol_j == "H" and distance > 0.90:
                continue
            if distance <= threshold:
                bonds.append({
                    "atom1": atom_i["index"],
                    "atom2": atom_j["index"],
                    "symbol1": symbol_i,
                    "symbol2": symbol_j,
                    "distance": float(distance),
                    "order": 1,
                })
    return bonds


def build_adjacency(atoms, bonds):
    adjacency = {atom["index"]: [] for atom in atoms}
    for bond in bonds:
        atom1 = bond["atom1"]
        atom2 = bond["atom2"]
        adjacency[atom1].append(atom2)
        adjacency[atom2].append(atom1)
    for atom_index in adjacency:
        adjacency[atom_index] = sorted(adjacency[atom_index])
    return adjacency


def infer_angles(atoms, bonds):
    atom_by_index = {atom["index"]: atom for atom in atoms}
    adjacency = build_adjacency(atoms, bonds)
    angles = []
    for center in sorted(adjacency):
        neighbors = adjacency[center]
        for a in range(len(neighbors)):
            for b in range(a + 1, len(neighbors)):
                atom_i_index = neighbors[a]
                atom_k_index = neighbors[b]
                angle_value = calculate_angle_degrees(
                    atom_by_index[atom_i_index],
                    atom_by_index[center],
```

```
                atom_by_index[atom_k_index],
            )
                if angle_value is None:
                continue
            angles.append({
                "atom1": atom_i_index,
                "atom2": center,
                "atom3": atom_k_index,
                "symbol1": atom_by_index[atom_i_index]["symbol"],
                "symbol2": atom_by_index[center]["symbol"],
                "symbol3": atom_by_index[atom_k_index]["symbol"],
                "angle_degrees": float(angle_value),
            })
    return angles


def infer_dihedrals(atoms, bonds):
    atom_by_index = {atom["index"]: atom for atom in atoms}
    adjacency = build_adjacency(atoms, bonds)
    seen = set()
    dihedrals = []
    for bond in bonds:
        j = bond["atom1"]
        k = bond["atom2"]
        for i in adjacency[j]:
            if i == k:
                continue
            for l in adjacency[k]:
                if l == j:
                continue
            dihedral_tuple = (i, j, k, l)
            reverse_tuple = tuple(reversed(dihedral_tuple))
            canonical = min(dihedral_tuple, reverse_tuple)
                if canonical in seen:
                continue
            seen.add(canonical)
            dihedral_value = calculate_dihedral_degrees(
                atom_by_index[i],
                atom_by_index[j],
                atom_by_index[k],
                atom_by_index[l],
            )
                if dihedral_value is None:
                continue
            dihedrals.append({
                "atom1": i,
                "atom2": j,
                "atom3": k,
                "atom4": l,
                "symbol1": atom_by_index[i]["symbol"],
                "symbol2": atom_by_index[j]["symbol"],
                "symbol3": atom_by_index[k]["symbol"],
                "symbol4": atom_by_index[l]["symbol"],
                "dihedral_degrees": float(dihedral_value),
            })
    return dihedrals
```

```
def calculate_distance_matrix(atoms):
    return [
        [float(distance_between_atoms(atom_i, atom_j)) for atom_j in atoms]
        for atom_i in atoms
    ]


# ============================================================
```

13. PENULISAN FILE HASIL

Subbab ini menguraikan # 13. PENULISAN FILE HASIL. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def make_output_dir(args, method_name, basis_name):
    timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
    folder_name = safe_filename(
        f"{MOLECULE_NAME}_{MOLECULE_FORMULA}_{method_name}_{basis_name}_{args.job}_{timestamp}"
    )
    if args.output_dir:
        output_path = Path(args.output_dir)
    else:
        output_path = Path(OUTPUT_ROOT_DEFAULT) / folder_name
    output_path.mkdir(parents=True, exist_ok=True)
    return output_path

def make_output_prefix(method_name, basis_name, job):
    timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
    return safe_filename(f"{MOLECULE_NAME}_{MOLECULE_FORMULA}_{method_name}_{basis_name}_{job}_{timestamp}")

def format_xyz_coordinate_block(atoms):
    return "\n".join(
        f"{atom['symbol']:2s}  {atom['x']:14.8f}  {atom['y']:14.8f}  {atom['z']:14.8f}"
        for atom in atoms
    )

def identity_header_lines():
    return [
        f"author: {AUTHOR_DEFAULT}",
        f"Telp.: {AUTHOR_PHONE_DEFAULT}",
    ]

def identity_header_text():
```

```
    return "\n".join(identity_header_lines())

def write_txt_coordinates(path, atoms, metadata):
    lines = identity_header_lines() + [""]
    lines.append("KOORDINAT AKHIR MOLEKUL")
    lines.append("=" * 60)
    for key, value in metadata.items():
        lines.append(f"{key}: {value}")
    lines.append("=" * 60)
    lines.append("Format: SimbolAtom        X              Y              Z")
    lines.append("Satuan: Angstrom")
    lines.append("")
    lines.append(format_xyz_coordinate_block(atoms))
    lines.append("")
    lines.append("BLOK KOORDINAT SIAP SALIN UNTUK PySCF/XYZ")
    lines.append("=" * 60)
    lines.append('\"\"\"')
    lines.append(format_xyz_coordinate_block(atoms))
    lines.append('\"\"\"')
    lines.append("")
    Path(path).write_text("\n".join(lines), encoding="utf-8")

def write_xyz_file(path, atoms, metadata):
    title = (
        f"{metadata['Nama molekul']} | author: {AUTHOR_DEFAULT} | Telp.: {AUTHOR_PHONE_DEFAULT} | "
        f"{metadata['Metode']} / {metadata['Basis set']} | "
        f"E = {metadata['Energi total Hartree']} Hartree | "
        f"SCF converged = {metadata['SCF konvergen']}"
    )
    lines = [str(len(atoms)), title, format_xyz_coordinate_block(atoms)]
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_mol_file(path, atoms, bonds, title=None):
    title = title or MOLECULE_NAME
    lines = []
    lines.append(title[:80])
    lines.append(f"  author: {AUTHOR_DEFAULT} | Telp.: {AUTHOR_PHONE_DEFAULT}")
    lines.append("  Generated final 3D coordinates and inferred bonds by PySCF Python Script")
    lines.append(f"{len(atoms):3d}{len(bonds):3d}  0  0  0  0            999 V2000")

    for atom in atoms:
        lines.append(
            f"{atom['x']:10.4f}"
            f"{atom['y']:10.4f}"
            f"{atom['z']:10.4f} "
            f"{atom['symbol']:<3s}"
            " 0  0  0  0  0  0  0  0  0  0  0  0"
        )
```

```
    for bond in bonds:
        lines.append(
        f"{bond['atom1']:3d}"
        f"{bond['atom2']:3d}"
        f"{int(bond.get('order', 1)):3d}"
        "  0  0  0  0"
    )

    lines.append("M  END")
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")

def write_sdf_file(path, atoms, bonds, metadata):
    mol_text_path = Path(path).with_suffix(".tmp_mol")
    write_mol_file(mol_text_path, atoms, bonds, title=metadata["Nama molekul"])
    mol_text = mol_text_path.read_text(encoding="utf-8")
    mol_text_path.unlink(missing_ok=True)

    lines = [mol_text.rstrip("\n")]
    for key, value in metadata.items():
        lines.append(f">  <{key}>")
        lines.append(str(value))
        lines.append("")
    lines.append("$$$$")
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")

def write_pdb_file(path, atoms, bonds, metadata):
    lines = []
    lines.append(f"HEADER    {metadata['Nama molekul'][:50]}")
    lines.append(f"REMARK    author: {AUTHOR_DEFAULT}")
    lines.append(f"REMARK    Telp.: {AUTHOR_PHONE_DEFAULT}")
    lines.append(f"TITLE     {metadata['Metode']} / {metadata['Basis set']}")
    lines.append(f"REMARK    Energy Hartree: {metadata['Energi total Hartree']}")
    lines.append(f"REMARK    SCF converged: {metadata['SCF konvergen']}")

    for atom in atoms:
        atom_index = atom["index"]
        symbol = atom["symbol"]
        lines.append(
        f"HETATM{atom_index:5d} "
        f"{symbol:<4s}"
        f" MOL     1    "
        f"{atom['x']:8.3f}"
        f"{atom['y']:8.3f}"
        f"{atom['z']:8.3f}"
        f"  1.00  0.00          "
```

```
        f"{symbol:>2s}"
    )

adjacency = build_adjacency(atoms, bonds)
    for atom_index in sorted(adjacency):
        if not adjacency[atom_index]:
            continue
        conect_targets = "".join(f"{target:5d}" for target in adjacency[atom_index])
        lines.append(f"CONECT{atom_index:5d}{conect_targets}")

    lines.append("END")
Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_cml_file(path, atoms, bonds, metadata):
    lines = []
    lines.append('<?xml version="1.0" encoding="UTF-8"?>')
    lines.append('<cml xmlns="http://www.xml-cml.org/schema">')
    lines.append(f'  <molecule id="{escape(safe_filename(metadata["Nama molekul"]))}">')
    lines.append("    <metadataList>")
    for key, value in metadata.items():
        lines.append(f'      <metadata name="{escape(str(key))}"
content="{escape(str(value))}"/>')
    lines.append("    </metadataList>")
    lines.append("    <atomArray>")
    for atom in atoms:
        lines.append(
            f'      <atom id="a{atom["index"]}" '
            f'elementType="{escape(atom["symbol"])}" '
            f'x3="{atom["x"]:.8f}" '
            f'y3="{atom["y"]:.8f}" '
            f'z3="{atom["z"]:.8f}"/>'
        )
    lines.append("    </atomArray>")
    lines.append("    <bondArray>")
    for bond in bonds:
        lines.append(
            f'      <bond atomRefs2="a{bond["atom1"]} a{bond["atom2"]}" '
            f'order="{int(bond.get("order", 1))}"/>'
        )
    lines.append("    </bondArray>")
    lines.append("  </molecule>")
    lines.append("</cml>")
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_json_structure_file(path, atoms, bonds, angles, dihedrals, distance_matrix,
metadata):
    structure_data = {
        "metadata": metadata,
```

```
        "atoms": atoms,
        "bonds": bonds,
        "angles": angles,
        "dihedrals": dihedrals,
        "distance_matrix_angstrom": distance_matrix,
        "notes": {
            "bond_inference": (
                "Ikatan diinferensi dari jarak antaratom dan jari-jari kovalen. "
                "Orde ikatan ditulis 1 untuk tujuan visualisasi struktur 3D."
            ),
            "indexing": "Semua indeks atom memakai sistem 1-based.",
        },
    }
    Path(path).write_text(
        json.dumps(structure_data, indent=2, ensure_ascii=False),
        encoding="utf-8",
    )

def write_log_file(path, atoms, bonds, angles, dihedrals, distance_matrix, metadata,
output_files):
    lines = identity_header_lines() + [""]
    lines.append("=" * 80)
    lines.append("LAPORAN AKHIR PERHITUNGAN KIMIA KOMPUTASI PySCF")
    lines.append("=" * 80)
    for key, value in metadata.items():
        lines.append(f"{key:28s}: {value}")

    lines.append("")
    lines.append("=" * 80)
    lines.append("KOORDINAT AKHIR MOLEKUL")
    lines.append("=" * 80)
    lines.append("No.  Atom            X             Y             Z       Satuan")
    lines.append("-" * 80)
    for atom in atoms:
        lines.append(
            f"{atom['index']:3d}  {atom['symbol']:2s}  "
            f"{atom['x']:14.8f}  {atom['y']:14.8f}  {atom['z']:14.8f}  Angstrom"
        )

    lines.append("")
    lines.append("=" * 80)
    lines.append("DAFTAR IKATAN HASIL INFERENSI")
    lines.append("=" * 80)
    lines.append("No.  Atom-1       Atom-2       Orde    Jarak (Angstrom)")
    lines.append("-" * 80)
    if bonds:
        for number, bond in enumerate(bonds, start=1):
            lines.append(
```

```
            f"{number:3d}  "
            f"{bond['atom1']:3d}-{bond['symbol1']:<2s}      "
            f"{bond['atom2']:3d}-{bond['symbol2']:<2s}      "
            f"{int(bond.get('order', 1)):3d}     "
            f"{bond['distance']:12.6f}"
        )
    else:
        lines.append("Tidak ada ikatan terdeteksi. Periksa --bond-scale atau --bond-
tolerance.")

    lines.append("")
    lines.append("=" * 80)
    lines.append("DAFTAR SUDUT IKATAN")
    lines.append("=" * 80)
    lines.append("No.  Atom-1       Atom-2/Pusat Atom-3       Sudut (derajat)")
    lines.append("-" * 80)
    if angles:
        for number, angle in enumerate(angles, start=1):
            lines.append(
            f"{number:3d}  "
            f"{angle['atom1']:3d}-{angle['symbol1']:<2s}      "
            f"{angle['atom2']:3d}-{angle['symbol2']:<2s}       "
            f"{angle['atom3']:3d}-{angle['symbol3']:<2s}      "
            f"{angle['angle_degrees']:12.6f}"
        )
    else:
        lines.append("Tidak ada sudut ikatan terdeteksi.")

    lines.append("")
    lines.append("=" * 80)
    lines.append("DAFTAR SUDUT DIHEDRAL")
    lines.append("=" * 80)
    lines.append("No.  Atom-1       Atom-2       Atom-3       Atom-4       Dihedral
(derajat)")
    lines.append("-" * 80)
    if dihedrals:
        for number, dihedral in enumerate(dihedrals, start=1):
            lines.append(
            f"{number:3d}  "
            f"{dihedral['atom1']:3d}-{dihedral['symbol1']:<2s}      "
            f"{dihedral['atom2']:3d}-{dihedral['symbol2']:<2s}      "
            f"{dihedral['atom3']:3d}-{dihedral['symbol3']:<2s}      "
            f"{dihedral['atom4']:3d}-{dihedral['symbol4']:<2s}      "
            f"{dihedral['dihedral_degrees']:12.6f}"
        )
    else:
        lines.append("Tidak ada sudut dihedral terdeteksi.")

    lines.append("")
    lines.append("=" * 80)
```

```
    lines.append("MATRIKS JARAK ANTARATOM (Angstrom)")
    lines.append("=" * 80)
```

header = "Atom ".ljust(8) + "".join(f"{atom['index']:>10d}" for atom in atoms)

```
    lines.append(header)
    lines.append("-" * max(80, len(header)))
    for atom, row in zip(atoms, distance_matrix):
```

row_text = f"{atom['index']:3d}-{atom['symbol']:<2s} ".ljust(8)

row_text += "".join(f"{value:10.4f}" for value in row)

```
        lines.append(row_text)

    lines.append("")
    lines.append("=" * 80)
    lines.append("FILE HASIL YANG DIBUAT")
    lines.append("=" * 80)
    for label, filename in output_files.items():
        lines.append(f"{label:18s}: {filename}")

    lines.append("")
    lines.append("CATATAN")
    lines.append("- Format .xyz hanya menyimpan koordinat, bukan konektivitas ikatan.")
    lines.append("- Format .mol, .sdf, .pdb, dan .cml lebih tepat untuk visualisasi
struktur 3D.")
    lines.append("- Ikatan diinferensi dari jarak antaratom dan jari-jari kovalen.")
    lines.append("- Jika konektivitas visual belum sesuai, ubah --bond-scale atau --bond-
tolerance.")
    lines.append("")
```

Path(path).write_text("\n".join(lines), encoding="utf-8")

```
def save_final_structure_outputs(mol, mf, energy, method_name, method_info, basis_name,
pyscf_basis, args, output_dir, running_output_log_path=None):
```

atoms = get_atom_records(mol)

bonds = infer_bonds(atoms, bond_scale=args.bond_scale, bond_tolerance=args.bond_tolerance)

angles = infer_angles(atoms, bonds)

dihedrals = infer_dihedrals(atoms, bonds)

distance_matrix = calculate_distance_matrix(atoms)

prefix = make_output_prefix(method_name, basis_name, args.job)

metadata = {

"author": AUTHOR_DEFAULT,

"Telp.": AUTHOR_PHONE_DEFAULT,

"Nama molekul": MOLECULE_NAME,

"Rumus molekul": MOLECULE_FORMULA,

"SMILES": SMILES_DEFAULT,
"IUPAC-like name": IUPAC_NAME_DEFAULT,

"Catatan riset": RESEARCH_NOTE_DEFAULT,

"Jumlah atom": mol.natm,

"Jumlah elektron": mol.nelectron,

```
    "Muatan total": args.charge,
    "Spin PySCF": args.spin,
    "Metode": method_name,
    "Family metode": method_info["family"],
    "XC functional": method_info.get("xc"),
    "Dispersion": method_info.get("disp"),
    "Basis set": basis_name,
    "Basis PySCF": pyscf_basis,
    "Jenis perhitungan": args.job,
    "Energi total Hartree": f"{energy:.12f}",
    "SCF konvergen": bool(getattr(mf, "converged", False)),
    "File checkpoint": getattr(mf, "chkfile", ""),
    "Satuan koordinat": "Angstrom",
    "Bond scale": args.bond_scale,
    "Bond tolerance": args.bond_tolerance,
    "Waktu simpan": datetime.now().strftime("%Y-%m-%d %H:%M:%S"),
}

output_files = {
    "txt_coordinates": str(output_dir / f"{prefix}_koordinat_akhir.txt"),
    "xyz_structure": str(output_dir / f"{prefix}.xyz"),
    "mol_structure": str(output_dir / f"{prefix}.mol"),
    "sdf_structure": str(output_dir / f"{prefix}.sdf"),
    "pdb_structure": str(output_dir / f"{prefix}.pdb"),
    "cml_structure": str(output_dir / f"{prefix}.cml"),
    "json_structure": str(output_dir / f"{prefix}_struktur_lengkap.json"),
    "log_report": str(output_dir / f"{prefix}.log"),
}

if running_output_log_path:
    output_files["running_output_log"] = str(running_output_log_path)

write_txt_coordinates(output_files["txt_coordinates"], atoms, metadata)
write_xyz_file(output_files["xyz_structure"], atoms, metadata)
write_mol_file(output_files["mol_structure"], atoms, bonds, title=MOLECULE_NAME)
```

write_sdf_file(output_files["sdf_structure"], atoms, bonds, metadata)

write_pdb_file(output_files["pdb_structure"], atoms, bonds, metadata)

write_cml_file(output_files["cml_structure"], atoms, bonds, metadata)

write_json_structure_file(output_files["json_structure"], atoms, bonds, angles, dihedrals, distance_matrix, metadata)

write_log_file(output_files["log_report"], atoms, bonds, angles, dihedrals, distance_matrix, metadata, output_files)

```
    return output_files, {
```
"atoms": atoms,

"bonds": bonds,

"angles": angles,

"dihedrals": dihedrals,

}

```
# ============================================================
```
14. INFORMASI ORBITAL

Subbab ini menguraikan # 14. INFORMASI ORBITAL. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def print_orbital_info(mf):
```
mo_energy = getattr(mf, "mo_energy", None)

mo_occ = getattr(mf, "mo_occ", None)

```
    if mo_energy is None or mo_occ is None:
        print("\nInformasi orbital tidak tersedia.")
```
return

```
    if isinstance(mo_energy, tuple) or isinstance(mo_occ, tuple):
        print("\nSistem terbuka-shell terdeteksi.")
        print("Pelaporan HOMO-LUMO sederhana dilewati.")
```
return

occupied = [i for i, occ in enumerate(mo_occ) if occ > 0]

virtual = [i for i, occ in enumerate(mo_occ) if occ == 0]

```
    if not occupied or not virtual:
        print("\nHOMO/LUMO tidak dapat ditentukan dari okupansi orbital.")
```
return

homo_index = max(occupied)

lumo_index = min(virtual)

homo_hartree = mo_energy[homo_index]

lumo_hartree = mo_energy[lumo_index]

gap_hartree = lumo_hartree - homo_hartree

hartree_to_ev = 27.211386245988

homo_ev = homo_hartree * hartree_to_ev

lumo_ev = lumo_hartree * hartree_to_ev

gap_ev = gap_hartree * hartree_to_ev

```
    print("\n============================================================")
    print("INFORMASI ORBITAL SEDERHANA")
    print("============================================================")
    print(f"HOMO index          : {homo_index}")
    print(f"LUMO index          : {lumo_index}")
    print(f"Energi HOMO         : {homo_hartree: .10f} Hartree")
    print(f"Energi LUMO         : {lumo_hartree: .10f} Hartree")
    print(f"Gap HOMO-LUMO       : {gap_hartree: .10f} Hartree")
    print(f"Energi HOMO         : {homo_ev: .6f} eV")
    print(f"Energi LUMO         : {lumo_ev: .6f} eV")
    print("============================================================")
```

```
# ============================================================
```

15. ARGUMEN TERMINAL

Subbab ini menguraikan # 15. ARGUMEN TERMINAL. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================
```

```
def parse_arguments():
```

parser = argparse.ArgumentParser(

description="Template generik perhitungan kimia komputasi molekul dengan PySCF. Default: LAH-01 Letrozole-Anastrozole Hybrid."

)

```
    parser.add_argument("--name", default=MOLECULE_NAME_DEFAULT, help="Nama molekul.")
    parser.add_argument("--formula", default=FORMULA_DEFAULT, help="Rumus molekul
opsional.")
```

```
    parser.add_argument("--xyz-file", default=None, help="File XYZ standar atau blok
koordinat XYZ.")
    parser.add_argument("--coord-file", default=None, help="File teks koordinat:
SimbolAtom X Y Z.")
```

```
    parser.add_argument("--method", default=METHOD_DEFAULT, help=f"Metode komputasi.
Default: {METHOD_DEFAULT}")
    parser.add_argument("--xc", default=None, help="Functional PySCF untuk --method
CUSTOM, contoh: 'm06-2x'.")
    parser.add_argument("--disp", default=None, help="Koreksi dispersi manual, contoh:
d3bj, d3zero, d4.")
```

```
    parser.add_argument("--basis", default=BASIS_DEFAULT, help=f"Basis set. Default:
{BASIS_DEFAULT}")
    parser.add_argument("--basis-custom", default=None, help="Nama basis PySCF untuk --
basis CUSTOM.")

    parser.add_argument("--charge", type=int, default=CHARGE_DEFAULT, help=f"Muatan total
molekul. Default: {CHARGE_DEFAULT}")
    parser.add_argument(
     "--spin",
     type=int,
     default=SPIN_DEFAULT,
     help=f"Spin PySCF = jumlah elektron alfa - beta. Default: {SPIN_DEFAULT}",
   )

    parser.add_argument(
     "--job",
     choices=["singlepoint", "optimize"],
     default=JOB_DEFAULT,
     help=f"Jenis perhitungan: singlepoint atau optimize. Default: {JOB_DEFAULT}",
   )
    parser.add_argument("--optimize", action="store_true", help="Alias lama untuk --job
optimize.")

    parser.add_argument("--grid", type=int, default=DFT_GRID_LEVEL_DEFAULT, help=f"Level
grid DFT. Default: {DFT_GRID_LEVEL_DEFAULT}")
    parser.add_argument("--conv-tol", type=float, default=CONV_TOL_DEFAULT,
help=f"Toleransi konvergensi SCF. Default: {CONV_TOL_DEFAULT}")
    parser.add_argument("--max-cycle", type=int, default=MAX_CYCLE_DEFAULT,
help=f"Maksimum iterasi SCF. Default: {MAX_CYCLE_DEFAULT}")
    parser.add_argument("--verbose", type=int, default=VERBOSE_DEFAULT, help=f"Level
output PySCF. Default: {VERBOSE_DEFAULT}")
    parser.add_argument("--density-fit", action="store_true",
default=USE_DENSITY_FITTING_DEFAULT, help="Aktifkan density fitting.")

    parser.add_argument("--output-dir", default=None, help="Direktori output. Jika kosong,
dibuat otomatis di hasil_pyscf/.")
    parser.add_argument("--bond-scale", type=float, default=BOND_SCALE_DEFAULT,
help=f"Faktor skala inferensi ikatan. Default: {BOND_SCALE_DEFAULT}")
    parser.add_argument("--bond-tolerance", type=float, default=BOND_TOLERANCE_DEFAULT,
help=f"Toleransi inferensi ikatan dalam Angstrom. Default: {BOND_TOLERANCE_DEFAULT}")

    parser.add_argument("--list", action="store_true", help="Tampilkan daftar metode,
basis set, dan contoh pemakaian.")

    return parser.parse_args()


# ============================================================
```

 # 16. PROGRAM UTAMA

Bagian ini memuat # 16. PROGRAM UTAMA sebagai contoh implementasi. Pembaca sebaiknya membaca tujuan blok kerja terlebih dahulu sebelum menjalankan kode, kemudian membandingkan output yang diperoleh dengan parameter metode yang digunakan.

```
# ============================================================
```

```
def main():
    global MOLECULE_NAME, MOLECULE_FORMULA, BASIS_LABEL

    args = parse_arguments()

    if args.list:
        print_available_options()
        return

    if args.optimize:
        args.job = "optimize"

    validate_nonnegative_int(args.grid, "grid")
    validate_positive_float(args.conv_tol, "conv_tol")
    validate_positive_int(args.max_cycle, "max_cycle")
    validate_positive_float(args.bond_scale, "bond_scale")
    validate_nonnegative_int(args.verbose, "verbose")

    MOLECULE_NAME = args.name.strip() if args.name else MOLECULE_NAME_DEFAULT
    MOLECULE_FORMULA = args.formula.strip() if args.formula else "-"

    method_name, method_info = resolve_method(args)
    basis_name, pyscf_basis = resolve_basis(args)
    BASIS_LABEL = basis_name

    # Direktori output dan file running log dibuat sebelum PySCF mulai bekerja,
    # sehingga pesan proses SCF/DFT dan pesan error tetap tersimpan walaupun perhitungan
gagal.
    output_dir = make_output_dir(args, method_name, basis_name)
    running_log_prefix = make_output_prefix(method_name, basis_name, args.job)
    running_output_log_path = output_dir / f"{running_log_prefix}_running_output.log"

    original_stdout = sys.stdout
    original_stderr = sys.stderr
    status_akhir = "SELESAI"

    with open(running_output_log_path, "w", encoding="utf-8", buffering=1) as
running_log_file:
        sys.stdout = TeeOutput(original_stdout, running_log_file)
        sys.stderr = TeeOutput(original_stderr, running_log_file)
        try:
            write_running_log_header(
                log_path=running_output_log_path,
                args=args,
```

```
    method_name=method_name,
    basis_name=basis_name,
    pyscf_basis=pyscf_basis,
)

coordinate_atoms, coordinate_source = get_coordinate_atoms(args)
atom_block = format_atom_block(coordinate_atoms)

mol = build_molecule(
    atom_block=atom_block,
    pyscf_basis=pyscf_basis,
    charge=args.charge,
    spin=args.spin,
    verbose=args.verbose,
)

print_molecule_info(
    mol=mol,
    method_name=method_name,
    method_info=method_info,
    basis_name=basis_name,
    pyscf_basis=pyscf_basis,
    charge=args.charge,
    spin=args.spin,
    job=args.job,
    coordinate_source=coordinate_source,
)

print_coordinates(mol, "KOORDINAT AWAL MOLEKUL")

mf = build_calculator(
    mol=mol,
    method_name=method_name,
    method_info=method_info,
    grid_level=args.grid,
```

```
        conv_tol=args.conv_tol,
        max_cycle=args.max_cycle,
        use_density_fitting=args.density_fit,
    )

    if args.job == "optimize":
        mol_opt = optimize_geometry(mf)
        print_coordinates(mol_opt, "KOORDINAT HASIL OPTIMASI GEOMETRI")
        mf = build_calculator(
            mol=mol_opt,
            method_name=method_name,
            method_info=method_info,
            grid_level=args.grid,
            conv_tol=args.conv_tol,
            max_cycle=args.max_cycle,
            use_density_fitting=args.density_fit,
        )
    print("\n============================================================")
    print("PERHITUNGAN ENERGI DIMULAI")
    print("============================================================")
    print(f"Nama molekul : {MOLECULE_NAME}")
    print(f"Metode       : {method_name}")
    print(f"XC           : {method_info.get('xc')}")
    print(f"Dispersion   : {method_info.get('disp')}")
    print(f"Basis set    : {basis_name} ({pyscf_basis})")
    print(f"Grid DFT     : {args.grid}")
    print(f"Conv tol     : {args.conv_tol}")
    print(f"Max cycle    : {args.max_cycle}")
    print(f"Density fit  : {args.density_fit}")
    print("============================================================\n")

    try:
        energy = mf.kernel()
    except Exception as exc:
        status_akhir = "GAGAL"
        print("\nERROR: Perhitungan PySCF gagal.")
        print(str(exc))
        if method_info.get("disp"):
            print("\nKemungkinan penyebab: koreksi dispersi belum tersedia pada
instalasi PySCF Anda.")
            print("Coba salah satu opsi berikut:")
            print("1. Jalankan tanpa dispersi:")
            print("   python lah01-letrozole-anastrozole-hybrid.py --method B3LYP
--basis def2-SVP")
            print("2. Instal dukungan dispersi sesuai lingkungan Python Anda,
misalnya:")
            print("   pip install pyscf-dispersion simple-dftd3")
        print("\nPENTING:")
        print(f"Teks running/error sudah otomatis tersimpan di:
{running_output_log_path}")
        sys.exit(1)
```

```
            print("\n============================================================")
            print("HASIL PERHITUNGAN")
            print("============================================================")
            print(f"Nama molekul       : {MOLECULE_NAME}")
            print(f"Rumus molekul      : {MOLECULE_FORMULA}")
            print(f"Metode             : {method_name}")
            print(f"Basis set          : {basis_name}")
            print(f"Energi total       : {energy: .12f} Hartree")
            print(f"SCF konvergen      : {mf.converged}")
            print(f"File checkpoint    : {mf.chkfile}")
```

```
            if not mf.converged:
                print("\nPERINGATAN:")
                print("SCF belum konvergen.")
                print("Coba salah satu langkah berikut:")
                print("1. Naikkan --max-cycle, misalnya --max-cycle 250")
                print("2. Longgarkan sementara --conv-tol, misalnya --conv-tol 1e-7")
                print("3. Gunakan basis lebih kecil untuk uji awal, misalnya STO-3G")
                print("4. Optimasi geometri awal di Avogadro atau jalankan --job
optimize")
```

```
            print("============================================================")
```

output_files, structure_summary = save_final_structure_outputs(

mol=mf.mol,

mf=mf,

energy=energy,

method_name=method_name,

method_info=method_info,

basis_name=basis_name,

pyscf_basis=pyscf_basis,

args=args,

output_dir=output_dir,

running_output_log_path=running_output_log_path,

)

```
            print("\n============================================================")
            print("FILE HASIL AKHIR BERHASIL DISIMPAN")
            print("============================================================")
            print(f"Direktori output            : {output_dir}")
            print(f"Jumlah ikatan terdeteksi    : {len(structure_summary['bonds'])}")
            print(f"Jumlah sudut terdeteksi     : {len(structure_summary['angles'])}")
            print(f"Jumlah dihedral terdeteksi  : {len(structure_summary['dihedrals'])}")
            print("")
            for label, filename in output_files.items():
                print(f"{label:18s}: {filename}")
            print("============================================================")
            print("Catatan:")
            print("- .txt berisi koordinat akhir molekul.")
            print("- .xyz berisi koordinat 3D standar.")
            print("- .mol/.sdf/.pdb/.cml dapat dipakai untuk visualisasi struktur 3D.")
            print("- .json dan .log menyimpan data ikatan, sudut, dihedral, jarak, dan
teks running.")
            print("============================================================")
```

print_orbital_info(mf)

```
            print("\nSELESAI.")
```

```
        except SystemExit:
            if status_akhir != "GAGAL":
                status_akhir = "DIHENTIKAN"
            raise
        except Exception:
            status_akhir = "GAGAL"
            print("\nERROR TAK TERDUGA:", file=sys.stderr)
            import traceback
            traceback.print_exc()
            print("\nPENTING:")
            print(f"Teks running/error sudah otomatis tersimpan di:
{running_output_log_path}")
            raise
    finally:
        try:
            write_running_log_footer(running_output_log_path, status=status_akhir)
        finally:
            sys.stdout = original_stdout
            sys.stderr = original_stderr

if __name__ == "__main__":
    main()
```

BAB 6

RANCANGAN PROPOSAL RISET HIBRIDA PARP BERBASIS OLAPARIB-TALAZOPARIB

Pengantar Bab

Bab ini menyajikan rancangan proposal kedua dengan fokus pada senyawa hibrida berbasis Olaparib dan Talazoparib. Pengembangan ini menunjukkan bahwa pola riset komputasi dapat digunakan ulang untuk target dan keluarga senyawa yang berbeda, selama alasan ilmiahnya jelas.

Pembahasan bab ini menekankan pentingnya konsistensi antara mekanisme biologis PARP, desain struktur hibrida, parameter DFT, molecular docking, prediksi ADMET, dan interpretasi hasil. Mahasiswa perlu membedakan dugaan komputasi dari klaim biologis yang membutuhkan validasi eksperimental.

Setelah pengantar ini, pembahasan dilanjutkan dengan orientasi bab, peta kompetensi, uraian materi inti, contoh kerja, dan catatan metodologis yang diperlukan untuk menuntun pembaca dari konsep menuju praktik.

Targetnya jelas yaitu **PARP-1**, dan kedua molekul tersedia dalam daftar senyawa kanker payudara beserta IUPAC dan SMILES. Formatnya disusun mengikuti pola proposal contoh yang dilampirkan:

ada ringkasan, latar belakang, rumusan masalah, tujuan, metode, software, basis set, prosedur, instrumen analisis, jadwal 6 bulan, luaran, risiko, dan daftar pustaka inti.

Untuk mahasiswa S1, bab ini berguna sebagai contoh bahwa penelitian komputasi dapat dikembangkan secara modular. Setelah satu format proposal dikuasai, mahasiswa dapat membuat variasi target atau variasi scaffold dengan tetap menjaga standar validasi dan pelaporan.

Bab ini menjadi pengembangan lanjutan dari pola proposal sebelumnya. Fokusnya bergeser pada target PARP dan desain hibrida berbasis fragmen olaparib-talazoparib. Perbandingan proposal dalam buku ini membantu pembaca memahami bahwa format metodologi dapat dipertahankan, sementara latar belakang biologis, target protein, desain kandidat, dan parameter analisis disesuaikan dengan sistem yang dikaji.

Orientasi Bab

Subbab ini menguraikan Orientasi Bab. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

Judul

Desain In Silico Senyawa Hibrida Olaparib–Talazoparib sebagai Kandidat Inhibitor PARP-1 untuk Terapi Kanker Payudara BRCA-Mutated Berbasis DFT, Molecular Docking, dan Prediksi ADMET

---

1. Ringkasan Proposal

Kanker payudara dengan mutasi gen **BRCA1/BRCA2** memiliki kerentanan biologis terhadap penghambatan mekanisme perbaikan DNA. Salah satu target penting pada jalur ini adalah **poly(ADP-ribose) polymerase-1** atau **PARP-1**, yaitu protein yang berperan dalam perbaikan kerusakan DNA untai tunggal. Penghambatan PARP-1 dapat menimbulkan efek **synthetic lethality** pada sel kanker dengan defisiensi perbaikan DNA homologous recombination, terutama pada kanker dengan mutasi BRCA. Olaparib dan Talazoparib merupakan PARP inhibitor yang relevan pada kanker payudara HER2-negatif dengan mutasi BRCA. (LBBC)

Penelitian ini bertujuan merancang kandidat senyawa hibrida baru berbasis struktur **Olaparib** dan **Talazoparib**. Strategi desain dilakukan dengan mempertahankan farmakofor penting dari kedua molekul, terutama kerangka heterosiklik pengikat kantong nikotinamida PARP-1, gugus karbonil/laktam, cincin aromatik berfluorin, serta fragmen heteroaromatik yang berperan dalam pembentukan interaksi hidrogen dan interaksi hidrofobik. Dalam dokumen molekul kanker payudara yang digunakan sebagai data awal, Olaparib dan Talazoparib tercantum sebagai molekul kecil prioritas dengan nama IUPAC dan SMILES masing-masing.

Metode penelitian meliputi pembuatan struktur 3D kandidat hibrida, optimasi geometri menggunakan DFT, analisis sifat elektronik, molecular docking terhadap PARP-1, validasi docking, prediksi drug-likeness, prediksi ADMET, prediksi toksisitas awal, serta pemeringkatan kandidat. Penelitian ini bersifat komputasi murni, sehingga sesuai untuk mahasiswa S1 dan dapat diselesaikan dalam waktu sekitar enam bulan dengan perangkat lunak terbuka atau semi-terbuka.

---

2. Latar Belakang

Kanker payudara merupakan penyakit yang sangat heterogen secara molekuler. Salah satu kelompok penting adalah kanker payudara dengan mutasi germline atau somatic pada **BRCA1/BRCA2**. Gen BRCA berperan dalam perbaikan DNA melalui mekanisme homologous recombination. Ketika mekanisme tersebut terganggu, sel kanker menjadi lebih bergantung pada jalur perbaikan DNA lain, termasuk jalur yang melibatkan PARP. Oleh karena itu, PARP-1 menjadi target molekuler penting untuk terapi kanker payudara tertentu. (ScienceDirect)

PARP inhibitor bekerja dengan menghambat aktivitas PARP dan dapat menyebabkan akumulasi kerusakan DNA pada sel kanker yang sudah mengalami defisiensi BRCA. Dalam konteks terapi kanker payudara, Olaparib dan Talazoparib banyak dibahas sebagai PARP inhibitor yang digunakan pada pasien kanker payudara HER2-negatif dengan mutasi BRCA. Talazoparib juga dikenal memiliki kemampuan PARP trapping yang kuat, sedangkan Olaparib merupakan salah satu PARP inhibitor penting yang digunakan luas sebagai pembanding dalam riset komputasi PARP-1. (LBBC)

Olaparib dan Talazoparib memiliki karakter struktur yang menarik untuk desain hibrida. Olaparib memiliki fragmen phthalazinone, gugus benzil berfluorin, dan piperazine amide. Talazoparib memiliki kerangka triazole, fluorophenyl, dan sistem heterosiklik yang lebih kaku. Berdasarkan dokumen molekul kanker payudara, Olaparib ditulis dengan nama IUPAC "4-[(3-[(4-cyclopropylcarbonyl)piperazin-1-yl]carbonyl)-4-fluorophenyl]methyl(2H)phthalazin-1-one", sedangkan Talazoparib ditulis dengan nama IUPAC "(8S,9R)-5-fluoro-8-(4-fluorophenyl)-9-(1-methyl-1H-1,2,4-triazol-5-yl)-2,7,8,9-tetrahydro-3H-pyrido[4,3,2-de]phthalazin-3-one".

Riset komputasi terhadap PARP-1 inhibitor sangat relevan karena target ini telah banyak dianalisis menggunakan molecular docking, molecular dynamics, ADMET, QSAR, dan metode berbasis DFT. Kajian komputasi modern menunjukkan bahwa docking dapat digunakan untuk memperkirakan mode ikatan inhibitor PARP-1 pada kantong aktif, sedangkan DFT dapat membantu menjelaskan sifat elektronik ligan seperti HOMO, LUMO, gap energi, momen dipol, dan peta potensial elektrostatik. (PMC)

Pendekatan **molecular hybridization** dipilih karena penggabungan dua farmakofor aktif dalam satu molekul dapat menjadi strategi rasional untuk menghasilkan kandidat baru. Dalam konteks kanker payudara, molecular hybrids telah dibahas sebagai strategi pengembangan kandidat antikanker, baik dalam bentuk **linked hybrid**, **merged hybrid**, maupun **fused hybrid**. (PMC)

Berdasarkan latar belakang tersebut, penelitian ini mengusulkan desain in silico senyawa hibrida Olaparib–Talazoparib sebagai kandidat inhibitor PARP-1 untuk kanker payudara BRCA-mutated. Penelitian diarahkan untuk memperoleh kandidat struktur baru yang secara komputasi memiliki afinitas lebih baik atau minimal sebanding dengan molekul induknya, serta memiliki profil ADMET yang masih layak untuk riset lanjutan.

---

3. Rumusan Masalah

Penelitian ini dirancang untuk menjawab beberapa masalah utama.

Pertama, bagaimana rancangan struktur senyawa hibrida Olaparib–Talazoparib yang rasional secara kimia, tidak terlalu besar, dan masih mempertahankan farmakofor penting inhibitor PARP-1?

Kedua, bagaimana sifat elektronik kandidat hibrida berdasarkan perhitungan DFT, khususnya energi HOMO, energi LUMO, gap HOMO–LUMO, momen dipol, dan molecular electrostatic potential?

Ketiga, bagaimana afinitas prediktif kandidat hibrida terhadap PARP-1 berdasarkan molecular docking?

Keempat, bagaimana pola interaksi kandidat hibrida dengan residu penting dalam binding pocket PARP-1?

Kelima, kandidat hibrida mana yang memiliki kombinasi terbaik antara docking score, pola interaksi, sifat elektronik, drug-likeness, ADMET, dan toksisitas awal?

Keenam, apakah kandidat hibrida tertentu secara komputasi lebih menjanjikan dibandingkan Olaparib dan Talazoparib sebagai ligan pembanding?

---

4. Tujuan Penelitian

Tujuan umum penelitian ini adalah merancang dan mengevaluasi kandidat senyawa hibrida Olaparib–Talazoparib sebagai calon inhibitor PARP-1 untuk kanker payudara BRCA-mutated menggunakan pendekatan kimia komputasi.

Tujuan khusus penelitian ini adalah sebagai berikut.

Pertama, menyusun pustaka kecil kandidat hibrida Olaparib–Talazoparib sebanyak 8–12 senyawa berdasarkan prinsip molecular hybridization.

Kedua, mengoptimasi struktur geometri Olaparib, Talazoparib, dan kandidat hibrida menggunakan metode DFT.

Ketiga, menganalisis sifat elektronik kandidat hibrida melalui parameter HOMO, LUMO, gap HOMO–LUMO, momen dipol, dan molecular electrostatic potential.

Keempat, melakukan molecular docking kandidat hibrida terhadap protein PARP-1.

Kelima, menganalisis interaksi protein-ligan, termasuk ikatan hidrogen, interaksi hidrofobik, π-π stacking, π-alkyl, dan van der Waals.

Keenam, memprediksi drug-likeness, ADMET, dan toksisitas awal kandidat hibrida.

Ketujuh, menentukan 1–3 kandidat hibrida terbaik yang layak direkomendasikan untuk riset komputasi lanjutan, molecular dynamics, atau sintesis eksperimental.

---

5. Hipotesis Penelitian

Hipotesis utama penelitian ini adalah bahwa penggabungan farmakofor Olaparib dan Talazoparib melalui desain hibrida yang rasional dapat menghasilkan kandidat molekul baru yang secara

komputasi menunjukkan afinitas lebih baik atau minimal sebanding terhadap PARP-1 dibandingkan salah satu molekul induknya.

Hipotesis operasionalnya adalah kandidat hibrida yang mempertahankan motif heterosiklik pengikat kantong nikotinamida PARP-1, gugus karbonil/laktam, serta cincin aromatik berfluorin akan menunjukkan docking score lebih negatif, pola interaksi residu yang lebih stabil, dan profil ADMET yang masih dapat diterima.

---

6. Kebaruan Penelitian

Kebaruan penelitian ini terletak pada desain pustaka kecil senyawa hibrida Olaparib–Talazoparib yang difokuskan pada kandidat inhibitor PARP-1 untuk kanker payudara BRCA-mutated. Penelitian sebelumnya telah banyak membahas PARP inhibitor secara klinis dan komputasi, tetapi desain hibrida sederhana yang menggabungkan motif struktur Olaparib dan Talazoparib masih dapat dikembangkan sebagai topik skripsi komputasi S1.

Kebaruan praktisnya adalah penelitian ini menghasilkan alur kerja yang mudah direplikasi mahasiswa S1: desain struktur, optimasi DFT, docking, visualisasi interaksi, ADMET, toksisitas, dan pemeringkatan kandidat dalam satu workflow yang rapi dan dapat diselesaikan dalam waktu enam bulan.

---

7. Batasan Penelitian

Penelitian ini merupakan penelitian **in silico** dan tidak mencakup sintesis kimia, uji enzim PARP-1, uji kultur sel, uji hewan, maupun uji klinis.

Jumlah kandidat hibrida dibatasi pada 8–12 senyawa agar sesuai untuk skripsi S1.

Target protein utama hanya **PARP-1**. Target lain seperti ERα, aromatase, HER2, CDK4/6, dan PI3Kα tidak menjadi target utama dalam penelitian ini.

Molecular docking digunakan sebagai metode prediksi awal afinitas, bukan bukti aktivitas biologis final.

Prediksi ADMET dan toksisitas bersifat komputasional dan digunakan hanya untuk penyaringan awal.

DFT dilakukan pada ligan bebas, bukan pada kompleks protein-ligan penuh, karena perhitungan kuantum seluruh kompleks protein-ligan terlalu besar untuk skripsi S1.

Molecular dynamics tidak dijadikan metode wajib. Jika waktu dan fasilitas mencukupi, MD 10 ns dapat dilakukan hanya untuk satu kandidat terbaik dan ligan pembanding.

---

8. Manfaat Penelitian

Secara teoretis, penelitian ini memberikan contoh penerapan molecular hybridization, DFT, docking, dan ADMET dalam desain kandidat inhibitor PARP-1.

Secara praktis, penelitian ini menghasilkan kandidat struktur baru berbasis Olaparib–Talazoparib yang dapat menjadi dasar penelitian lanjutan.

Secara pendidikan, penelitian ini melatih mahasiswa S1 dalam alur kerja kimia komputasi modern, mulai dari desain ligan, pembuatan struktur 3D, optimasi geometri, docking, analisis elektronik, sampai evaluasi ADMET.

Secara metodologis, penelitian ini dapat menjadi model skripsi desain obat berbasis struktur untuk topik antikanker payudara.

---

9. Desain Penelitian

Jenis penelitian ini adalah **penelitian kimia komputasi berbasis desain molekul, docking, dan virtual screening terbatas**.

Model penelitian yang digunakan adalah:

Desain struktur hibrida → pembuatan struktur 3D → optimasi awal → optimasi DFT → docking PARP-1 → validasi docking → analisis interaksi → prediksi ADMET → pemeringkatan kandidat → rekomendasi kandidat terbaik.

Objek penelitian adalah Olaparib, Talazoparib, dan kandidat hibrida Olaparib–Talazoparib.

Target molekuler adalah PARP-1.

Kontrol positif adalah Olaparib dan Talazoparib.

Parameter utama keberhasilan adalah docking score, posisi ligan dalam binding pocket PARP-1, jumlah dan kualitas interaksi residu, HOMO–LUMO gap, molecular electrostatic potential, drug-likeness, ADMET, dan prediksi toksisitas.

---

10. Data dan Bahan Komputasi

Data ligan utama adalah Olaparib dan Talazoparib. Struktur awal dapat diperoleh dari SMILES yang sudah tersedia dalam dokumen molekul kanker payudara Bapak atau dari basis data molekul publik. Dalam dokumen tersebut, Olaparib dan Talazoparib dicantumkan sebagai molekul kecil yang relevan untuk kanker payudara dan memiliki representasi SMILES yang dapat digunakan untuk membangun struktur 3D.

Data protein target adalah struktur tiga dimensi PARP-1 dari Protein Data Bank. Struktur yang disarankan adalah struktur PARP-1 yang memiliki ligan ko-kristal berupa inhibitor PARP atau analog nikotinamida sehingga dapat digunakan untuk validasi redocking.

Data pembanding berupa docking score dan pola interaksi Olaparib serta Talazoparib terhadap PARP-1.

Data output berupa geometri teroptimasi, energi total, HOMO, LUMO, gap energi, momen dipol, molecular electrostatic potential, docking score, residu interaksi, parameter ADMET, dan prediksi toksisitas.

---

11. Perangkat Lunak yang Digunakan

RDKit digunakan untuk membaca SMILES, membuat struktur 3D, menambahkan atom hidrogen, melakukan minimisasi awal MMFF94, menghasilkan konformer, menghitung descriptor molekul dasar, dan membantu pembuatan pustaka kandidat hibrida.

Avogadro digunakan untuk menggambar struktur molekul, memeriksa geometri 3D, mengatur konformasi awal, dan melakukan minimisasi awal secara visual.

Open Babel digunakan untuk konversi format molekul seperti SMILES, MOL, SDF, PDB, dan PDBQT.

ORCA atau PySCF digunakan untuk optimasi geometri DFT, perhitungan energi elektronik, energi HOMO, energi LUMO, gap HOMO–LUMO, dan momen dipol.

xTB digunakan sebagai pilihan tambahan untuk pre-optimasi cepat menggunakan metode GFN2-xTB, terutama jika kandidat hibrida terlalu besar untuk langsung dioptimasi dengan DFT.

AutoDockTools digunakan untuk preparasi protein dan ligan ke format PDBQT, penambahan hidrogen polar, pengaturan muatan, serta penentuan grid box.

AutoDock Vina digunakan untuk molecular docking.

PyRx dapat digunakan sebagai antarmuka docking agar workflow lebih mudah bagi mahasiswa S1.

Discovery Studio Visualizer digunakan untuk visualisasi interaksi 2D protein-ligan.

PyMOL digunakan untuk visualisasi struktur 3D kompleks protein-ligan.

SwissADME digunakan untuk prediksi drug-likeness, Lipinski, TPSA, logP, GI absorption, bioavailability radar, dan synthetic accessibility.

pkCSM atau admetSAR digunakan untuk prediksi ADMET lanjutan.

ProTox-II digunakan untuk prediksi toksisitas awal.

Python, pandas, matplotlib, dan Excel/LibreOffice Calc digunakan untuk rekapitulasi data, grafik, analisis deskriptif, dan pemeringkatan kandidat.

---

12. Metode Komputasi dan Parameter

12.1 Desain Senyawa Hibrida

Kandidat hibrida dirancang berdasarkan penggabungan farmakofor Olaparib dan Talazoparib. Farmakofor yang dipertahankan adalah cincin phthalazinone atau motif laktam terkait, cincin aromatik berfluorin, gugus karbonil, fragmen heteroaromatik, dan pusat interaksi yang memungkinkan pembentukan ikatan hidrogen dengan PARP-1.

Jenis hibrida yang dirancang meliputi **merged hybrid** dan **linked hybrid**. Merged hybrid dibuat dengan menggabungkan motif phthalazinone Olaparib dengan motif triazole/fluorophenyl Talazoparib dalam satu kerangka kompak. Linked hybrid dibuat dengan menghubungkan fragmen Olaparib dan Talazoparib menggunakan linker pendek seperti metilena, amida, urea, atau eter.

Jumlah kandidat hibrida dibatasi 8–12 senyawa. Setiap kandidat diberi kode, misalnya OTH-01 sampai OTH-12.

12.2 Optimasi Awal Struktur

Semua ligan dibangun dari SMILES atau digambar manual. Struktur 3D dibangkitkan menggunakan RDKit dengan metode embedding ETKDG. Atom hidrogen ditambahkan secara eksplisit.

Struktur awal diminimisasi menggunakan **MMFF94** atau **UFF**. Jika kandidat memiliki banyak rotatable bonds, dilakukan pencarian konformer sederhana sebanyak 20–50 konformer. Konformer dengan energi paling rendah digunakan sebagai struktur awal untuk DFT.

12.3 Optimasi Geometri DFT

Optimasi geometri ligan dilakukan menggunakan metode:

B3LYP-D3BJ/def2-SVP

Metode ini dipilih karena cukup seimbang untuk molekul organik berukuran sedang dan mengandung interaksi dispersi. Koreksi dispersi D3BJ digunakan karena kandidat hibrida mengandung cincin aromatik dan interaksi non-kovalen internal yang dapat memengaruhi geometri.

Untuk validasi terbatas pada 2–3 kandidat terbaik, dapat dilakukan single-point energy menggunakan:

B3LYP-D3BJ/def2-TZVP

atau

M06-2X/def2-SVP

Parameter DFT yang dianalisis adalah energi total, energi HOMO, energi LUMO, gap HOMO–LUMO, momen dipol, dan molecular electrostatic potential.

Jika perhitungan DFT terlalu berat, workflow alternatif yang masih sesuai untuk S1 adalah:

pre-optimasi **GFN2-xTB**, kemudian single-point **B3LYP-D3BJ/def2-SVP** pada struktur terbaik.

12.4 Preparasi Protein PARP-1

Struktur PARP-1 diunduh dari Protein Data Bank. Protein dipersiapkan dengan menghapus molekul air yang tidak relevan, mempertahankan ligan ko-kristal untuk validasi redocking, menambahkan hidrogen polar, dan menentukan muatan sesuai kebutuhan docking.

Binding pocket ditentukan berdasarkan posisi ligan ko-kristal atau kantong nikotinamida PARP-1. Grid box dibuat mencakup seluruh daerah pengikatan ligan ko-kristal.

12.5 Validasi Docking

Validasi docking dilakukan dengan metode redocking. Ligan ko-kristal dipisahkan dari protein, lalu didocking kembali ke binding pocket asalnya.

Docking dianggap valid jika pose hasil redocking mendekati pose kristal. Kriteria umum yang digunakan adalah RMSD kurang dari 2,0 Å. Jika RMSD lebih besar, parameter grid box, protonasi, atau preparasi protein diperbaiki.

Jika ligan ko-kristal sulit digunakan, validasi alternatif dilakukan dengan docking Olaparib dan Talazoparib, lalu membandingkan pola interaksi dengan literatur komputasi PARP-1.

12.6 Molecular Docking

Docking dilakukan menggunakan AutoDock Vina. Semua ligan hasil optimasi dikonversi menjadi format PDBQT. Protein PARP-1 juga disiapkan dalam format PDBQT.

Parameter docking yang digunakan adalah:

exhaustiveness: 8–16

num_modes: 10–20

energy_range: 3 kcal/mol

Pose terbaik dipilih berdasarkan docking score paling negatif, posisi ligan dalam binding pocket, dan kualitas interaksi residu.

12.7 Analisis Interaksi Protein-Ligan

Analisis interaksi dilakukan menggunakan Discovery Studio Visualizer dan PyMOL. Interaksi yang dianalisis meliputi ikatan hidrogen, interaksi hidrofobik, π-π stacking, π-alkyl, van der Waals, dan interaksi dengan residu penting PARP-1.

Kandidat terbaik tidak hanya ditentukan dari docking score, tetapi juga dari kesesuaian orientasi, kemiripan pola interaksi dengan inhibitor pembanding, serta tidak adanya benturan sterik besar dalam binding pocket.

12.8 Prediksi Drug-Likeness dan ADMET

Prediksi drug-likeness dilakukan menggunakan SwissADME. Parameter yang dianalisis meliputi molecular weight, logP, TPSA, hydrogen bond donor, hydrogen bond acceptor, rotatable bonds, Lipinski rule of five, Veber rule, GI absorption, dan synthetic accessibility.

Prediksi ADMET dilakukan menggunakan pkCSM atau admetSAR. Parameter yang dianalisis meliputi absorpsi intestinal, distribusi, blood-brain barrier permeability, metabolisme CYP450, clearance, hepatotoxicity, AMES toxicity, dan skin sensitization.

Prediksi toksisitas awal dilakukan menggunakan ProTox-II untuk memperkirakan kelas toksisitas, LD50 prediktif, dan kemungkinan endpoint toksik.

---

13. Prosedur Penelitian

Tahap 1: Studi Literatur dan Penentuan Molekul Induk

Mahasiswa mengumpulkan literatur tentang kanker payudara BRCA-mutated, PARP-1, synthetic lethality, Olaparib, Talazoparib, molecular hybridization, DFT, docking, ADMET, dan toksisitas komputasi. Pada tahap ini juga dikumpulkan struktur SMILES Olaparib dan Talazoparib dari dokumen awal dan basis data publik.

Tahap 2: Desain Kandidat Hibrida

Mahasiswa membuat 8–12 kandidat hibrida Olaparib–Talazoparib. Setiap kandidat diberi kode OTH-01 sampai OTH-12. Struktur digambar menggunakan Avogadro, MarvinSketch, ChemDraw, atau RDKit.

Kandidat yang terlalu besar, terlalu fleksibel, atau melanggar banyak aturan drug-likeness dieliminasi sejak awal.

Tahap 3: Pembuatan Struktur 3D dan Minimisasi Awal

Semua kandidat dikonversi menjadi struktur 3D. Atom hidrogen ditambahkan. Struktur diminimisasi menggunakan MMFF94, UFF, atau GFN2-xTB.

Struktur yang gagal diminimisasi diperiksa ulang secara manual.

Tahap 4: Optimasi Geometri DFT

Olaparib, Talazoparib, dan kandidat hibrida dioptimasi menggunakan B3LYP-D3BJ/def2-SVP. Output DFT diperiksa untuk memastikan struktur konvergen dan tidak memiliki geometri yang tidak masuk akal.

Data HOMO, LUMO, gap HOMO–LUMO, energi total, dan momen dipol dikumpulkan.

Tahap 5: Preparasi Protein PARP-1

Struktur PARP-1 diunduh dari PDB. Protein dibersihkan, hidrogen polar ditambahkan, muatan disiapkan, dan ligan ko-kristal dipisahkan untuk redocking.

Grid box ditentukan berdasarkan posisi ligan ko-kristal atau binding pocket PARP-1.

Tahap 6: Validasi Docking

Validasi redocking dilakukan. Jika nilai RMSD redocking memenuhi kriteria, parameter docking digunakan untuk seluruh kandidat. Jika tidak memenuhi, grid box, protonasi, dan preparasi protein diperbaiki.

Tahap 7: Docking Ligan

Olaparib, Talazoparib, dan kandidat hibrida didocking ke PARP-1. Setiap docking menghasilkan beberapa pose. Pose terbaik dipilih berdasarkan docking score, orientasi di binding pocket, dan interaksi residu.

Tahap 8: Analisis Interaksi

Pose docking divisualisasikan menggunakan Discovery Studio Visualizer dan PyMOL. Interaksi 2D dan 3D dianalisis, kemudian dibandingkan dengan Olaparib dan Talazoparib sebagai kontrol positif.

Tahap 9: Prediksi ADMET dan Toksisitas

Kandidat dengan docking score baik dianalisis menggunakan SwissADME, pkCSM/admetSAR, dan ProTox-II. Kandidat dengan toksisitas prediktif tinggi, nilai logP ekstrem, TPSA terlalu tinggi, atau pelanggaran drug-likeness berat diberi peringkat rendah.

Tahap 10: Pemeringkatan Kandidat

Pemeringkatan dilakukan dengan sistem skor gabungan. Parameter yang dipakai adalah docking score, jumlah interaksi penting, HOMO–LUMO gap, MEP, logP, TPSA, Lipinski, GI absorption, toksisitas, dan synthetic accessibility.

Tahap 11: Penyusunan Kesimpulan

Mahasiswa menentukan 1–3 kandidat terbaik dan menjelaskan alasan pemilihannya berdasarkan data komputasi.

---

14. Instrumen Analisis Data

Analisis struktur dan energi elektronik menggunakan output DFT dari ORCA, PySCF, atau Gaussian bila tersedia. Data yang diambil meliputi energi total, HOMO, LUMO, gap energi, momen dipol, dan MEP.

Analisis docking menggunakan output AutoDock Vina. Parameter utama adalah docking score dalam kcal/mol, pose ligan, dan interaksi dengan residu PARP-1.

Analisis visual interaksi menggunakan Discovery Studio Visualizer dan PyMOL. Parameter yang diamati meliputi ikatan hidrogen, interaksi hidrofobik, π-π stacking, π-alkyl, dan van der Waals.

Analisis drug-likeness menggunakan SwissADME. Parameter utama adalah Lipinski, TPSA, logP, rotatable bonds, GI absorption, dan synthetic accessibility.

Analisis ADMET menggunakan pkCSM atau admetSAR. Parameter utama adalah absorpsi, distribusi, metabolisme CYP450, ekskresi, hepatotoxicity, AMES toxicity, dan skin sensitization.

Analisis toksisitas awal menggunakan ProTox-II. Parameter utama adalah LD50 prediktif, kelas toksisitas, dan endpoint toksik.

Analisis pemeringkatan menggunakan Excel, LibreOffice Calc, atau Python pandas. Kandidat diberi skor gabungan untuk menentukan molekul terbaik.

---

15. Teknik Analisis dan Kriteria Pemilihan Kandidat Terbaik

Kandidat terbaik dipilih berdasarkan beberapa kriteria.

Pertama, docking score kandidat harus lebih negatif atau minimal sebanding dengan Olaparib dan Talazoparib.

Kedua, pose docking harus berada di binding pocket PARP-1 dan tidak menyimpang dari daerah pengikatan ligan pembanding.

Ketiga, kandidat harus memiliki interaksi penting, seperti ikatan hidrogen, interaksi hidrofobik, π-π stacking, atau interaksi van der Waals dengan residu penting PARP-1.

Keempat, HOMO–LUMO gap dianalisis secara relatif terhadap molekul induk. Gap yang terlalu kecil dapat menunjukkan reaktivitas terlalu tinggi, sedangkan gap yang terlalu besar dapat menunjukkan molekul terlalu inert.

Kelima, MEP digunakan untuk menilai daerah kaya elektron dan miskin elektron yang berpotensi berperan dalam interaksi dengan protein.

Keenam, parameter ADMET harus layak. Kandidat yang melanggar banyak aturan Lipinski, memiliki logP ekstrem, TPSA terlalu tinggi, atau toksisitas prediktif tinggi tidak diprioritaskan.

Ketujuh, kandidat harus masih rasional secara struktur dan tidak terlalu sulit untuk kemungkinan sintesis lanjutan.

---

16. Rencana Jadwal Penelitian Enam Bulan

Bulan pertama digunakan untuk studi literatur, pengumpulan struktur ligan, pengunduhan protein PARP-1, dan penyusunan desain awal kandidat hibrida.

Bulan kedua digunakan untuk menggambar struktur, membuat SMILES kandidat, membangkitkan struktur 3D, melakukan minimisasi MMFF94/UFF/GFN2-xTB, dan memilih kandidat awal.

Bulan ketiga digunakan untuk optimasi geometri DFT dan analisis sifat elektronik.

Bulan keempat digunakan untuk preparasi protein, validasi docking, dan molecular docking semua kandidat.

Bulan kelima digunakan untuk analisis interaksi protein-ligan, prediksi ADMET, prediksi toksisitas, dan pemeringkatan kandidat.

Bulan keenam digunakan untuk penyusunan skripsi, pembuatan grafik, pembahasan, revisi, dan persiapan ujian.

---

17. Luaran Penelitian

Luaran utama penelitian adalah:

satu pustaka kecil kandidat hibrida Olaparib–Talazoparib;

struktur 3D teroptimasi dari Olaparib, Talazoparib, dan kandidat hibrida;

data DFT berupa energi total, HOMO, LUMO, gap energi, momen dipol, dan MEP;

data docking terhadap PARP-1;

visualisasi interaksi 2D dan 3D;

data drug-likeness, ADMET, dan toksisitas awal;

peringkat kandidat terbaik;

naskah skripsi S1;

dan satu draft artikel ilmiah sederhana berbasis hasil skripsi.

---

18. Risiko Penelitian dan Strategi Mitigasi

Risiko pertama adalah struktur hibrida menjadi terlalu besar dan melanggar aturan drug-likeness. Strateginya adalah menggunakan desain merged hybrid yang kompak dan membatasi panjang linker.

Risiko kedua adalah optimasi DFT memerlukan waktu lama. Strateginya adalah melakukan screening awal dengan MMFF94, UFF, atau GFN2-xTB, kemudian DFT dilakukan pada kandidat yang lolos seleksi awal.

Risiko ketiga adalah validasi docking gagal. Strateginya adalah memperbaiki grid box, memeriksa protonasi, mempertahankan ligan ko-kristal pada tahap validasi, dan membandingkan dengan protokol docking PARP-1 dalam literatur.

Risiko keempat adalah kandidat hibrida tidak lebih baik dari molekul induk. Strateginya adalah tetap menjadikan hasil tersebut sebagai temuan ilmiah dan membahas hubungan struktur-aktivitas secara rasional.

Risiko kelima adalah kandidat memiliki ADMET buruk. Strateginya adalah melakukan redesign dengan mengurangi berat molekul, menurunkan logP, mengurangi rotatable bonds, atau mengganti linker yang terlalu hidrofobik.

---

19. Etika dan Keselamatan Penelitian

Penelitian ini tidak menggunakan manusia, hewan, jaringan biologis, data pasien, atau sampel klinis, sehingga tidak memerlukan izin etik biomedis. Namun, penelitian tetap harus mengikuti etika akademik, yaitu mencantumkan sumber data molekul, sumber protein, perangkat lunak, dan referensi ilmiah secara benar.

Hasil penelitian tidak boleh diklaim sebagai obat kanker baru yang terbukti efektif. Istilah yang tepat adalah “kandidat senyawa hibrida potensial berdasarkan prediksi komputasi” atau “kandidat awal untuk penelitian lanjutan”.

---

20. Struktur Bab Skripsi yang Disarankan

Bab I Pendahuluan berisi latar belakang, rumusan masalah, tujuan, manfaat, batasan penelitian, dan kebaruan penelitian.

Bab II Tinjauan Pustaka berisi kanker payudara BRCA-mutated, PARP-1, synthetic lethality, Olaparib, Talazoparib, molecular hybridization, DFT, molecular docking, ADMET, dan toksisitas komputasi.

Bab III Metode Penelitian berisi desain penelitian, alat dan bahan komputasi, software, prosedur desain ligan, DFT, docking, ADMET, toksisitas, dan teknik analisis data.

Bab IV Hasil dan Pembahasan berisi desain kandidat, hasil optimasi geometri, hasil DFT, docking score, interaksi residu, ADMET, toksisitas, dan pemeringkatan kandidat.

Bab V Kesimpulan dan Saran berisi kandidat terbaik, kesimpulan utama, keterbatasan, dan rekomendasi penelitian lanjutan.

---

21. Contoh Kandidat Hibrida yang Dirancang

Kandidat awal dapat diberi kode sebagai berikut.

OTH-01: hibrida Olaparib–Talazoparib dengan scaffold phthalazinone Olaparib dan cincin fluorophenyl Talazoparib.

OTH-02: hibrida dengan linker metilena pendek antara fragmen phthalazinone dan triazole.

OTH-03: hibrida dengan linker amida untuk meningkatkan potensi ikatan hidrogen.

OTH-04: hibrida dengan linker urea untuk meningkatkan donor dan akseptor ikatan hidrogen.

OTH-05: hibrida dengan pengurangan fragmen piperazine agar berat molekul lebih rendah.

OTH-06: hibrida dengan dua atom fluor pada cincin aromatik untuk menguji efek halogenasi.

OTH-07: hibrida dengan motif triazole Talazoparib dan phthalazinone Olaparib yang digabung secara merged hybrid.

OTH-08: hibrida dengan penggantian cyclopropyl carbonyl Olaparib menjadi fragmen heteroaromatik kecil.

OTH-09: hibrida dengan rotatable bond rendah untuk meningkatkan rigiditas.

OTH-10: hibrida kompak dengan modifikasi cincin aromatik berfluorin dan satu gugus karbonil utama.

Daftar kandidat tersebut masih berupa rancangan awal dan dapat disesuaikan setelah analisis drug-likeness awal.

---

22. Format Data Hasil yang Dikumpulkan

Untuk setiap ligan, data yang dikumpulkan meliputi:

kode ligan;

nama ligan;

SMILES;

formula molekul;

berat molekul;

logP;

TPSA;

jumlah donor ikatan hidrogen;

jumlah akseptor ikatan hidrogen;

jumlah rotatable bonds;

energi total DFT;

energi HOMO;

energi LUMO;

gap HOMO–LUMO;

momen dipol;

docking score;

residu interaksi utama;

jumlah ikatan hidrogen;

jenis interaksi hidrofobik;

status Lipinski;

prediksi absorpsi gastrointestinal;

prediksi toksisitas;

synthetic accessibility;

dan skor akhir kandidat.

23. Daftar Pustaka Inti

Dokumen “Molekul Kanker Payudara” sebagai sumber awal nama molekul, IUPAC, dan SMILES Olaparib serta Talazoparib.

Contoh proposal hibrida Letrozole–Anastrozole yang digunakan sebagai pola penyusunan struktur proposal, alur metode, software, jadwal, instrumen analisis, dan format pembahasan.

Review kemajuan terapi PARP inhibitor pada kanker payudara, termasuk pembahasan Olaparib dan Talazoparib pada kanker payudara BRCA-mutated. (ScienceDirect)

Artikel tentang Olaparib sebagai terapi untuk metastatic breast cancer dengan germline BRCA mutation dan HER2-negative disease. (PMC)

Ringkasan klinis PARP inhibitors untuk kanker payudara BRCA-positive, termasuk Olaparib dan Talazoparib. (LBBC)

Kajian computational chemistry untuk pengembangan PARP1 inhibitors yang membahas docking, interaksi Olaparib, QSAR, MD, dan pendekatan komputasi lain dalam desain inhibitor PARP1. (PMC)

Studi computational evaluation kandidat PARP-1 inhibitors untuk breast cancer yang menggunakan docking, ADMET, dan pembanding Olaparib/Talazoparib. (MDPI)

Riset in silico drug discovery kandidat inhibitor PARP1 untuk triple-negative breast cancer yang menggunakan molecular docking dan ADMET analysis. (ScienceDirect)

---

24. Kesimpulan Rancangan Proposal

Proposal ini layak untuk skripsi S1 karena ruang lingkupnya jelas, target molekulernya spesifik, molekul induknya sudah dikenal, metode komputasinya standar, dan alur kerja dapat diselesaikan dalam waktu enam bulan. Desain hibrida Olaparib–Talazoparib memberikan kesempatan kepada mahasiswa untuk memahami desain inhibitor PARP-1 berbasis struktur, sekaligus mempelajari DFT, docking, ADMET, dan toksisitas komputasi dalam satu penelitian yang terintegrasi.

Penelitian ini tidak bertujuan langsung membuktikan aktivitas klinis, tetapi menghasilkan kandidat molekul prioritas berdasarkan prediksi komputasi. Kandidat terbaik dari penelitian ini dapat direkomendasikan untuk molecular dynamics, perhitungan MM/GBSA, sintesis, atau uji biologis lanjutan.

Proposal ini dapat langsung dijadikan contoh kedua setelah proposal **Letrozole–Anastrozole**. Topik **Olaparib–Talazoparib** sangat cocok untuk mahasiswa S1 yang ingin riset desain obat berbasis target **PARP-1** tanpa harus menangani molekul terlalu besar seperti Paclitaxel, Docetaxel, atau Doxorubicin.

File Python Olaparib–Talazoparib Hybrid

Berikut file Python untuk kandidat riset proposal **Olaparib–Talazoparib Hybrid** dengan nama default:

OTH-01 Olaparib-Talazoparib Hybrid

File diturunkan dari pola **paracetamol.py** dan mengikuti arahan **prompt-genmolekul.txt**: memakai PySCF, mendukung HF/DFT, basis set pilihan, input XYZ eksternal, optimasi geometri, output otomatis, log running, serta ekspor struktur ke TXT, XYZ, MOL, SDF, PDB, CML, JSON, dan LOG.

Nama file:

oth01-olaparib-talazoparib-hybrid.py

Molekul default:

OTH-01 Olaparib-Talazoparib Hybrid

Formula: C24H22FN7O3

Jumlah atom XYZ: 57 atom

Charge: 0

Spin: 0

Metode default: B3LYP-D3BJ

Basis default: def2-SVP

SMILES kandidat hibrida yang dimasukkan:

```
O=C1NN=C(c2ccccc21)Cc3ccc(F)c(c3)C(=O)N4CCN(CC4)C(=O)c5nn(C)cn5
```

File sudah saya validasi:

- struktur XYZ 3D sudah dimasukkan langsung ke dalam DEFAULT_MOLECULE_XYZ;
- jumlah atom pada header XYZ = **57**, sesuai jumlah baris atom;
- semua koordinat numerik;
- file lolos pemeriksaan sintaks Python dengan py_compile;
- PySCF belum dijalankan di sandbox ini karena modul PySCF tidak tersedia di lingkungan eksekusi.

Download file Python OTH-01 Olaparib–Talazoparib Hybrid

Cara menjalankan:

python oth01-olaparib-talazoparib-hybrid.py

Jika koreksi D3BJ belum tersedia:

python oth01-olaparib-talazoparib-hybrid.py --method B3LYP --basis def2-SVP

Untuk optimasi geometri:

python oth01-olaparib-talazoparib-hybrid.py --job optimize --method B3LYP --basis def2-SVP

File hasil ini dapat dipakai untuk menghasilkan struktur teroptimasi dan file **XYZ/SDF/PDB** yang kemudian digunakan untuk **docking PARP-1**, ADMET, dan analisis lanjutan sesuai proposal.

Skrip File Python oth01-olaparib-talazoparib-hybrid.py

```
#!/usr/bin/env python3
# author: Kasmui
# Telp.: 0818294312
# ============================================================
#!/usr/bin/env python3
# author: Kasmui
# Telp.: 0818294312
# ============================================================
# FILE RISET KIMIA KOMPUTASI PySCF UNTUK HIBRIDA OLAPARIB-TALAZOPARIB
# ============================================================
#
# Molekul default : OTH-01 Olaparib-Talazoparib Hybrid
# Rumus molekul   : C24H22FN7O3
# Program         : PySCF
#
# Metode default yang dipilih:
#   B3LYP-D3BJ / def2-SVP
#
# Alasan ringkas:
#   - OTH-01 Olaparib-Talazoparib Hybrid adalah molekul organik netral tertutup-shell.
#   - B3LYP umum dipakai untuk molekul organik.
#   - Koreksi dispersi D3BJ membantu interaksi non-kovalen dan konformasi.
#   - def2-SVP cukup seimbang untuk latihan/skrining awal tanpa terlalu berat.
#
# Catatan penting:
#   - Untuk memakai koreksi dispersi D3BJ, PySCF memerlukan dukungan
#     pyscf-dispersion/simple-dftd3 pada beberapa instalasi.
#   - Jika belum tersedia, jalankan metode B3LYP biasa:
#       python oth01-olaparib-talazoparib-hybrid.py --method B3LYP --basis def2-SVP
#
# Contoh pemakaian:
#   python oth01-olaparib-talazoparib-hybrid.py
#   python oth01-olaparib-talazoparib-hybrid.py --list
#   python oth01-olaparib-talazoparib-hybrid.py --method B3LYP --basis def2-SVP
#   python oth01-olaparib-talazoparib-hybrid.py --method PBE0 --basis def2-TZVP
#   python oth01-olaparib-talazoparib-hybrid.py --method CUSTOM --xc "m06-2x" --basis
def2-TZVP
#   python oth01-olaparib-talazoparib-hybrid.py --xyz-file molekul.xyz --name "Molekul
Baru" --formula "C2H6O"
#   python oth01-olaparib-talazoparib-hybrid.py --coord-file koordinat.txt --method B3LYP
--basis "6-31G**"
#   python oth01-olaparib-talazoparib-hybrid.py --job optimize --method B3LYP --basis
def2-SVP
#
# ============================================================

import argparse
import json
import math
import sys
from datetime import datetime
from html import escape
```

```
from pathlib import Path

from pyscf import dft, gto, scf


# ============================================================
# 1. DAFTAR METODE KOMPUTASI
# ============================================================
# family:
#   HF  = Hartree-Fock
#   DFT = Kohn-Sham DFT
# xc:
#   String functional yang dibaca oleh PySCF.
# disp:
#   None       = tanpa koreksi dispersi eksplisit.
#   "d3bj"    = koreksi dispersi D3 dengan Becke-Johnson damping.
#
# CUSTOM:
#   Dipakai bersama argumen --xc.

AVAILABLE_METHODS = {
    "HF": {
        "family": "HF",
        "xc": None,
        "disp": None,
        "description": "Hartree-Fock; RHF untuk spin 0, UHF untuk spin bukan 0.",
    },
    "LDA": {
        "family": "DFT",
        "xc": "lda,vwn",
        "disp": None,
        "description": "DFT LDA sederhana.",
    },
    "PBE": {
        "family": "DFT",
        "xc": "pbe,pbe",
        "disp": None,
        "description": "DFT GGA PBE.",
    },
    "BLYP": {
        "family": "DFT",
        "xc": "b88,lyp",
        "disp": None,
        "description": "DFT GGA BLYP.",
    },
    "B3LYP": {
        "family": "DFT",
        "xc": "b3lyp",
        "disp": None,
        "description": "DFT hybrid B3LYP, umum untuk molekul organik.",
    },
    "B3LYP-D3BJ": {
        "family": "DFT",
        "xc": "b3lyp",
        "disp": "d3bj",
        "description": "B3LYP dengan koreksi dispersi D3BJ; default untuk kandidat hibrida
OTH-01.",
    },
    "PBE0": {
        "family": "DFT",
        "xc": "pbe0",
        "disp": None,
        "description": "DFT hybrid PBE0.",
    },
    "PBE0-D3BJ": {
        "family": "DFT",
        "xc": "pbe0",
        "disp": "d3bj",
        "description": "PBE0 dengan koreksi dispersi D3BJ.",
    },
    "TPSS": {
        "family": "DFT",
        "xc": "tpss",
        "disp": None,
```

```
        "description": "Meta-GGA TPSS.",
    },
    "M06": {
        "family": "DFT",
        "xc": "m06",
        "disp": None,
        "description": "Meta-hybrid Minnesota M06 jika didukung instalasi PySCF/libxc.",
    },
    "CAM-B3LYP": {
        "family": "DFT",
        "xc": "cam-b3lyp",
        "disp": None,
        "description": "Range-separated hybrid CAM-B3LYP.",
    },
    "WB97X-D": {
        "family": "DFT",
        "xc": "wb97x-d",
        "disp": None,
        "description": "Range-separated functional dengan dispersi bawaan jika didukung.",
    },
    "CUSTOM": {
        "family": "DFT",
        "xc": None,
        "disp": None,
        "description": "Functional DFT custom dari argumen --xc.",
    },
}


# ============================================================
# 2. DAFTAR BASIS SET
# ============================================================

AVAILABLE_BASIS_SETS = {
    "STO-3G": "sto-3g",
    "3-21G": "3-21g",
    "6-31G": "6-31g",
    "6-31G*": "6-31g*",
    "6-31G**": "6-31g**",
    "6-31+G*": "6-31+g*",
    "6-31+G**": "6-31+g**",
    "def2-SVP": "def2-svp",
    "def2-TZVP": "def2-tzvp",
    "def2-QZVP": "def2-qzvp",
    "cc-pVDZ": "cc-pvdz",
    "cc-pVTZ": "cc-pvtz",
    "aug-cc-pVDZ": "aug-cc-pvdz",
    "CUSTOM": None,
}


# ============================================================
# 3. KONFIGURASI DEFAULT
# ============================================================

MOLECULE_NAME_DEFAULT = "OTH-01 Olaparib-Talazoparib Hybrid"
FORMULA_DEFAULT = "C24H22FN7O3"

METHOD_DEFAULT = "B3LYP-D3BJ"
BASIS_DEFAULT = "def2-SVP"

CHARGE_DEFAULT = 0
SPIN_DEFAULT = 0

DFT_GRID_LEVEL_DEFAULT = 3
CONV_TOL_DEFAULT = 1e-9
MAX_CYCLE_DEFAULT = 150
VERBOSE_DEFAULT = 4

USE_DENSITY_FITTING_DEFAULT = False
JOB_DEFAULT = "singlepoint"

OUTPUT_ROOT_DEFAULT = "hasil_pyscf"
BOND_SCALE_DEFAULT = 1.25
```

```
BOND_TOLERANCE_DEFAULT = 0.15

AUTHOR_DEFAULT = "Kasmui"
AUTHOR_PHONE_DEFAULT = "0818294312"
SMILES_DEFAULT = "O=C1NN=C(c2ccccc21)Cc3ccc(F)c(c3)C(=O)N4CCN(CC4)C(=O)c5nn(C)cn5"
IUPAC_NAME_DEFAULT = "Olaparib-Talazoparib merged hybrid candidate: phthalazinone-
piperazine-carbonyl-methyltriazole fluorophenyl scaffold"
RESEARCH_NOTE_DEFAULT = (
    "Kandidat hibrida in silico Olaparib-Talazoparib untuk riset inhibitor PARP-1 pada
kanker payudara BRCA-mutated. "
    "Struktur ini adalah kandidat awal komputasi, bukan obat tervalidasi klinis."
)


# ============================================================
# 4. KOORDINAT DEFAULT OTH-01 OLAPARIB-TALAZOPARIB HYBRID
# ============================================================
# Format XYZ standar juga diterima:
# baris 1 = jumlah atom, baris 2 = judul/komentar.
# Program otomatis membuang dua baris tersebut saat membaca koordinat.

DEFAULT_MOLECULE_XYZ = """
57
OTH-01 Olaparib-Talazoparib Hybrid | author: Kasmui | Telp.: 0818294312 | generated from
RDKit MMFF94 conformer
O      -9.17676470      0.41437369     -0.67062951
C      -8.06176180      0.49482581     -0.16753486
N      -7.50673299      1.69531264      0.20115037
N      -6.28131490      1.87477513      0.76091134
C      -5.54269570      0.82653472      0.98468150
C      -5.97939478     -0.55486754      0.65383130
C      -5.19811638     -1.69258147      0.88802322
C      -5.68112436     -2.95865261      0.54139571
C      -6.94247566     -3.09717826     -0.03562814
C      -7.72558316     -1.96832484     -0.26888636
C      -7.23968478     -0.70105794      0.07678584
C      -4.17178725      1.06850580      1.60376597
C      -3.03824647      0.98696543      0.60548772
C      -3.07497902      1.72154253     -0.59068736
C      -2.03175940      1.62826903     -1.51526107
C      -0.93528759      0.81632429     -1.24359438
F       0.03572176      0.71999733     -2.16257007
C      -0.85982639      0.11767181     -0.04291134
C      -1.91401717      0.19247264      0.87718205
C       0.27311029     -0.77498988      0.26021544
O       0.00603014     -1.94341179      0.53732934
N       1.56755358     -0.26864472      0.26492648
C       1.91033087      1.15992845      0.22190315
C       3.23860972      1.47187533      0.92421264
N       4.34355559      0.66544721      0.40635318
C       4.01523165     -0.69500253     -0.03806045
C       2.67112405     -1.19961378      0.50606128
C       5.64133847      1.19994490      0.45503981
O       5.84045831      2.27337185      1.03239277
C       6.75303753      0.46376794     -0.20920260
N       8.05190099      0.75294464      0.01836758
N       8.68585954     -0.09887769     -0.79813534
C      10.12252636     -0.11004365     -0.86555396
C       7.81205748     -0.87106256     -1.48701459
N       6.58696974     -0.54853175     -1.14731906
H      -8.06183634      2.53065827      0.04238987
H      -4.21016294     -1.63285756      1.33239943
H      -5.06782788     -3.84004186      0.72001252
H      -7.31468893     -4.08303557     -0.30509415
H      -8.71039211     -2.07474723     -0.71981209
H      -4.15618629      2.06768938      2.05797733
H      -4.02621436      0.36230053      2.42925773
H      -3.92527178      2.36461362     -0.81750445
H      -2.07388541      2.17846071     -2.45074927
H      -1.85079948     -0.38234106      1.79989477
H       1.98879972      1.43451349     -0.83598133
H       1.10717691      1.75072120      0.67286849
H       3.46528213      2.53843853      0.82111102
H       3.15957237      1.25107044      1.99564180
```

```
H       4.80203965     -1.38614209      0.27829957
H       3.97874494     -0.67078349     -1.13323721
H       2.43549258     -2.17719940      0.07093817
H       2.73676323     -1.33164850      1.59328922
H      10.44751420     -0.86759724     -1.58300446
H      10.51425615     -0.34533850      0.12715758
H      10.46213994      0.87751312     -1.18756512
H       8.10562011     -1.62625694     -2.20260445
"""


# ============================================================
# 5. DATA ATOM DAN JARI-JARI KOVALEN
# ============================================================

COVALENT_RADII_ANGSTROM = {
    "H": 0.31,
    "He": 0.28,
    "Li": 1.28,
    "Be": 0.96,
    "B": 0.84,
    "C": 0.76,
    "N": 0.71,
    "O": 0.66,
    "F": 0.57,
    "Ne": 0.58,
    "Na": 1.66,
    "Mg": 1.41,
    "Al": 1.21,
    "Si": 1.11,
    "P": 1.07,
    "S": 1.05,
    "Cl": 1.02,
    "Ar": 1.06,
    "K": 2.03,
    "Ca": 1.76,
    "Fe": 1.24,
    "Cu": 1.32,
    "Zn": 1.22,
    "Br": 1.20,
    "I": 1.39,
}

ATOMIC_NUMBERS = {
    "H": 1,
    "He": 2,
    "Li": 3,
    "Be": 4,
    "B": 5,
    "C": 6,
    "N": 7,
    "O": 8,
    "F": 9,
    "Ne": 10,
    "Na": 11,
    "Mg": 12,
    "Al": 13,
    "Si": 14,
    "P": 15,
    "S": 16,
    "Cl": 17,
    "Ar": 18,
    "K": 19,
    "Ca": 20,
    "Fe": 26,
    "Cu": 29,
    "Zn": 30,
    "Br": 35,
    "I": 53,
}


# ============================================================
# 6. FUNGSI BANTUAN UMUM
```

```
# ============================================================

def clean_element_symbol(symbol):
    letters = "".join(ch for ch in str(symbol).strip() if ch.isalpha())
    if not letters:
        return ""
    if len(letters) == 1:
        return letters.upper()
    return letters[0].upper() + letters[1:].lower()


def safe_filename(text):
    safe_chars = []
    for char in str(text):
        if char.isalnum() or char in ["-", "_"]:
            safe_chars.append(char)
        elif char in ["*", "+"]:
            safe_chars.append("x")
        else:
            safe_chars.append("_")
    value = "".join(safe_chars).strip("_")
    while "__" in value:
        value = value.replace("__", "_")
    return value or "molecule"


def normalize_choice(user_value, allowed_dict, label):
    if user_value is None:
        return None
    user_compact = str(user_value).strip().lower().replace(" ", "")
    for key in allowed_dict:
        key_compact = key.strip().lower().replace(" ", "")
        if user_compact == key_compact:
            return key
    print(f"\nERROR: {label} '{user_value}' tidak tersedia.")
    print(f"Pilihan {label} yang tersedia:")
    for key in allowed_dict:
        print(f"  - {key}")
    sys.exit(1)


def validate_positive_float(value, label):
    if value <= 0:
        print(f"ERROR: {label} harus bernilai positif.")
        sys.exit(1)


def validate_nonnegative_int(value, label):
    if value < 0:
        print(f"ERROR: {label} harus integer >= 0.")
        sys.exit(1)


def validate_positive_int(value, label):
    if value <= 0:
        print(f"ERROR: {label} harus integer positif.")
        sys.exit(1)


class TeeOutput:
    """
    Menyalin semua teks yang ditulis ke terminal sekaligus ke file log.

    Tujuan:
    - Teks proses running tetap tampil normal di layar.
    - Teks yang sama otomatis tersimpan ke file .log.
    - stdout dan stderr ikut ditangkap, termasuk pesan error PySCF.
    """

    def __init__(self, terminal_stream, log_stream):
        self.terminal_stream = terminal_stream
        self.log_stream = log_stream

    def write(self, message):
```

```
        self.terminal_stream.write(message)
        self.log_stream.write(message)
        self.flush()

    def flush(self):
        self.terminal_stream.flush()
        self.log_stream.flush()

    def isatty(self):
        return getattr(self.terminal_stream, "isatty", lambda: False)()

    def fileno(self):
        return self.terminal_stream.fileno()


def write_running_log_header(log_path, args, method_name, basis_name, pyscf_basis):
    print(identity_header_text())
    print("\n============================================================")
    print("LOG RUNNING OTOMATIS AKTIF")
    print("============================================================")
    print(f"File log running  : {log_path}")
    print(f"Waktu mulai       : {datetime.now().strftime('%Y-%m-%d %H:%M:%S')}")
    print(f"Nama molekul      : {args.name}")
    print(f"Rumus molekul     : {args.formula}")
    print(f"Metode            : {method_name}")
    print(f"Basis set         : {basis_name} ({pyscf_basis})")
    print(f"Jenis perhitungan : {args.job}")
    print("Catatan           : Semua teks terminal stdout/stderr disalin ke file ini.")
    print("============================================================")


def write_running_log_footer(log_path, status="SELESAI"):
    print("\n============================================================")
    print("LOG RUNNING OTOMATIS DITUTUP")
    print("============================================================")
    print(f"Status akhir      : {status}")
    print(f"Waktu selesai     : {datetime.now().strftime('%Y-%m-%d %H:%M:%S')}")
    print(f"File log running  : {log_path}")
    print("============================================================")


# ============================================================
# 7. PEMBACAAN DAN VALIDASI KOORDINAT
# ============================================================

def parse_xyz_like_text(text):
    """
    Membaca koordinat dari:
    1. XYZ standar:
       jumlah_atom
       komentar
       Atom x y z
    2. Blok koordinat langsung:
       Atom x y z
    3. File teks dengan komentar diawali #.
    """

    if not text or not str(text).strip():
        raise ValueError("Blok koordinat kosong.")

    raw_lines = []
    for line in str(text).splitlines():
        stripped = line.strip()
        if not stripped:
            continue
        if stripped.startswith("#"):
            continue
        raw_lines.append(stripped)

    if not raw_lines:
        raise ValueError("Tidak ada baris koordinat yang dapat dibaca.")

    expected_count = None
    coordinate_lines = raw_lines
```

```
    first_parts = raw_lines[0].split()
    if len(first_parts) == 1:
        try:
            expected_count = int(first_parts[0])
            coordinate_lines = raw_lines[2:2 + expected_count]
        except ValueError:
            expected_count = None

    atoms = []
    for number, line in enumerate(coordinate_lines, start=1):
        parts = line.split()
        if len(parts) < 4:
            raise ValueError(
                f"Format koordinat salah pada baris koordinat ke-{number}: '{line}'. "
                "Format benar: SimbolAtom X Y Z"
            )

        symbol = clean_element_symbol(parts[0])
        if symbol not in ATOMIC_NUMBERS:
            raise ValueError(
                f"Simbol atom '{parts[0]}' pada baris ke-{number} tidak dikenali."
            )

        try:
            x = float(parts[1])
            y = float(parts[2])
            z = float(parts[3])
        except ValueError as exc:
            raise ValueError(
                f"Koordinat X/Y/Z harus numerik pada baris ke-{number}: '{line}'."
            ) from exc

        atoms.append({
            "symbol": symbol,
            "x": x,
            "y": y,
            "z": z,
        })

    if expected_count is not None and len(atoms) != expected_count:
        raise ValueError(
            f"Jumlah atom di header XYZ = {expected_count}, tetapi yang terbaca =
{len(atoms)}."
        )

    if not atoms:
        raise ValueError("Tidak ada atom yang berhasil dibaca.")

    return atoms


def read_text_file(path):
    file_path = Path(path)
    if not file_path.exists():
        print(f"ERROR: File koordinat tidak ditemukan: {file_path}")
        sys.exit(1)
    return file_path.read_text(encoding="utf-8")


def format_atom_block(atoms):
    return "\n".join(
        f"{atom['symbol']:2s}  {atom['x']:14.8f}  {atom['y']:14.8f}  {atom['z']:14.8f}"
        for atom in atoms
    )


def get_coordinate_atoms(args):
    if args.xyz_file:
        source_text = read_text_file(args.xyz_file)
        source_label = f"file XYZ: {args.xyz_file}"
    elif args.coord_file:
        source_text = read_text_file(args.coord_file)
        source_label = f"file koordinat: {args.coord_file}"
```

```
    else:
        source_text = DEFAULT_MOLECULE_XYZ
        source_label = "koordinat default di dalam kode"

    try:
        atoms = parse_xyz_like_text(source_text)
    except ValueError as exc:
        print("\nERROR: Koordinat molekul tidak valid.")
        print(str(exc))
        print("\nContoh format benar:")
        print("C   0.000000   0.000000   0.000000")
        print("H   0.000000   0.000000   1.089000")
        sys.exit(1)

    return atoms, source_label


# ============================================================
# 8. RESOLUSI METODE DAN BASIS SET
# ============================================================

def resolve_method(args):
    method_name = normalize_choice(args.method, AVAILABLE_METHODS, "metode")
    info = dict(AVAILABLE_METHODS[method_name])

    if method_name == "CUSTOM":
        if not args.xc or not args.xc.strip():
            print("ERROR: --method CUSTOM wajib disertai --xc, contoh: --xc 'm06-2x'")
            sys.exit(1)
        info["xc"] = args.xc.strip()
        info["description"] = f"DFT custom dengan functional PySCF: {args.xc.strip()}"

    if args.disp:
        info["disp"] = args.disp.strip().lower()

    return method_name, info


def resolve_basis(args):
    basis_name = normalize_choice(args.basis, AVAILABLE_BASIS_SETS, "basis set")

    if basis_name == "CUSTOM":
        if not args.basis_custom or not args.basis_custom.strip():
            print("ERROR: --basis CUSTOM wajib disertai --basis-custom, contoh: --basis-
custom 'def2-tzvpp'")
            sys.exit(1)
        pyscf_basis = args.basis_custom.strip()
    else:
        pyscf_basis = AVAILABLE_BASIS_SETS[basis_name]

    return basis_name, pyscf_basis


# ============================================================
# 9. MEMBANGUN OBJEK MOLEKUL DAN KALKULATOR
# ============================================================

def build_molecule(atom_block, pyscf_basis, charge, spin, verbose):
    mol = gto.M(
        atom=atom_block,
        basis=pyscf_basis,
        unit="Angstrom",
        charge=charge,
        spin=spin,
        verbose=verbose,
    )
    return mol


def build_calculator(mol, method_name, method_info, grid_level, conv_tol, max_cycle,
use_density_fitting):
    family = method_info["family"]

    if family == "HF":
```

```
        mf = scf.RHF(mol) if mol.spin == 0 else scf.UHF(mol)
    elif family == "DFT":
        mf = dft.RKS(mol) if mol.spin == 0 else dft.UKS(mol)
        mf.xc = method_info["xc"]
        mf.grids.level = grid_level
    else:
        raise ValueError(f"Family metode tidak dikenal: {family}")

    if method_info.get("disp"):
        # PySCF modern mendukung atribut mf.disp untuk D3/D4 jika ekstensi tersedia.
        mf.disp = method_info["disp"]

    if use_density_fitting:
        mf = mf.density_fit()

    mf.conv_tol = conv_tol
    mf.max_cycle = max_cycle
    mf.chkfile = safe_filename(f"{MOLECULE_NAME}_{method_name}_{BASIS_LABEL}.chk")

    return mf


# ============================================================
# 10. OPTIMASI GEOMETRI OPSIONAL
# ============================================================

def optimize_geometry(mf):
    try:
        from pyscf.geomopt.geometric_solver import optimize
    except ImportError:
        print("\nERROR: Optimasi geometri memerlukan paket geomeTRIC.")
        print("Instal terlebih dahulu:")
        print("  pip install geometric")
        sys.exit(1)

    print("\n============================================================")
    print("OPTIMASI GEOMETRI DIMULAI")
    print("============================================================")
    mol_opt = optimize(mf)
    print("\n============================================================")
    print("OPTIMASI GEOMETRI SELESAI")
    print("============================================================")
    return mol_opt


# ============================================================
# 11. INFORMASI DAN PELAPORAN DASAR
# ============================================================

def print_available_options():
    print("\n============================================================")
    print("DAFTAR METODE KOMPUTASI")
    print("============================================================")
    for method_name, info in AVAILABLE_METHODS.items():
        xc_text = info["xc"] if info["xc"] else "-"
        disp_text = info["disp"] if info["disp"] else "-"
        print(f"{method_name:12s} | family={info['family']:3s} | xc={xc_text:12s} |
disp={disp_text:5s} | {info['description']}")

    print("\n============================================================")
    print("DAFTAR BASIS SET")
    print("============================================================")
    for basis_name, pyscf_name in AVAILABLE_BASIS_SETS.items():
        print(f"{basis_name:12s} : {pyscf_name if pyscf_name else 'isi melalui --basis-
custom'}")

    print("\n============================================================")
    print("CONTOH PERINTAH")
    print("============================================================")
    print("python oth01-olaparib-talazoparib-hybrid.py")
    print("python oth01-olaparib-talazoparib-hybrid.py --method B3LYP --basis def2-SVP")
    print("python oth01-olaparib-talazoparib-hybrid.py --xyz-file molekul.xyz --name
'Molekul Baru' --method PBE0 --basis def2-TZVP")
```

```
    print("python oth01-olaparib-talazoparib-hybrid.py --method CUSTOM --xc 'm06-2x' --
basis CUSTOM --basis-custom 'def2-tzvpp'")
    print("python oth01-olaparib-talazoparib-hybrid.py --job optimize --method B3LYP --
basis def2-SVP")

    print("\n============================================================")
    print("CONTOH FORMAT KOORDINAT")
    print("============================================================")
    print("3")
    print("Water")
    print("O   0.000000   0.000000   0.000000")
    print("H   0.000000  -0.757000   0.587000")
    print("H   0.000000   0.757000   0.587000")
    print()


def print_molecule_info(mol, method_name, method_info, basis_name, pyscf_basis, charge,
spin, job, coordinate_source):
    print("\n============================================================")
    print("INFORMASI SISTEM")
    print("============================================================")
    print(f"Nama molekul        : {MOLECULE_NAME}")
    print(f"Rumus molekul       : {MOLECULE_FORMULA}")
    print(f"Sumber koordinat    : {coordinate_source}")
    print(f"Jumlah atom         : {mol.natm}")
    print(f"Jumlah elektron     : {mol.nelectron}")
    print(f"Jumlah fungsi basis : {mol.nao_nr()}")
    print(f"Muatan total        : {charge}")
    print(f"Spin PySCF          : {spin}")
    print(f"Metode komputasi    : {method_name}")
    print(f"Family metode       : {method_info['family']}")
    print(f"XC functional       : {method_info.get('xc')}")
    print(f"Dispersion          : {method_info.get('disp')}")
    print(f"Basis set           : {basis_name} ({pyscf_basis})")
    print(f"Jenis perhitungan   : {job}")
    print("Satuan koordinat    : Angstrom")
    print("============================================================")


def print_coordinates(mol, title):
    print("\n============================================================")
    print(title)
    print("============================================================")
    coords = mol.atom_coords(unit="Angstrom")
    for i in range(mol.natm):
        symbol = clean_element_symbol(mol.atom_symbol(i))
        x, y, z = coords[i]
        print(f"{symbol:2s}  {x:14.8f}  {y:14.8f}  {z:14.8f}")


# ============================================================
# 12. ANALISIS STRUKTUR AKHIR
# ============================================================

def get_atom_records(mol):
    coords = mol.atom_coords(unit="Angstrom")
    atoms = []
    for i in range(mol.natm):
        symbol = clean_element_symbol(mol.atom_symbol(i))
        atomic_number = ATOMIC_NUMBERS.get(symbol)
        try:
            atomic_number = int(mol.atom_charge(i))
        except Exception:
            pass
        x, y, z = coords[i]
        atoms.append({
            "index": i + 1,
            "symbol": symbol,
            "atomic_number": atomic_number,
            "x": float(x),
            "y": float(y),
            "z": float(z),
        })
    return atoms
```

```
def distance_between_atoms(atom_a, atom_b):
    dx = atom_a["x"] - atom_b["x"]
    dy = atom_a["y"] - atom_b["y"]
    dz = atom_a["z"] - atom_b["z"]
    return math.sqrt(dx * dx + dy * dy + dz * dz)


def vector_between_atoms(atom_a, atom_b):
    return [
        atom_b["x"] - atom_a["x"],
        atom_b["y"] - atom_a["y"],
        atom_b["z"] - atom_a["z"],
    ]


def dot_product(vec_a, vec_b):
    return vec_a[0] * vec_b[0] + vec_a[1] * vec_b[1] + vec_a[2] * vec_b[2]


def cross_product(vec_a, vec_b):
    return [
        vec_a[1] * vec_b[2] - vec_a[2] * vec_b[1],
        vec_a[2] * vec_b[0] - vec_a[0] * vec_b[2],
        vec_a[0] * vec_b[1] - vec_a[1] * vec_b[0],
    ]


def vector_norm(vec):
    return math.sqrt(dot_product(vec, vec))


def normalize_vector(vec):
    norm = vector_norm(vec)
    if norm < 1e-15:
        return [0.0, 0.0, 0.0]
    return [value / norm for value in vec]


def calculate_angle_degrees(atom_i, atom_j, atom_k):
    vec_ji = vector_between_atoms(atom_j, atom_i)
    vec_jk = vector_between_atoms(atom_j, atom_k)
    norm_ji = vector_norm(vec_ji)
    norm_jk = vector_norm(vec_jk)
    if norm_ji < 1e-15 or norm_jk < 1e-15:
        return None
    cos_theta = dot_product(vec_ji, vec_jk) / (norm_ji * norm_jk)
    cos_theta = max(-1.0, min(1.0, cos_theta))
    return math.degrees(math.acos(cos_theta))


def calculate_dihedral_degrees(atom_i, atom_j, atom_k, atom_l):
    p0 = [atom_i["x"], atom_i["y"], atom_i["z"]]
    p1 = [atom_j["x"], atom_j["y"], atom_j["z"]]
    p2 = [atom_k["x"], atom_k["y"], atom_k["z"]]
    p3 = [atom_l["x"], atom_l["y"], atom_l["z"]]

    b0 = [p0[n] - p1[n] for n in range(3)]
    b1 = [p2[n] - p1[n] for n in range(3)]
    b2 = [p3[n] - p2[n] for n in range(3)]
    b1_unit = normalize_vector(b1)

    v = [b0[n] - dot_product(b0, b1_unit) * b1_unit[n] for n in range(3)]
    w = [b2[n] - dot_product(b2, b1_unit) * b1_unit[n] for n in range(3)]

    if vector_norm(v) < 1e-15 or vector_norm(w) < 1e-15:
        return None

    x_value = dot_product(v, w)
    y_value = dot_product(cross_product(b1_unit, v), w)
    return math.degrees(math.atan2(y_value, x_value))
```

```
def infer_bonds(atoms, bond_scale=BOND_SCALE_DEFAULT,
bond_tolerance=BOND_TOLERANCE_DEFAULT):
    bonds = []
    for i in range(len(atoms)):
        for j in range(i + 1, len(atoms)):
            atom_i = atoms[i]
            atom_j = atoms[j]
            symbol_i = atom_i["symbol"]
            symbol_j = atom_j["symbol"]
            radius_i = COVALENT_RADII_ANGSTROM.get(symbol_i, 0.77)
            radius_j = COVALENT_RADII_ANGSTROM.get(symbol_j, 0.77)
            distance = distance_between_atoms(atom_i, atom_j)
            threshold = bond_scale * (radius_i + radius_j) + bond_tolerance
            if symbol_i == "H" and symbol_j == "H" and distance > 0.90:
                continue
            if distance <= threshold:
                bonds.append({
                    "atom1": atom_i["index"],
                    "atom2": atom_j["index"],
                    "symbol1": symbol_i,
                    "symbol2": symbol_j,
                    "distance": float(distance),
                    "order": 1,
                })
    return bonds


def build_adjacency(atoms, bonds):
    adjacency = {atom["index"]: [] for atom in atoms}
    for bond in bonds:
        atom1 = bond["atom1"]
        atom2 = bond["atom2"]
        adjacency[atom1].append(atom2)
        adjacency[atom2].append(atom1)
    for atom_index in adjacency:
        adjacency[atom_index] = sorted(adjacency[atom_index])
    return adjacency


def infer_angles(atoms, bonds):
    atom_by_index = {atom["index"]: atom for atom in atoms}
    adjacency = build_adjacency(atoms, bonds)
    angles = []
    for center in sorted(adjacency):
        neighbors = adjacency[center]
        for a in range(len(neighbors)):
            for b in range(a + 1, len(neighbors)):
                atom_i_index = neighbors[a]
                atom_k_index = neighbors[b]
                angle_value = calculate_angle_degrees(
                    atom_by_index[atom_i_index],
                    atom_by_index[center],
                    atom_by_index[atom_k_index],
                )
                if angle_value is None:
                    continue
                angles.append({
                    "atom1": atom_i_index,
                    "atom2": center,
                    "atom3": atom_k_index,
                    "symbol1": atom_by_index[atom_i_index]["symbol"],
                    "symbol2": atom_by_index[center]["symbol"],
                    "symbol3": atom_by_index[atom_k_index]["symbol"],
                    "angle_degrees": float(angle_value),
                })
    return angles


def infer_dihedrals(atoms, bonds):
    atom_by_index = {atom["index"]: atom for atom in atoms}
    adjacency = build_adjacency(atoms, bonds)
    seen = set()
    dihedrals = []
    for bond in bonds:
```

```
        j = bond["atom1"]
        k = bond["atom2"]
        for i in adjacency[j]:
            if i == k:
                continue
            for l in adjacency[k]:
                if l == j:
                    continue
                dihedral_tuple = (i, j, k, l)
                reverse_tuple = tuple(reversed(dihedral_tuple))
                canonical = min(dihedral_tuple, reverse_tuple)
                if canonical in seen:
                    continue
                seen.add(canonical)
                dihedral_value = calculate_dihedral_degrees(
                    atom_by_index[i],
                    atom_by_index[j],
                    atom_by_index[k],
                    atom_by_index[l],
                )
                if dihedral_value is None:
                    continue
                dihedrals.append({
                    "atom1": i,
                    "atom2": j,
                    "atom3": k,
                    "atom4": l,
                    "symbol1": atom_by_index[i]["symbol"],
                    "symbol2": atom_by_index[j]["symbol"],
                    "symbol3": atom_by_index[k]["symbol"],
                    "symbol4": atom_by_index[l]["symbol"],
                    "dihedral_degrees": float(dihedral_value),
                })
    return dihedrals


def calculate_distance_matrix(atoms):
    return [
        [float(distance_between_atoms(atom_i, atom_j)) for atom_j in atoms]
        for atom_i in atoms
    ]


# ============================================================
# 13. PENULISAN FILE HASIL
# ============================================================

def make_output_dir(args, method_name, basis_name):
    timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
    folder_name = safe_filename(

f"{MOLECULE_NAME}_{MOLECULE_FORMULA}_{method_name}_{basis_name}_{args.job}_{timestamp}"
    )
    if args.output_dir:
        output_path = Path(args.output_dir)
    else:
        output_path = Path(OUTPUT_ROOT_DEFAULT) / folder_name
    output_path.mkdir(parents=True, exist_ok=True)
    return output_path


def make_output_prefix(method_name, basis_name, job):
    timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
    return
safe_filename(f"{MOLECULE_NAME}_{MOLECULE_FORMULA}_{method_name}_{basis_name}_{job}_{times
tamp}")


def format_xyz_coordinate_block(atoms):
    return "\n".join(
        f"{atom['symbol']:2s}  {atom['x']:14.8f}  {atom['y']:14.8f}  {atom['z']:14.8f}"
        for atom in atoms
    )
```

```
def identity_header_lines():
    return [
        f"author: {AUTHOR_DEFAULT}",
        f"Telp.: {AUTHOR_PHONE_DEFAULT}",
    ]


def identity_header_text():
    return "\n".join(identity_header_lines())


def write_txt_coordinates(path, atoms, metadata):
    lines = identity_header_lines() + [""]
    lines.append("KOORDINAT AKHIR MOLEKUL")
    lines.append("=" * 60)
    for key, value in metadata.items():
        lines.append(f"{key}: {value}")
    lines.append("=" * 60)
    lines.append("Format: SimbolAtom        X              Y              Z")
    lines.append("Satuan: Angstrom")
    lines.append("")
    lines.append(format_xyz_coordinate_block(atoms))
    lines.append("")
    lines.append("BLOK KOORDINAT SIAP SALIN UNTUK PySCF/XYZ")
    lines.append("=" * 60)
    lines.append('\"\"\"')
    lines.append(format_xyz_coordinate_block(atoms))
    lines.append('\"\"\"')
    lines.append("")
    Path(path).write_text("\n".join(lines), encoding="utf-8")


def write_xyz_file(path, atoms, metadata):
    title = (
        f"{metadata['Nama molekul']} | author: {AUTHOR_DEFAULT} | Telp.:
{AUTHOR_PHONE_DEFAULT} | "
        f"{metadata['Metode']} / {metadata['Basis set']} | "
        f"E = {metadata['Energi total Hartree']} Hartree | "
        f"SCF converged = {metadata['SCF konvergen']}"
    )
    lines = [str(len(atoms)), title, format_xyz_coordinate_block(atoms)]
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_mol_file(path, atoms, bonds, title=None):
    title = title or MOLECULE_NAME
    lines = []
    lines.append(title[:80])
    lines.append(f"  author: {AUTHOR_DEFAULT} | Telp.: {AUTHOR_PHONE_DEFAULT}")
    lines.append("  Generated final 3D coordinates and inferred bonds by PySCF Python
Script")
    lines.append(f"{len(atoms):3d}{len(bonds):3d}  0  0  0  0            999 V2000")

    for atom in atoms:
        lines.append(
            f"{atom['x']:10.4f}"
            f"{atom['y']:10.4f}"
            f"{atom['z']:10.4f} "
            f"{atom['symbol']:<3s}"
            " 0  0  0  0  0  0  0  0  0  0  0  0"
        )

    for bond in bonds:
        lines.append(
            f"{bond['atom1']:3d}"
            f"{bond['atom2']:3d}"
            f"{int(bond.get('order', 1)):3d}"
            "  0  0  0  0"
        )

    lines.append("M  END")
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")
```

```
def write_sdf_file(path, atoms, bonds, metadata):
    mol_text_path = Path(path).with_suffix(".tmp_mol")
    write_mol_file(mol_text_path, atoms, bonds, title=metadata["Nama molekul"])
    mol_text = mol_text_path.read_text(encoding="utf-8")
    mol_text_path.unlink(missing_ok=True)

    lines = [mol_text.rstrip("\n")]
    for key, value in metadata.items():
        lines.append(f">  <{key}>")
        lines.append(str(value))
        lines.append("")
    lines.append("$$$$")
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_pdb_file(path, atoms, bonds, metadata):
    lines = []
    lines.append(f"HEADER    {metadata['Nama molekul'][:50]}")
    lines.append(f"REMARK    author: {AUTHOR_DEFAULT}")
    lines.append(f"REMARK    Telp.: {AUTHOR_PHONE_DEFAULT}")
    lines.append(f"TITLE     {metadata['Metode']} / {metadata['Basis set']}")
    lines.append(f"REMARK    Energy Hartree: {metadata['Energi total Hartree']}")
    lines.append(f"REMARK    SCF converged: {metadata['SCF konvergen']}")

    for atom in atoms:
        atom_index = atom["index"]
        symbol = atom["symbol"]
        lines.append(
            f"HETATM{atom_index:5d} "
            f"{symbol:<4s}"
            f" MOL     1    "
            f"{atom['x']:8.3f}"
            f"{atom['y']:8.3f}"
            f"{atom['z']:8.3f}"
            f"  1.00  0.00          "
            f"{symbol:>2s}"
        )

    adjacency = build_adjacency(atoms, bonds)
    for atom_index in sorted(adjacency):
        if not adjacency[atom_index]:
            continue
        conect_targets = "".join(f"{target:5d}" for target in adjacency[atom_index])
        lines.append(f"CONECT{atom_index:5d}{conect_targets}")

    lines.append("END")
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_cml_file(path, atoms, bonds, metadata):
    lines = []
    lines.append('<?xml version="1.0" encoding="UTF-8"?>')
    lines.append('<cml xmlns="http://www.xml-cml.org/schema">')
    lines.append(f'  <molecule id="{escape(safe_filename(metadata["Nama molekul"]))}">')
    lines.append("    <metadataList>")
    for key, value in metadata.items():
        lines.append(f'      <metadata name="{escape(str(key))}"
content="{escape(str(value))}"/>')
    lines.append("    </metadataList>")
    lines.append("    <atomArray>")
    for atom in atoms:
        lines.append(
            f'      <atom id="a{atom["index"]}" '
            f'elementType="{escape(atom["symbol"])}" '
            f'x3="{atom["x"]:.8f}" '
            f'y3="{atom["y"]:.8f}" '
            f'z3="{atom["z"]:.8f}"/>'
        )
    lines.append("    </atomArray>")
    lines.append("    <bondArray>")
    for bond in bonds:
        lines.append(
            f'      <bond atomRefs2="a{bond["atom1"]} a{bond["atom2"]}" '
```

```
            f'order="{int(bond.get("order", 1))}"/>'
        )
    lines.append("    </bondArray>")
    lines.append("  </molecule>")
    lines.append("</cml>")
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_json_structure_file(path, atoms, bonds, angles, dihedrals, distance_matrix,
metadata):
    structure_data = {
        "metadata": metadata,
        "atoms": atoms,
        "bonds": bonds,
        "angles": angles,
        "dihedrals": dihedrals,
        "distance_matrix_angstrom": distance_matrix,
        "notes": {
            "bond_inference": (
                "Ikatan diinferensi dari jarak antaratom dan jari-jari kovalen. "
                "Orde ikatan ditulis 1 untuk tujuan visualisasi struktur 3D."
            ),
            "indexing": "Semua indeks atom memakai sistem 1-based.",
        },
    }
    Path(path).write_text(
        json.dumps(structure_data, indent=2, ensure_ascii=False),
        encoding="utf-8",
    )


def write_log_file(path, atoms, bonds, angles, dihedrals, distance_matrix, metadata,
output_files):
    lines = identity_header_lines() + [""]
    lines.append("=" * 80)
    lines.append("LAPORAN AKHIR PERHITUNGAN KIMIA KOMPUTASI PySCF")
    lines.append("=" * 80)
    for key, value in metadata.items():
        lines.append(f"{key:28s}: {value}")

    lines.append("")
    lines.append("=" * 80)
    lines.append("KOORDINAT AKHIR MOLEKUL")
    lines.append("=" * 80)
    lines.append("No.  Atom            X              Y              Z       Satuan")
    lines.append("-" * 80)
    for atom in atoms:
        lines.append(
            f"{atom['index']:3d}  {atom['symbol']:2s}  "
            f"{atom['x']:14.8f}  {atom['y']:14.8f}  {atom['z']:14.8f}  Angstrom"
        )

    lines.append("")
    lines.append("=" * 80)
    lines.append("DAFTAR IKATAN HASIL INFERENSI")
    lines.append("=" * 80)
    lines.append("No.  Atom-1       Atom-2       Orde    Jarak (Angstrom)")
    lines.append("-" * 80)
    if bonds:
        for number, bond in enumerate(bonds, start=1):
            lines.append(
                f"{number:3d}  "
                f"{bond['atom1']:3d}-{bond['symbol1']:<2s}      "
                f"{bond['atom2']:3d}-{bond['symbol2']:<2s}      "
                f"{int(bond.get('order', 1)):3d}     "
                f"{bond['distance']:12.6f}"
            )
    else:
        lines.append("Tidak ada ikatan terdeteksi. Periksa --bond-scale atau --bond-
tolerance.")

    lines.append("")
    lines.append("=" * 80)
    lines.append("DAFTAR SUDUT IKATAN")
```

```
    lines.append("=" * 80)
    lines.append("No.  Atom-1       Atom-2/Pusat Atom-3       Sudut (derajat)")
    lines.append("-" * 80)
    if angles:
        for number, angle in enumerate(angles, start=1):
            lines.append(
                f"{number:3d}  "
                f"{angle['atom1']:3d}-{angle['symbol1']:<2s}      "
                f"{angle['atom2']:3d}-{angle['symbol2']:<2s}        "
                f"{angle['atom3']:3d}-{angle['symbol3']:<2s}      "
                f"{angle['angle_degrees']:12.6f}"
            )
    else:
        lines.append("Tidak ada sudut ikatan terdeteksi.")

    lines.append("")
    lines.append("=" * 80)
    lines.append("DAFTAR SUDUT DIHEDRAL")
    lines.append("=" * 80)
    lines.append("No.  Atom-1       Atom-2       Atom-3       Atom-4       Dihedral
(derajat)")
    lines.append("-" * 80)
    if dihedrals:
        for number, dihedral in enumerate(dihedrals, start=1):
            lines.append(
                f"{number:3d}  "
                f"{dihedral['atom1']:3d}-{dihedral['symbol1']:<2s}      "
                f"{dihedral['atom2']:3d}-{dihedral['symbol2']:<2s}      "
                f"{dihedral['atom3']:3d}-{dihedral['symbol3']:<2s}      "
                f"{dihedral['atom4']:3d}-{dihedral['symbol4']:<2s}      "
                f"{dihedral['dihedral_degrees']:12.6f}"
            )
    else:
        lines.append("Tidak ada sudut dihedral terdeteksi.")

    lines.append("")
    lines.append("=" * 80)
    lines.append("MATRIKS JARAK ANTARATOM (Angstrom)")
    lines.append("=" * 80)
    header = "Atom ".ljust(8) + "".join(f"{atom['index']:>10d}" for atom in atoms)
    lines.append(header)
    lines.append("-" * max(80, len(header)))
    for atom, row in zip(atoms, distance_matrix):
        row_text = f"{atom['index']:3d}-{atom['symbol']:<2s} ".ljust(8)
        row_text += "".join(f"{value:10.4f}" for value in row)
        lines.append(row_text)

    lines.append("")
    lines.append("=" * 80)
    lines.append("FILE HASIL YANG DIBUAT")
    lines.append("=" * 80)
    for label, filename in output_files.items():
        lines.append(f"{label:18s}: {filename}")

    lines.append("")
    lines.append("CATATAN")
    lines.append("- Format .xyz hanya menyimpan koordinat, bukan konektivitas ikatan.")
    lines.append("- Format .mol, .sdf, .pdb, dan .cml lebih tepat untuk visualisasi
struktur 3D.")
    lines.append("- Ikatan diinferensi dari jarak antaratom dan jari-jari kovalen.")
    lines.append("- Jika konektivitas visual belum sesuai, ubah --bond-scale atau --bond-
tolerance.")
    lines.append("")

    Path(path).write_text("\n".join(lines), encoding="utf-8")


def save_final_structure_outputs(mol, mf, energy, method_name, method_info, basis_name,
pyscf_basis, args, output_dir, running_output_log_path=None):
    atoms = get_atom_records(mol)
    bonds = infer_bonds(atoms, bond_scale=args.bond_scale,
bond_tolerance=args.bond_tolerance)
    angles = infer_angles(atoms, bonds)
    dihedrals = infer_dihedrals(atoms, bonds)
```

```
    distance_matrix = calculate_distance_matrix(atoms)

    prefix = make_output_prefix(method_name, basis_name, args.job)

    metadata = {
        "author": AUTHOR_DEFAULT,
        "Telp.": AUTHOR_PHONE_DEFAULT,
        "Nama molekul": MOLECULE_NAME,
        "Rumus molekul": MOLECULE_FORMULA,
        "SMILES": SMILES_DEFAULT,
        "IUPAC-like name": IUPAC_NAME_DEFAULT,
        "Catatan riset": RESEARCH_NOTE_DEFAULT,
        "Jumlah atom": mol.natm,
        "Jumlah elektron": mol.nelectron,
        "Muatan total": args.charge,
        "Spin PySCF": args.spin,
        "Metode": method_name,
        "Family metode": method_info["family"],
        "XC functional": method_info.get("xc"),
        "Dispersion": method_info.get("disp"),
        "Basis set": basis_name,
        "Basis PySCF": pyscf_basis,
        "Jenis perhitungan": args.job,
        "Energi total Hartree": f"{energy:.12f}",
        "SCF konvergen": bool(getattr(mf, "converged", False)),
        "File checkpoint": getattr(mf, "chkfile", ""),
        "Satuan koordinat": "Angstrom",
        "Bond scale": args.bond_scale,
        "Bond tolerance": args.bond_tolerance,
        "Waktu simpan": datetime.now().strftime("%Y-%m-%d %H:%M:%S"),
    }

    output_files = {
        "txt_coordinates": str(output_dir / f"{prefix}_koordinat_akhir.txt"),
        "xyz_structure": str(output_dir / f"{prefix}.xyz"),
        "mol_structure": str(output_dir / f"{prefix}.mol"),
        "sdf_structure": str(output_dir / f"{prefix}.sdf"),
        "pdb_structure": str(output_dir / f"{prefix}.pdb"),
        "cml_structure": str(output_dir / f"{prefix}.cml"),
        "json_structure": str(output_dir / f"{prefix}_struktur_lengkap.json"),
        "log_report": str(output_dir / f"{prefix}.log"),
    }

    if running_output_log_path:
        output_files["running_output_log"] = str(running_output_log_path)

    write_txt_coordinates(output_files["txt_coordinates"], atoms, metadata)
    write_xyz_file(output_files["xyz_structure"], atoms, metadata)
    write_mol_file(output_files["mol_structure"], atoms, bonds, title=MOLECULE_NAME)
    write_sdf_file(output_files["sdf_structure"], atoms, bonds, metadata)
    write_pdb_file(output_files["pdb_structure"], atoms, bonds, metadata)
    write_cml_file(output_files["cml_structure"], atoms, bonds, metadata)
    write_json_structure_file(output_files["json_structure"], atoms, bonds, angles,
dihedrals, distance_matrix, metadata)
    write_log_file(output_files["log_report"], atoms, bonds, angles, dihedrals,
distance_matrix, metadata, output_files)

    return output_files, {
        "atoms": atoms,
        "bonds": bonds,
        "angles": angles,
        "dihedrals": dihedrals,
    }


# ============================================================
# 14. INFORMASI ORBITAL
# ============================================================

def print_orbital_info(mf):
    mo_energy = getattr(mf, "mo_energy", None)
    mo_occ = getattr(mf, "mo_occ", None)

    if mo_energy is None or mo_occ is None:
```

```
        print("\nInformasi orbital tidak tersedia.")
        return

    if isinstance(mo_energy, tuple) or isinstance(mo_occ, tuple):
        print("\nSistem terbuka-shell terdeteksi.")
        print("Pelaporan HOMO-LUMO sederhana dilewati.")
        return

    occupied = [i for i, occ in enumerate(mo_occ) if occ > 0]
    virtual = [i for i, occ in enumerate(mo_occ) if occ == 0]

    if not occupied or not virtual:
        print("\nHOMO/LUMO tidak dapat ditentukan dari okupansi orbital.")
        return

    homo_index = max(occupied)
    lumo_index = min(virtual)

    homo_hartree = mo_energy[homo_index]
    lumo_hartree = mo_energy[lumo_index]
    gap_hartree = lumo_hartree - homo_hartree

    hartree_to_ev = 27.211386245988
    homo_ev = homo_hartree * hartree_to_ev
    lumo_ev = lumo_hartree * hartree_to_ev
    gap_ev = gap_hartree * hartree_to_ev

    print("\n============================================================")
    print("INFORMASI ORBITAL SEDERHANA")
    print("============================================================")
    print(f"HOMO index          : {homo_index}")
    print(f"LUMO index          : {lumo_index}")
    print(f"Energi HOMO         : {homo_hartree: .10f} Hartree")
    print(f"Energi LUMO         : {lumo_hartree: .10f} Hartree")
    print(f"Gap HOMO-LUMO       : {gap_hartree: .10f} Hartree")
    print(f"Energi HOMO         : {homo_ev: .6f} eV")
    print(f"Energi LUMO         : {lumo_ev: .6f} eV")
    print(f"Gap HOMO-LUMO       : {gap_ev: .6f} eV")
    print("============================================================")


# ============================================================
# 15. ARGUMEN TERMINAL
# ============================================================

def parse_arguments():
    parser = argparse.ArgumentParser(
        description="Template generik perhitungan kimia komputasi molekul dengan PySCF.
Default: OTH-01 Olaparib-Talazoparib Hybrid."
    )

    parser.add_argument("--name", default=MOLECULE_NAME_DEFAULT, help="Nama molekul.")
    parser.add_argument("--formula", default=FORMULA_DEFAULT, help="Rumus molekul
opsional.")

    parser.add_argument("--xyz-file", default=None, help="File XYZ standar atau blok
koordinat XYZ.")
    parser.add_argument("--coord-file", default=None, help="File teks koordinat:
SimbolAtom X Y Z.")

    parser.add_argument("--method", default=METHOD_DEFAULT, help=f"Metode komputasi.
Default: {METHOD_DEFAULT}")
    parser.add_argument("--xc", default=None, help="Functional PySCF untuk --method
CUSTOM, contoh: 'm06-2x'.")
    parser.add_argument("--disp", default=None, help="Koreksi dispersi manual, contoh:
d3bj, d3zero, d4.")

    parser.add_argument("--basis", default=BASIS_DEFAULT, help=f"Basis set. Default:
{BASIS_DEFAULT}")
    parser.add_argument("--basis-custom", default=None, help="Nama basis PySCF untuk --
basis CUSTOM.")

    parser.add_argument("--charge", type=int, default=CHARGE_DEFAULT, help=f"Muatan total
molekul. Default: {CHARGE_DEFAULT}")
```

```
    parser.add_argument(
        "--spin",
        type=int,
        default=SPIN_DEFAULT,
        help=f"Spin PySCF = jumlah elektron alfa - beta. Default: {SPIN_DEFAULT}",
    )

    parser.add_argument(
        "--job",
        choices=["singlepoint", "optimize"],
        default=JOB_DEFAULT,
        help=f"Jenis perhitungan: singlepoint atau optimize. Default: {JOB_DEFAULT}",
    )
    parser.add_argument("--optimize", action="store_true", help="Alias lama untuk --job
optimize.")

    parser.add_argument("--grid", type=int, default=DFT_GRID_LEVEL_DEFAULT, help=f"Level
grid DFT. Default: {DFT_GRID_LEVEL_DEFAULT}")
    parser.add_argument("--conv-tol", type=float, default=CONV_TOL_DEFAULT,
help=f"Toleransi konvergensi SCF. Default: {CONV_TOL_DEFAULT}")
    parser.add_argument("--max-cycle", type=int, default=MAX_CYCLE_DEFAULT,
help=f"Maksimum iterasi SCF. Default: {MAX_CYCLE_DEFAULT}")
    parser.add_argument("--verbose", type=int, default=VERBOSE_DEFAULT, help=f"Level
output PySCF. Default: {VERBOSE_DEFAULT}")
    parser.add_argument("--density-fit", action="store_true",
default=USE_DENSITY_FITTING_DEFAULT, help="Aktifkan density fitting.")

    parser.add_argument("--output-dir", default=None, help="Direktori output. Jika kosong,
dibuat otomatis di hasil_pyscf/.")
    parser.add_argument("--bond-scale", type=float, default=BOND_SCALE_DEFAULT,
help=f"Faktor skala inferensi ikatan. Default: {BOND_SCALE_DEFAULT}")
    parser.add_argument("--bond-tolerance", type=float, default=BOND_TOLERANCE_DEFAULT,
help=f"Toleransi inferensi ikatan dalam Angstrom. Default: {BOND_TOLERANCE_DEFAULT}")

    parser.add_argument("--list", action="store_true", help="Tampilkan daftar metode,
basis set, dan contoh pemakaian.")

    return parser.parse_args()


# ============================================================
# 16. PROGRAM UTAMA
# ============================================================

def main():
    global MOLECULE_NAME, MOLECULE_FORMULA, BASIS_LABEL

    args = parse_arguments()

    if args.list:
        print_available_options()
        return

    if args.optimize:
        args.job = "optimize"

    validate_nonnegative_int(args.grid, "grid")
    validate_positive_float(args.conv_tol, "conv_tol")
    validate_positive_int(args.max_cycle, "max_cycle")
    validate_positive_float(args.bond_scale, "bond_scale")
    validate_nonnegative_int(args.verbose, "verbose")

    MOLECULE_NAME = args.name.strip() if args.name else MOLECULE_NAME_DEFAULT
    MOLECULE_FORMULA = args.formula.strip() if args.formula else "-"

    method_name, method_info = resolve_method(args)
    basis_name, pyscf_basis = resolve_basis(args)
    BASIS_LABEL = basis_name

    # Direktori output dan file running log dibuat sebelum PySCF mulai bekerja,
    # sehingga pesan proses SCF/DFT dan pesan error tetap tersimpan walaupun perhitungan
gagal.
    output_dir = make_output_dir(args, method_name, basis_name)
    running_log_prefix = make_output_prefix(method_name, basis_name, args.job)
```

```
    running_output_log_path = output_dir / f"{running_log_prefix}_running_output.log"

    original_stdout = sys.stdout
    original_stderr = sys.stderr
    status_akhir = "SELESAI"

    with open(running_output_log_path, "w", encoding="utf-8", buffering=1) as
running_log_file:
        sys.stdout = TeeOutput(original_stdout, running_log_file)
        sys.stderr = TeeOutput(original_stderr, running_log_file)
        try:
            write_running_log_header(
                log_path=running_output_log_path,
                args=args,
                method_name=method_name,
                basis_name=basis_name,
                pyscf_basis=pyscf_basis,
            )

            coordinate_atoms, coordinate_source = get_coordinate_atoms(args)
            atom_block = format_atom_block(coordinate_atoms)

            mol = build_molecule(
                atom_block=atom_block,
                pyscf_basis=pyscf_basis,
                charge=args.charge,
                spin=args.spin,
                verbose=args.verbose,
            )

            print_molecule_info(
                mol=mol,
                method_name=method_name,
                method_info=method_info,
                basis_name=basis_name,
                pyscf_basis=pyscf_basis,
                charge=args.charge,
                spin=args.spin,
                job=args.job,
                coordinate_source=coordinate_source,
            )

            print_coordinates(mol, "KOORDINAT AWAL MOLEKUL")

            mf = build_calculator(
                mol=mol,
                method_name=method_name,
                method_info=method_info,
                grid_level=args.grid,
                conv_tol=args.conv_tol,
                max_cycle=args.max_cycle,
                use_density_fitting=args.density_fit,
            )

            if args.job == "optimize":
                mol_opt = optimize_geometry(mf)
                print_coordinates(mol_opt, "KOORDINAT HASIL OPTIMASI GEOMETRI")
                mf = build_calculator(
                    mol=mol_opt,
                    method_name=method_name,
                    method_info=method_info,
                    grid_level=args.grid,
                    conv_tol=args.conv_tol,
                    max_cycle=args.max_cycle,
                    use_density_fitting=args.density_fit,
                )

            print("\n============================================================")
            print("PERHITUNGAN ENERGI DIMULAI")
            print("============================================================")
            print(f"Nama molekul : {MOLECULE_NAME}")
            print(f"Metode       : {method_name}")
            print(f"XC           : {method_info.get('xc')}")
            print(f"Dispersion   : {method_info.get('disp')}")
```

```
            print(f"Basis set    : {basis_name} ({pyscf_basis})")
            print(f"Grid DFT     : {args.grid}")
            print(f"Conv tol     : {args.conv_tol}")
            print(f"Max cycle    : {args.max_cycle}")
            print(f"Density fit  : {args.density_fit}")
            print("============================================================\n")

            try:
                energy = mf.kernel()
            except Exception as exc:
                status_akhir = "GAGAL"
                print("\nERROR: Perhitungan PySCF gagal.")
                print(str(exc))
                if method_info.get("disp"):
                    print("\nKemungkinan penyebab: koreksi dispersi belum tersedia pada
instalasi PySCF Anda.")
                    print("Coba salah satu opsi berikut:")
                    print("1. Jalankan tanpa dispersi:")
                    print("   python oth01-olaparib-talazoparib-hybrid.py --method B3LYP -
-basis def2-SVP")
                    print("2. Instal dukungan dispersi sesuai lingkungan Python Anda,
misalnya:")
                    print("   pip install pyscf-dispersion simple-dftd3")
                print("\nPENTING:")
                print(f"Teks running/error sudah otomatis tersimpan di:
{running_output_log_path}")
                sys.exit(1)

            print("\n============================================================")
            print("HASIL PERHITUNGAN")
            print("============================================================")
            print(f"Nama molekul       : {MOLECULE_NAME}")
            print(f"Rumus molekul      : {MOLECULE_FORMULA}")
            print(f"Metode             : {method_name}")
            print(f"Basis set          : {basis_name}")
            print(f"Energi total       : {energy: .12f} Hartree")
            print(f"SCF konvergen      : {mf.converged}")
            print(f"File checkpoint    : {mf.chkfile}")

            if not mf.converged:
                print("\nPERINGATAN:")
                print("SCF belum konvergen.")
                print("Coba salah satu langkah berikut:")
                print("1. Naikkan --max-cycle, misalnya --max-cycle 250")
                print("2. Longgarkan sementara --conv-tol, misalnya --conv-tol 1e-7")
                print("3. Gunakan basis lebih kecil untuk uji awal, misalnya STO-3G")
                print("4. Optimasi geometri awal di Avogadro atau jalankan --job
optimize")

            print("============================================================")

            output_files, structure_summary = save_final_structure_outputs(
                mol=mf.mol,
                mf=mf,
                energy=energy,
                method_name=method_name,
                method_info=method_info,
                basis_name=basis_name,
                pyscf_basis=pyscf_basis,
                args=args,
                output_dir=output_dir,
                running_output_log_path=running_output_log_path,
            )

            print("\n============================================================")
            print("FILE HASIL AKHIR BERHASIL DISIMPAN")
            print("============================================================")
            print(f"Direktori output            : {output_dir}")
            print(f"Jumlah ikatan terdeteksi    : {len(structure_summary['bonds'])}")
            print(f"Jumlah sudut terdeteksi     : {len(structure_summary['angles'])}")
            print(f"Jumlah dihedral terdeteksi  : {len(structure_summary['dihedrals'])}")
            print("")
            for label, filename in output_files.items():
                print(f"{label:18s}: {filename}")
```

```
            print("============================================================")
            print("Catatan:")
            print("- .txt berisi koordinat akhir molekul.")
            print("- .xyz berisi koordinat 3D standar.")
            print("- .mol/.sdf/.pdb/.cml dapat dipakai untuk visualisasi struktur 3D.")
            print("- .json dan .log menyimpan data ikatan, sudut, dihedral, jarak, dan
teks running.")
            print("============================================================")

            print_orbital_info(mf)
            print("\nSELESAI.")

        except SystemExit:
            if status_akhir != "GAGAL":
                status_akhir = "DIHENTIKAN"
            raise
        except Exception:
            status_akhir = "GAGAL"
            print("\nERROR TAK TERDUGA:", file=sys.stderr)
            import traceback
            traceback.print_exc()
            print("\nPENTING:")
            print(f"Teks running/error sudah otomatis tersimpan di:
{running_output_log_path}")
            raise
        finally:
            try:
                write_running_log_footer(running_output_log_path, status=status_akhir)
            finally:
                sys.stdout = original_stdout
                sys.stderr = original_stderr


if __name__ == "__main__":
    main()
```

BAB 7

IMPLEMENTASI PySCF UNTUK HIBRIDA OLAPARIB-TALAZOPARIB

Pengantar Bab

Bab ini memperlihatkan implementasi program PySCF untuk hibrida Olaparib-Talazoparib. Dengan adanya contoh kedua, pembaca dapat membandingkan cara penyusunan file riset, struktur kode, parameter komputasi, dan pola pelaporan hasil dengan contoh pada bab sebelumnya.

Tujuan utama bab ini adalah melatih pembaca membaca program sebagai bagian dari metodologi ilmiah. Setiap baris kode perlu dipahami hubungannya dengan input molekul, metode komputasi, validasi struktur, dan kualitas data yang akan digunakan dalam pembahasan skripsi.

Setelah pengantar ini, pembahasan dilanjutkan dengan orientasi bab, peta kompetensi, uraian materi inti, contoh kerja, dan catatan metodologis yang diperlukan untuk menuntun pembaca dari konsep menuju praktik.

Bab ini memperlihatkan implementasi kedua yang dapat dibandingkan dengan contoh sebelumnya. Dengan adanya dua skrip, pembaca dapat belajar membedakan bagian program yang bersifat umum dan bagian yang spesifik untuk molekul tertentu. Bagian umum meliputi fungsi validasi, pelaporan, dan argumen terminal, sedangkan bagian spesifik meliputi rumus, SMILES, koordinat, dan catatan riset.

Orientasi Bab

Subbab ini menguraikan Orientasi Bab. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

Penjelasan Skrip

```
# FILE RISET KIMIA KOMPUTASI PySCF UNTUK HIBRIDA OLAPARIB-TALAZOPARIB
# ============================================================
#
# Molekul default : OTH-01 Olaparib-Talazoparib Hybrid
# Rumus molekul   : C24H22FN7O3
# Program         : PySCF
#
# Metode default yang dipilih:
#   B3LYP-D3BJ / def2-SVP
#
# Alasan ringkas:
#   - OTH-01 Olaparib-Talazoparib Hybrid adalah molekul organik netral tertutup-shell.
#   - B3LYP umum dipakai untuk molekul organik.
#   - Koreksi dispersi D3BJ membantu interaksi non-kovalen dan konformasi.
#   - def2-SVP cukup seimbang untuk latihan/skrining awal tanpa terlalu berat.
#
# Catatan penting:
#   - Untuk memakai koreksi dispersi D3BJ, PySCF memerlukan dukungan
#     pyscf-dispersion/simple-dftd3 pada beberapa instalasi.
#   - Jika belum tersedia, jalankan metode B3LYP biasa:
#       python oth01-olaparib-talazoparib-hybrid.py --method B3LYP --basis def2-SVP
#
# Contoh pemakaian:
#   python oth01-olaparib-talazoparib-hybrid.py
#   python oth01-olaparib-talazoparib-hybrid.py --list
#   python oth01-olaparib-talazoparib-hybrid.py --method B3LYP --basis def2-SVP
#   python oth01-olaparib-talazoparib-hybrid.py --method PBE0 --basis def2-TZVP
#   python oth01-olaparib-talazoparib-hybrid.py --method CUSTOM --xc "m06-2x" --basis
def2-TZVP
#   python oth01-olaparib-talazoparib-hybrid.py --xyz-file molekul.xyz --name "Molekul
Baru" --formula "C2H6O"
#   python oth01-olaparib-talazoparib-hybrid.py --coord-file koordinat.txt --method B3LYP
--basis "6-31G**"
#   python oth01-olaparib-talazoparib-hybrid.py --job optimize --method B3LYP --basis
def2-SVP
#
# ============================================================
```

```
import argparse
import json
import math
import sys
from datetime import datetime
from html import escape
from pathlib import Path
```

```
from pyscf import dft, gto, scf
```

```
# ============================================================
```

1. DAFTAR METODE KOMPUTASI

Subbab ini menguraikan # 1. DAFTAR METODE KOMPUTASI. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================
# family:
#   HF  = Hartree-Fock
#   DFT = Kohn-Sham DFT
# xc:
#   String functional yang dibaca oleh PySCF.
# disp:
#   None       = tanpa koreksi dispersi eksplisit.
#   "d3bj"     = koreksi dispersi D3 dengan Becke-Johnson damping.
#
# CUSTOM:
#   Dipakai bersama argumen --xc.
```

```
AVAILABLE_METHODS = {
  "HF": {
    "family": "HF",
    "xc": None,
    "disp": None,
    "description": "Hartree-Fock; RHF untuk spin 0, UHF untuk spin bukan 0.",
  },
  "LDA": {
    "family": "DFT",
    "xc": "lda,vwn",
    "disp": None,
    "description": "DFT LDA sederhana.",
  },
  "PBE": {
    "family": "DFT",
    "xc": "pbe,pbe",
    "disp": None,
    "description": "DFT GGA PBE.",
  },
  "BLYP": {
    "family": "DFT",
    "xc": "b88,lyp",
    "disp": None,
    "description": "DFT GGA BLYP.",
  },
  "B3LYP": {
    "family": "DFT",
    "xc": "b3lyp",
    "disp": None,
    "description": "DFT hybrid B3LYP, umum untuk molekul organik.",
  },
  "B3LYP-D3BJ": {
    "family": "DFT",
    "xc": "b3lyp",
    "disp": "d3bj",
    "description": "B3LYP dengan koreksi dispersi D3BJ; default untuk kandidat hibrida OTH-01.",
  },
```

```
"PBE0": {
    "family": "DFT",
    "xc": "pbe0",
    "disp": None,
    "description": "DFT hybrid PBE0.",
},
"PBE0-D3BJ": {
    "family": "DFT",
    "xc": "pbe0",
    "disp": "d3bj",
    "description": "PBE0 dengan koreksi dispersi D3BJ.",
},
"TPSS": {
    "family": "DFT",
    "xc": "tpss",
    "disp": None,
    "description": "Meta-GGA TPSS.",
},
"M06": {
    "family": "DFT",
    "xc": "m06",
    "disp": None,
    "description": "Meta-hybrid Minnesota M06 jika didukung instalasi PySCF/libxc.",
},
"CAM-B3LYP": {
    "family": "DFT",
    "xc": "cam-b3lyp",
    "disp": None,
    "description": "Range-separated hybrid CAM-B3LYP.",
},
"WB97X-D": {
    "family": "DFT",
    "xc": "wb97x-d",
    "disp": None,
    "description": "Range-separated functional dengan dispersi bawaan jika didukung.",
},
"CUSTOM": {
```

```
        "family": "DFT",
        "xc": None,
        "disp": None,
        "description": "Functional DFT custom dari argumen --xc.",
    },
}


# ============================================================
# 2. DAFTAR BASIS SET
# ============================================================

AVAILABLE_BASIS_SETS = {
    "STO-3G": "sto-3g",
    "3-21G": "3-21g",
    "6-31G": "6-31g",
    "6-31G*": "6-31g*",
    "6-31G**": "6-31g**",
    "6-31+G*": "6-31+g*",
    "6-31+G**": "6-31+g**",
    "def2-SVP": "def2-svp",
    "def2-TZVP": "def2-tzvp",
    "def2-QZVP": "def2-qzvp",
    "cc-pVDZ": "cc-pvdz",
    "cc-pVTZ": "cc-pvtz",
    "aug-cc-pVDZ": "aug-cc-pvdz",
    "CUSTOM": None,
}


# ============================================================
# 3. KONFIGURASI DEFAULT
# ============================================================

MOLECULE_NAME_DEFAULT = "OTH-01 Olaparib-Talazoparib Hybrid"
FORMULA_DEFAULT = "C24H22FN7O3"
```

Subbab ini menguraikan FORMULA_DEFAULT = "C24H22FN7O3". Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

METHOD_DEFAULT = "B3LYP-D3BJ"
Subbab ini menguraikan METHOD_DEFAULT = "B3LYP-D3BJ". Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
BASIS_DEFAULT = "def2-SVP"
```

CHARGE_DEFAULT = 0
Subbab ini menguraikan CHARGE_DEFAULT = 0. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

SPIN_DEFAULT = 0
Subbab ini menguraikan SPIN_DEFAULT = 0. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

DFT_GRID_LEVEL_DEFAULT = 3
Subbab ini menguraikan DFT_GRID_LEVEL_DEFAULT = 3. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
CONV_TOL_DEFAULT = 1e-9
```

MAX_CYCLE_DEFAULT = 150
Subbab ini menguraikan MAX_CYCLE_DEFAULT = 150. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

VERBOSE_DEFAULT = 4
Subbab ini menguraikan VERBOSE_DEFAULT = 4. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
USE_DENSITY_FITTING_DEFAULT = False
JOB_DEFAULT = "singlepoint"
```

```
OUTPUT_ROOT_DEFAULT = "hasil_pyscf"
```

BOND_SCALE_DEFAULT = 1.25
Subbab ini menguraikan BOND_SCALE_DEFAULT = 1.25. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

BOND_TOLERANCE_DEFAULT = 0.15
Subbab ini menguraikan BOND_TOLERANCE_DEFAULT = 0.15. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
AUTHOR_DEFAULT = "Kasmui"
```

AUTHOR_PHONE_DEFAULT = "0818294312"

Subbab ini menguraikan AUTHOR_PHONE_DEFAULT = "0818294312". Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
SMILES_DEFAULT = "O=C1NN=C(c2ccccc21)Cc3ccc(F)c(c3)C(=O)N4CCN(CC4)C(=O)c5nn(C)cn5"
IUPAC_NAME_DEFAULT = "Olaparib-Talazoparib merged hybrid candidate: phthalazinone-
piperazine-carbonyl-methyltriazole fluorophenyl scaffold"
```

RESEARCH_NOTE_DEFAULT = (

"Kandidat hibrida in silico Olaparib-Talazoparib untuk riset inhibitor PARP-1 pada kanker payudara BRCA-mutated. "

"Struktur ini adalah kandidat awal komputasi, bukan obat tervalidasi klinis."

)

```
# ============================================================
```

4. KOORDINAT DEFAULT OTH-01 OLAPARIB-TALAZOPARIB HYBRID

Subbab ini menguraikan # 4. KOORDINAT DEFAULT OTH-01 OLAPARIB-TALAZOPARIB HYBRID. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================
# Format XYZ standar juga diterima:
# baris 1 = jumlah atom, baris 2 = judul/komentar.
# Program otomatis membuang dua baris tersebut saat membaca koordinat.
```

DEFAULT_MOLECULE_XYZ = """
57

OTH-01 Olaparib-Talazoparib Hybrid | author: Kasmui | Telp.: 0818294312 | generated from RDKit MMFF94 conformer

O -9.17676470 0.41437369 -0.67062951

Subbab ini menguraikan O -9.17676470 0.41437369 -0.67062951. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -8.06176180 0.49482581 -0.16753486

Subbab ini menguraikan C -8.06176180 0.49482581 -0.16753486. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

N -7.50673299 1.69531264 0.20115037

Subbab ini menguraikan N -7.50673299 1.69531264 0.20115037. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

N -6.28131490 1.87477513 0.76091134

Subbab ini menguraikan N -6.28131490 1.87477513 0.76091134. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -5.54269570 0.82653472 0.98468150

Subbab ini menguraikan C -5.54269570 0.82653472 0.98468150. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -5.97939478 -0.55486754 0.65383130

Subbab ini menguraikan C -5.97939478 -0.55486754 0.65383130. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -5.19811638 -1.69258147 0.88802322

Subbab ini menguraikan C -5.19811638 -1.69258147 0.88802322. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -5.68112436 -2.95865261 0.54139571

Subbab ini menguraikan C -5.68112436 -2.95865261 0.54139571. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -6.94247566 -3.09717826 -0.03562814

Subbab ini menguraikan C -6.94247566 -3.09717826 -0.03562814. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -7.72558316 -1.96832484 -0.26888636

Subbab ini menguraikan C -7.72558316 -1.96832484 -0.26888636. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -7.23968478 -0.70105794 0.07678584

Subbab ini menguraikan C -7.23968478 -0.70105794 0.07678584. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -4.17178725 1.06850580 1.60376597

Subbab ini menguraikan C -4.17178725 1.06850580 1.60376597. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -3.03824647 0.98696543 0.60548772

Subbab ini menguraikan C -3.03824647 0.98696543 0.60548772. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -3.07497902 1.72154253 -0.59068736

Subbab ini menguraikan C -3.07497902 1.72154253 -0.59068736. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -2.03175940 1.62826903 -1.51526107

Subbab ini menguraikan C -2.03175940 1.62826903 -1.51526107. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -0.93528759 0.81632429 -1.24359438

Subbab ini menguraikan C -0.93528759 0.81632429 -1.24359438. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

F 0.03572176 0.71999733 -2.16257007

Subbab ini menguraikan F 0.03572176 0.71999733 -2.16257007. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -0.85982639 0.11767181 -0.04291134

Subbab ini menguraikan C -0.85982639 0.11767181 -0.04291134. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C -1.91401717 0.19247264 0.87718205
Subbab ini menguraikan C -1.91401717 0.19247264 0.87718205. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 0.27311029 -0.77498988 0.26021544
Subbab ini menguraikan C 0.27311029 -0.77498988 0.26021544. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
O 0.00603014 -1.94341179 0.53732934
Subbab ini menguraikan O 0.00603014 -1.94341179 0.53732934. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
N 1.56755358 -0.26864472 0.26492648
Subbab ini menguraikan N 1.56755358 -0.26864472 0.26492648. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 1.91033087 1.15992845 0.22190315
Subbab ini menguraikan C 1.91033087 1.15992845 0.22190315. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 3.23860972 1.47187533 0.92421264
Subbab ini menguraikan C 3.23860972 1.47187533 0.92421264. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
N 4.34355559 0.66544721 0.40635318
Subbab ini menguraikan N 4.34355559 0.66544721 0.40635318. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 4.01523165 -0.69500253 -0.03806045
Subbab ini menguraikan C 4.01523165 -0.69500253 -0.03806045. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 2.67112405 -1.19961378 0.50606128
Subbab ini menguraikan C 2.67112405 -1.19961378 0.50606128. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 5.64133847 1.19994490 0.45503981
Subbab ini menguraikan C 5.64133847 1.19994490 0.45503981. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
O 5.84045831 2.27337185 1.03239277
Subbab ini menguraikan O 5.84045831 2.27337185 1.03239277. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
C 6.75303753 0.46376794 -0.20920260
Subbab ini menguraikan C 6.75303753 0.46376794 -0.20920260. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
N 8.05190099 0.75294464 0.01836758

Subbab ini menguraikan N 8.05190099 0.75294464 0.01836758. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

N 8.68585954 -0.09887769 -0.79813534

Subbab ini menguraikan N 8.68585954 -0.09887769 -0.79813534. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C 10.12252636 -0.11004365 -0.86555396

Subbab ini menguraikan C 10.12252636 -0.11004365 -0.86555396. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

C 7.81205748 -0.87106256 -1.48701459

Subbab ini menguraikan C 7.81205748 -0.87106256 -1.48701459. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

N 6.58696974 -0.54853175 -1.14731906

Subbab ini menguraikan N 6.58696974 -0.54853175 -1.14731906. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

H -8.06183634 2.53065827 0.04238987

Subbab ini menguraikan H -8.06183634 2.53065827 0.04238987. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

H -4.21016294 -1.63285756 1.33239943

Subbab ini menguraikan H -4.21016294 -1.63285756 1.33239943. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

H -5.06782788 -3.84004186 0.72001252

Subbab ini menguraikan H -5.06782788 -3.84004186 0.72001252. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

H -7.31468893 -4.08303557 -0.30509415

Subbab ini menguraikan H -7.31468893 -4.08303557 -0.30509415. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

H -8.71039211 -2.07474723 -0.71981209

Subbab ini menguraikan H -8.71039211 -2.07474723 -0.71981209. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

H -4.15618629 2.06768938 2.05797733

Subbab ini menguraikan H -4.15618629 2.06768938 2.05797733. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

H -4.02621436 0.36230053 2.42925773

Subbab ini menguraikan H -4.02621436 0.36230053 2.42925773. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

H -3.92527178 2.36461362 -0.81750445

Subbab ini menguraikan H -3.92527178 2.36461362 -0.81750445. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

H -2.07388541 2.17846071 -2.45074927
Subbab ini menguraikan H -2.07388541 2.17846071 -2.45074927. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H -1.85079948 -0.38234106 1.79989477
Subbab ini menguraikan H -1.85079948 -0.38234106 1.79989477. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 1.98879972 1.43451349 -0.83598133
Subbab ini menguraikan H 1.98879972 1.43451349 -0.83598133. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 1.10717691 1.75072120 0.67286849
Subbab ini menguraikan H 1.10717691 1.75072120 0.67286849. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 3.46528213 2.53843853 0.82111102
Subbab ini menguraikan H 3.46528213 2.53843853 0.82111102. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 3.15957237 1.25107044 1.99564180
Subbab ini menguraikan H 3.15957237 1.25107044 1.99564180. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 4.80203965 -1.38614209 0.27829957
Subbab ini menguraikan H 4.80203965 -1.38614209 0.27829957. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 3.97874494 -0.67078349 -1.13323721
Subbab ini menguraikan H 3.97874494 -0.67078349 -1.13323721. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 2.43549258 -2.17719940 0.07093817
Subbab ini menguraikan H 2.43549258 -2.17719940 0.07093817. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 2.73676323 -1.33164850 1.59328922
Subbab ini menguraikan H 2.73676323 -1.33164850 1.59328922. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 10.44751420 -0.86759724 -1.58300446
Subbab ini menguraikan H 10.44751420 -0.86759724 -1.58300446. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 10.51425615 -0.34533850 0.12715758
Subbab ini menguraikan H 10.51425615 -0.34533850 0.12715758. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.
H 10.46213994 0.87751312 -1.18756512

Subbab ini menguraikan H 10.46213994 0.87751312 -1.18756512. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

H 8.10562011 -1.62625694 -2.20260445

"""

```
# ============================================================
```

5. DATA ATOM DAN JARI-JARI KOVALEN

Subbab ini menguraikan # 5. DATA ATOM DAN JARI-JARI KOVALEN. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

COVALENT_RADII_ANGSTROM = {
    "H": 0.31,
    "He": 0.28,
    "Li": 1.28,
    "Be": 0.96,
    "B": 0.84,
    "C": 0.76,
    "N": 0.71,
    "O": 0.66,
    "F": 0.57,
    "Ne": 0.58,
    "Na": 1.66,
    "Mg": 1.41,
    "Al": 1.21,
    "Si": 1.11,
    "P": 1.07,
    "S": 1.05,
    "Cl": 1.02,
    "Ar": 1.06,
    "K": 2.03,
    "Ca": 1.76,
    "Fe": 1.24,
    "Cu": 1.32,
```

```
    "Zn": 1.22,
    "Br": 1.20,
    "I": 1.39,
}

ATOMIC_NUMBERS = {
    "H": 1,
    "He": 2,
    "Li": 3,
    "Be": 4,
    "B": 5,
    "C": 6,
    "N": 7,
    "O": 8,
    "F": 9,
    "Ne": 10,
    "Na": 11,
    "Mg": 12,
    "Al": 13,
    "Si": 14,
    "P": 15,
    "S": 16,
    "Cl": 17,
    "Ar": 18,
    "K": 19,
    "Ca": 20,
    "Fe": 26,
    "Cu": 29,
    "Zn": 30,
    "Br": 35,
    "I": 53,
}

# ============================================================
# 6. FUNGSI BANTUAN UMUM
```

Subbab ini menguraikan # 6. FUNGSI BANTUAN UMUM. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def clean_element_symbol(symbol):
    letters = "".join(ch for ch in str(symbol).strip() if ch.isalpha())
    if not letters:
        return ""
    if len(letters) == 1:
        return letters.upper()
    return letters[0].upper() + letters[1:].lower()

def safe_filename(text):
    safe_chars = []
    for char in str(text):
        if char.isalnum() or char in ["-", "_"]:
            safe_chars.append(char)
        elif char in ["*", "+"]:
            safe_chars.append("x")
        else:
            safe_chars.append("_")
    value = "".join(safe_chars).strip("_")
    while "__" in value:
        value = value.replace("__", "_")
    return value or "molecule"

def normalize_choice(user_value, allowed_dict, label):
    if user_value is None:
        return None
    user_compact = str(user_value).strip().lower().replace(" ", "")
    for key in allowed_dict:
        key_compact = key.strip().lower().replace(" ", "")
        if user_compact == key_compact:
            return key
    print(f"\nERROR: {label} '{user_value}' tidak tersedia.")
    print(f"Pilihan {label} yang tersedia:")
    for key in allowed_dict:
        print(f"  - {key}")
    sys.exit(1)

def validate_positive_float(value, label):
    if value <= 0:
        print(f"ERROR: {label} harus bernilai positif.")
        sys.exit(1)

def validate_nonnegative_int(value, label):
    if value < 0:
        print(f"ERROR: {label} harus integer >= 0.")
        sys.exit(1)

def validate_positive_int(value, label):
    if value <= 0:
        print(f"ERROR: {label} harus integer positif.")
```

sys.exit(1)

```
class TeeOutput:
```

"""

Menyalin semua teks yang ditulis ke terminal sekaligus ke file log.

Tujuan:

- Teks proses running tetap tampil normal di layar.

- Teks yang sama otomatis tersimpan ke file .log.

- stdout dan stderr ikut ditangkap, termasuk pesan error PySCF.

"""

```
    def __init__(self, terminal_stream, log_stream):
```

self.terminal_stream = terminal_stream

self.log_stream = log_stream

```
    def write(self, message):
```

self.terminal_stream.write(message)

self.log_stream.write(message)

self.flush()

```
    def flush(self):
```

self.terminal_stream.flush()

self.log_stream.flush()

```
    def isatty(self):
        return getattr(self.terminal_stream, "isatty", lambda: False)()

    def fileno(self):
        return self.terminal_stream.fileno()

def write_running_log_header(log_path, args, method_name, basis_name, pyscf_basis):
    print(identity_header_text())
    print("\n============================================================")
    print("LOG RUNNING OTOMATIS AKTIF")
    print("============================================================")
    print(f"File log running  : {log_path}")
    print(f"Waktu mulai       : {datetime.now().strftime('%Y-%m-%d %H:%M:%S')}")
    print(f"Nama molekul      : {args.name}")
    print(f"Rumus molekul     : {args.formula}")
    print(f"Metode            : {method_name}")
    print(f"Basis set         : {basis_name} ({pyscf_basis})")
    print(f"Jenis perhitungan : {args.job}")
    print("Catatan           : Semua teks terminal stdout/stderr disalin ke file ini.")
    print("============================================================")


def write_running_log_footer(log_path, status="SELESAI"):
    print("\n============================================================")
    print("LOG RUNNING OTOMATIS DITUTUP")
```

```
    print("============================================================")
    print(f"Status akhir      : {status}")
    print(f"Waktu selesai     : {datetime.now().strftime('%Y-%m-%d %H:%M:%S')}")
    print(f"File log running  : {log_path}")
    print("============================================================")
```

```
# ============================================================
```

7. PEMBACAAN DAN VALIDASI KOORDINAT

Subbab ini menguraikan # 7. PEMBACAAN DAN VALIDASI KOORDINAT. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================
```

```
def parse_xyz_like_text(text):
```

"""

Membaca koordinat dari:

1. XYZ standar:

jumlah_atom

komentar

Atom x y z

2. Blok koordinat langsung:

Atom x y z

3. File teks dengan komentar diawali #.

"""

```
    if not text or not str(text).strip():
        raise ValueError("Blok koordinat kosong.")
```

raw_lines = []

```
    for line in str(text).splitlines():
```

stripped = line.strip()

```
        if not stripped:
```

continue

```
        if stripped.startswith("#"):
```

continue

raw_lines.append(stripped)

```
    if not raw_lines:
        raise ValueError("Tidak ada baris koordinat yang dapat dibaca.")
```

expected_count = None

coordinate_lines = raw_lines

first_parts = raw_lines[0].split()

```
    if len(first_parts) == 1:
        try:
```

expected_count = int(first_parts[0])

```
        coordinate_lines = raw_lines[2:2 + expected_count]
    except ValueError:
        expected_count = None

atoms = []
for number, line in enumerate(coordinate_lines, start=1):
    parts = line.split()
    if len(parts) < 4:
        raise ValueError(
            f"Format koordinat salah pada baris koordinat ke-{number}: '{line}'. "
            "Format benar: SimbolAtom X Y Z"
        )

    symbol = clean_element_symbol(parts[0])
    if symbol not in ATOMIC_NUMBERS:
        raise ValueError(
            f"Simbol atom '{parts[0]}' pada baris ke-{number} tidak dikenali."
        )

    try:
        x = float(parts[1])
        y = float(parts[2])
        z = float(parts[3])
    except ValueError as exc:
        raise ValueError(
            f"Koordinat X/Y/Z harus numerik pada baris ke-{number}: '{line}'."
        ) from exc

    atoms.append({
        "symbol": symbol,
        "x": x,
        "y": y,
        "z": z,
    })

if expected_count is not None and len(atoms) != expected_count:
    raise ValueError(
        f"Jumlah atom di header XYZ = {expected_count}, tetapi yang terbaca = {len(atoms)}."
    )

if not atoms:
    raise ValueError("Tidak ada atom yang berhasil dibaca.")
```

```
    return atoms

def read_text_file(path):
    file_path = Path(path)
    if not file_path.exists():
        print(f"ERROR: File koordinat tidak ditemukan: {file_path}")
        sys.exit(1)
    return file_path.read_text(encoding="utf-8")

def format_atom_block(atoms):
    return "\n".join(
        f"{atom['symbol']:2s}  {atom['x']:14.8f}  {atom['y']:14.8f}  {atom['z']:14.8f}"
        for atom in atoms
    )


def get_coordinate_atoms(args):
    if args.xyz_file:
        source_text = read_text_file(args.xyz_file)
        source_label = f"file XYZ: {args.xyz_file}"
    elif args.coord_file:
        source_text = read_text_file(args.coord_file)
        source_label = f"file koordinat: {args.coord_file}"
    else:
        source_text = DEFAULT_MOLECULE_XYZ
        source_label = "koordinat default di dalam kode"

    try:
        atoms = parse_xyz_like_text(source_text)
    except ValueError as exc:
        print("\nERROR: Koordinat molekul tidak valid.")
        print(str(exc))
        print("\nContoh format benar:")
        print("C   0.000000   0.000000   0.000000")
        print("H   0.000000   0.000000   1.089000")
        sys.exit(1)

    return atoms, source_label

# ============================================================
```

8. RESOLUSI METODE DAN BASIS SET

Subbab ini menguraikan # 8. RESOLUSI METODE DAN BASIS SET. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def resolve_method(args):
    method_name = normalize_choice(args.method, AVAILABLE_METHODS, "metode")
    info = dict(AVAILABLE_METHODS[method_name])

    if method_name == "CUSTOM":
        if not args.xc or not args.xc.strip():
```

```
            print("ERROR: --method CUSTOM wajib disertai --xc, contoh: --xc 'm06-2x'")
        sys.exit(1)
    info["xc"] = args.xc.strip()
    info["description"] = f"DFT custom dengan functional PySCF: {args.xc.strip()}"

    if args.disp:
        info["disp"] = args.disp.strip().lower()

    return method_name, info

def resolve_basis(args):
    basis_name = normalize_choice(args.basis, AVAILABLE_BASIS_SETS, "basis set")

    if basis_name == "CUSTOM":
        if not args.basis_custom or not args.basis_custom.strip():
            print("ERROR: --basis CUSTOM wajib disertai --basis-custom, contoh: --basis-
custom 'def2-tzvpp'")
        sys.exit(1)
    pyscf_basis = args.basis_custom.strip()
    else:
        pyscf_basis = AVAILABLE_BASIS_SETS[basis_name]

    return basis_name, pyscf_basis

# ============================================================
```

9. MEMBANGUN OBJEK MOLEKUL DAN KALKULATOR

Subbab ini menguraikan # 9. MEMBANGUN OBJEK MOLEKUL DAN KALKULATOR. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def build_molecule(atom_block, pyscf_basis, charge, spin, verbose):
    mol = gto.M(
        atom=atom_block,
        basis=pyscf_basis,
        unit="Angstrom",
        charge=charge,
        spin=spin,
        verbose=verbose,
    )
    return mol

def build_calculator(mol, method_name, method_info, grid_level, conv_tol, max_cycle,
use_density_fitting):
    family = method_info["family"]
```

```
    if family == "HF":
        mf = scf.RHF(mol) if mol.spin == 0 else scf.UHF(mol)
    elif family == "DFT":
        mf = dft.RKS(mol) if mol.spin == 0 else dft.UKS(mol)
        mf.xc = method_info["xc"]
        mf.grids.level = grid_level
    else:
        raise ValueError(f"Family metode tidak dikenal: {family}")

    if method_info.get("disp"):
        # PySCF modern mendukung atribut mf.disp untuk D3/D4 jika ekstensi tersedia.
        mf.disp = method_info["disp"]

    if use_density_fitting:
        mf = mf.density_fit()

    mf.conv_tol = conv_tol
    mf.max_cycle = max_cycle
    mf.chkfile = safe_filename(f"{MOLECULE_NAME}_{method_name}_{BASIS_LABEL}.chk")

    return mf

# ============================================================
```

10. OPTIMASI GEOMETRI OPSIONAL

Subbab ini menguraikan # 10. OPTIMASI GEOMETRI OPSIONAL. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def optimize_geometry(mf):
    try:
        from pyscf.geomopt.geometric_solver import optimize
    except ImportError:
        print("\nERROR: Optimasi geometri memerlukan paket geomeTRIC.")
        print("Instal terlebih dahulu:")
        print("  pip install geometric")
        sys.exit(1)

    print("\n============================================================")
    print("OPTIMASI GEOMETRI DIMULAI")
    print("============================================================")
    mol_opt = optimize(mf)
    print("\n============================================================")
    print("OPTIMASI GEOMETRI SELESAI")
    print("============================================================")
    return mol_opt

# ============================================================
```

11. INFORMASI DAN PELAPORAN DASAR

Subbab ini menguraikan # 11. INFORMASI DAN PELAPORAN DASAR. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================
```

```
def print_available_options():
    print("\n============================================================")
    print("DAFTAR METODE KOMPUTASI")
    print("============================================================")
    for method_name, info in AVAILABLE_METHODS.items():
```

xc_text = info["xc"] if info["xc"] else "-"

disp_text = info["disp"] if info["disp"] else "-"

```
        print(f"{method_name:12s} | family={info['family']:3s} | xc={xc_text:12s} |
disp={disp_text:5s} | {info['description']}")
```

```
    print("\n============================================================")
    print("DAFTAR BASIS SET")
    print("============================================================")
    for basis_name, pyscf_name in AVAILABLE_BASIS_SETS.items():
        print(f"{basis_name:12s} : {pyscf_name if pyscf_name else 'isi melalui --basis-
custom'}")
```

```
    print("\n============================================================")
    print("CONTOH PERINTAH")
    print("============================================================")
    print("python oth01-olaparib-talazoparib-hybrid.py")
    print("python oth01-olaparib-talazoparib-hybrid.py --method B3LYP --basis def2-SVP")
    print("python oth01-olaparib-talazoparib-hybrid.py --xyz-file molekul.xyz --name
'Molekul Baru' --method PBE0 --basis def2-TZVP")
    print("python oth01-olaparib-talazoparib-hybrid.py --method CUSTOM --xc 'm06-2x' --
basis CUSTOM --basis-custom 'def2-tzvpp'")
    print("python oth01-olaparib-talazoparib-hybrid.py --job optimize --method B3LYP --
basis def2-SVP")
```

```
    print("\n============================================================")
    print("CONTOH FORMAT KOORDINAT")
    print("============================================================")
    print("3")
    print("Water")
    print("O   0.000000   0.000000   0.000000")
    print("H   0.000000  -0.757000   0.587000")
    print("H   0.000000   0.757000   0.587000")
    print()
```

```
def print_molecule_info(mol, method_name, method_info, basis_name, pyscf_basis, charge,
spin, job, coordinate_source):
    print("\n============================================================")
    print("INFORMASI SISTEM")
    print("============================================================")
    print(f"Nama molekul        : {MOLECULE_NAME}")
    print(f"Rumus molekul       : {MOLECULE_FORMULA}")
    print(f"Sumber koordinat    : {coordinate_source}")
    print(f"Jumlah atom         : {mol.natm}")
    print(f"Jumlah elektron     : {mol.nelectron}")
    print(f"Jumlah fungsi basis : {mol.nao_nr()}")
    print(f"Muatan total        : {charge}")
    print(f"Spin PySCF          : {spin}")
    print(f"Metode komputasi    : {method_name}")
    print(f"Family metode       : {method_info['family']}")
    print(f"XC functional       : {method_info.get('xc')}")
    print(f"Dispersion          : {method_info.get('disp')}")
    print(f"Basis set           : {basis_name} ({pyscf_basis})")
    print(f"Jenis perhitungan   : {job}")
    print("Satuan koordinat    : Angstrom")
    print("============================================================")
```

```
def print_coordinates(mol, title):
    print("\n============================================================")
    print(title)
    print("============================================================")
    coords = mol.atom_coords(unit="Angstrom")
    for i in range(mol.natm):
        symbol = clean_element_symbol(mol.atom_symbol(i))
        x, y, z = coords[i]
        print(f"{symbol:2s}  {x:14.8f}  {y:14.8f}  {z:14.8f}")


# ============================================================
```

12. ANALISIS STRUKTUR AKHIR

Subbab ini menguraikan # 12. ANALISIS STRUKTUR AKHIR. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def get_atom_records(mol):
    coords = mol.atom_coords(unit="Angstrom")
    atoms = []
    for i in range(mol.natm):
        symbol = clean_element_symbol(mol.atom_symbol(i))
        atomic_number = ATOMIC_NUMBERS.get(symbol)
        try:
            atomic_number = int(mol.atom_charge(i))
        except Exception:
            pass
        x, y, z = coords[i]
        atoms.append({
            "index": i + 1,
            "symbol": symbol,
            "atomic_number": atomic_number,
            "x": float(x),
            "y": float(y),
            "z": float(z),
        })
    return atoms

def distance_between_atoms(atom_a, atom_b):
    dx = atom_a["x"] - atom_b["x"]
    dy = atom_a["y"] - atom_b["y"]
    dz = atom_a["z"] - atom_b["z"]
    return math.sqrt(dx * dx + dy * dy + dz * dz)
```

```
def vector_between_atoms(atom_a, atom_b):
    return [
        atom_b["x"] - atom_a["x"],
        atom_b["y"] - atom_a["y"],
        atom_b["z"] - atom_a["z"],
    ]

def dot_product(vec_a, vec_b):
    return vec_a[0] * vec_b[0] + vec_a[1] * vec_b[1] + vec_a[2] * vec_b[2]

def cross_product(vec_a, vec_b):
    return [
        vec_a[1] * vec_b[2] - vec_a[2] * vec_b[1],
        vec_a[2] * vec_b[0] - vec_a[0] * vec_b[2],
        vec_a[0] * vec_b[1] - vec_a[1] * vec_b[0],
    ]

def vector_norm(vec):
    return math.sqrt(dot_product(vec, vec))

def normalize_vector(vec):
    norm = vector_norm(vec)
    if norm < 1e-15:
        return [0.0, 0.0, 0.0]
    return [value / norm for value in vec]

def calculate_angle_degrees(atom_i, atom_j, atom_k):
    vec_ji = vector_between_atoms(atom_j, atom_i)
    vec_jk = vector_between_atoms(atom_j, atom_k)
    norm_ji = vector_norm(vec_ji)
    norm_jk = vector_norm(vec_jk)
    if norm_ji < 1e-15 or norm_jk < 1e-15:
        return None
    cos_theta = dot_product(vec_ji, vec_jk) / (norm_ji * norm_jk)
    cos_theta = max(-1.0, min(1.0, cos_theta))
    return math.degrees(math.acos(cos_theta))

def calculate_dihedral_degrees(atom_i, atom_j, atom_k, atom_l):
    p0 = [atom_i["x"], atom_i["y"], atom_i["z"]]
    p1 = [atom_j["x"], atom_j["y"], atom_j["z"]]
    p2 = [atom_k["x"], atom_k["y"], atom_k["z"]]
    p3 = [atom_l["x"], atom_l["y"], atom_l["z"]]

    b0 = [p0[n] - p1[n] for n in range(3)]
```

```
    b1 = [p2[n] - p1[n] for n in range(3)]
    b2 = [p3[n] - p2[n] for n in range(3)]
    b1_unit = normalize_vector(b1)

    v = [b0[n] - dot_product(b0, b1_unit) * b1_unit[n] for n in range(3)]
    w = [b2[n] - dot_product(b2, b1_unit) * b1_unit[n] for n in range(3)]

    if vector_norm(v) < 1e-15 or vector_norm(w) < 1e-15:
        return None

    x_value = dot_product(v, w)
    y_value = dot_product(cross_product(b1_unit, v), w)
    return math.degrees(math.atan2(y_value, x_value))

def infer_bonds(atoms, bond_scale=BOND_SCALE_DEFAULT,
bond_tolerance=BOND_TOLERANCE_DEFAULT):
    bonds = []
    for i in range(len(atoms)):
        for j in range(i + 1, len(atoms)):
            atom_i = atoms[i]
            atom_j = atoms[j]
            symbol_i = atom_i["symbol"]
            symbol_j = atom_j["symbol"]
            radius_i = COVALENT_RADII_ANGSTROM.get(symbol_i, 0.77)
            radius_j = COVALENT_RADII_ANGSTROM.get(symbol_j, 0.77)
            distance = distance_between_atoms(atom_i, atom_j)
            threshold = bond_scale * (radius_i + radius_j) + bond_tolerance
            if symbol_i == "H" and symbol_j == "H" and distance > 0.90:
                continue
            if distance <= threshold:
                bonds.append({
                    "atom1": atom_i["index"],
                    "atom2": atom_j["index"],
                    "symbol1": symbol_i,
                    "symbol2": symbol_j,
                    "distance": float(distance),
                    "order": 1,
                })
    return bonds

def build_adjacency(atoms, bonds):
```

```
    adjacency = {atom["index"]: [] for atom in atoms}
    for bond in bonds:
        atom1 = bond["atom1"]
        atom2 = bond["atom2"]
        adjacency[atom1].append(atom2)
        adjacency[atom2].append(atom1)
    for atom_index in adjacency:
        adjacency[atom_index] = sorted(adjacency[atom_index])
    return adjacency


def infer_angles(atoms, bonds):
    atom_by_index = {atom["index"]: atom for atom in atoms}
    adjacency = build_adjacency(atoms, bonds)
    angles = []
    for center in sorted(adjacency):
        neighbors = adjacency[center]
        for a in range(len(neighbors)):
            for b in range(a + 1, len(neighbors)):
                atom_i_index = neighbors[a]
                atom_k_index = neighbors[b]
                angle_value = calculate_angle_degrees(
                    atom_by_index[atom_i_index],
                    atom_by_index[center],
                    atom_by_index[atom_k_index],
                )
                if angle_value is None:
                    continue
                angles.append({
                    "atom1": atom_i_index,
                    "atom2": center,
                    "atom3": atom_k_index,
                    "symbol1": atom_by_index[atom_i_index]["symbol"],
                    "symbol2": atom_by_index[center]["symbol"],
                    "symbol3": atom_by_index[atom_k_index]["symbol"],
                    "angle_degrees": float(angle_value),
                })
    return angles

def infer_dihedrals(atoms, bonds):
```

```
    atom_by_index = {atom["index"]: atom for atom in atoms}
    adjacency = build_adjacency(atoms, bonds)
    seen = set()
    dihedrals = []
    for bond in bonds:
        j = bond["atom1"]
        k = bond["atom2"]
        for i in adjacency[j]:
            if i == k:
                continue
            for l in adjacency[k]:
                if l == j:
                    continue
            dihedral_tuple = (i, j, k, l)
            reverse_tuple = tuple(reversed(dihedral_tuple))
            canonical = min(dihedral_tuple, reverse_tuple)
            if canonical in seen:
                continue
            seen.add(canonical)
            dihedral_value = calculate_dihedral_degrees(
                atom_by_index[i],
                atom_by_index[j],
                atom_by_index[k],
                atom_by_index[l],
            )
            if dihedral_value is None:
                continue
            dihedrals.append({
                "atom1": i,
                "atom2": j,
                "atom3": k,
                "atom4": l,
                "symbol1": atom_by_index[i]["symbol"],
                "symbol2": atom_by_index[j]["symbol"],
                "symbol3": atom_by_index[k]["symbol"],
                "symbol4": atom_by_index[l]["symbol"],
                "dihedral_degrees": float(dihedral_value),
```

```
        })
    return dihedrals

def calculate_distance_matrix(atoms):
    return [
        [float(distance_between_atoms(atom_i, atom_j)) for atom_j in atoms]
        for atom_i in atoms
    ]


# ============================================================
# 13. PENULISAN FILE HASIL
```

Subbab ini menguraikan # 13. PENULISAN FILE HASIL. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def make_output_dir(args, method_name, basis_name):
    timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
    folder_name = safe_filename(
        f"{MOLECULE_NAME}_{MOLECULE_FORMULA}_{method_name}_{basis_name}_{args.job}_{timestamp}"
    )
    if args.output_dir:
        output_path = Path(args.output_dir)
    else:
        output_path = Path(OUTPUT_ROOT_DEFAULT) / folder_name
    output_path.mkdir(parents=True, exist_ok=True)
    return output_path

def make_output_prefix(method_name, basis_name, job):
    timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
    return safe_filename(f"{MOLECULE_NAME}_{MOLECULE_FORMULA}_{method_name}_{basis_name}_{job}_{timestamp}")

def format_xyz_coordinate_block(atoms):
    return "\n".join(
        f"{atom['symbol']:2s}  {atom['x']:14.8f}  {atom['y']:14.8f}  {atom['z']:14.8f}"
        for atom in atoms
    )


def identity_header_lines():
    return [
        f"author: {AUTHOR_DEFAULT}",
        f"Telp.: {AUTHOR_PHONE_DEFAULT}",
    ]
```

```
def identity_header_text():
    return "\n".join(identity_header_lines())

def write_txt_coordinates(path, atoms, metadata):
    lines = identity_header_lines() + [""]
    lines.append("KOORDINAT AKHIR MOLEKUL")
    lines.append("=" * 60)
    for key, value in metadata.items():
        lines.append(f"{key}: {value}")
    lines.append("=" * 60)
    lines.append("Format: SimbolAtom        X              Y              Z")
    lines.append("Satuan: Angstrom")
    lines.append("")
    lines.append(format_xyz_coordinate_block(atoms))
    lines.append("")
    lines.append("BLOK KOORDINAT SIAP SALIN UNTUK PySCF/XYZ")
    lines.append("=" * 60)
    lines.append('\"\"\"')
    lines.append(format_xyz_coordinate_block(atoms))
    lines.append('\"\"\"')
    lines.append("")
    Path(path).write_text("\n".join(lines), encoding="utf-8")

def write_xyz_file(path, atoms, metadata):
    title = (
        f"{metadata['Nama molekul']} | author: {AUTHOR_DEFAULT} | Telp.: {AUTHOR_PHONE_DEFAULT} | "
        f"{metadata['Metode']} / {metadata['Basis set']} | "
        f"E = {metadata['Energi total Hartree']} Hartree | "
        f"SCF converged = {metadata['SCF konvergen']}"
    )
    lines = [str(len(atoms)), title, format_xyz_coordinate_block(atoms)]
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")

def write_mol_file(path, atoms, bonds, title=None):
    title = title or MOLECULE_NAME
    lines = []
    lines.append(title[:80])
    lines.append(f"  author: {AUTHOR_DEFAULT} | Telp.: {AUTHOR_PHONE_DEFAULT}")
    lines.append("  Generated final 3D coordinates and inferred bonds by PySCF Python Script")
    lines.append(f"{len(atoms):3d}{len(bonds):3d}  0  0  0  0            999 V2000")
    for atom in atoms:
        lines.append(
            f"{atom['x']:10.4f}"
            f"{atom['y']:10.4f}"
            f"{atom['z']:10.4f} "
            f"{atom['symbol']:<3s}"
            " 0  0  0  0  0  0  0  0  0  0  0  0"
```

```
        )

    for bond in bonds:
        lines.append(
            f"{bond['atom1']:3d}"
            f"{bond['atom2']:3d}"
            f"{int(bond.get('order', 1)):3d}"
            "  0  0  0  0"
        )

    lines.append("M  END")
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_sdf_file(path, atoms, bonds, metadata):
    mol_text_path = Path(path).with_suffix(".tmp_mol")
    write_mol_file(mol_text_path, atoms, bonds, title=metadata["Nama molekul"])
    mol_text = mol_text_path.read_text(encoding="utf-8")
    mol_text_path.unlink(missing_ok=True)

    lines = [mol_text.rstrip("\n")]
    for key, value in metadata.items():
        lines.append(f">  <{key}>")
        lines.append(str(value))
        lines.append("")
    lines.append("$$$$")
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_pdb_file(path, atoms, bonds, metadata):
    lines = []
    lines.append(f"HEADER    {metadata['Nama molekul'][:50]}")
    lines.append(f"REMARK    author: {AUTHOR_DEFAULT}")
    lines.append(f"REMARK    Telp.: {AUTHOR_PHONE_DEFAULT}")
    lines.append(f"TITLE     {metadata['Metode']} / {metadata['Basis set']}")
    lines.append(f"REMARK    Energy Hartree: {metadata['Energi total Hartree']}")
    lines.append(f"REMARK    SCF converged: {metadata['SCF konvergen']}")

    for atom in atoms:
        atom_index = atom["index"]
        symbol = atom["symbol"]
        lines.append(
            f"HETATM{atom_index:5d} "
            f"{symbol:<4s}"
            f" MOL     1    "
            f"{atom['x']:8.3f}"
            f"{atom['y']:8.3f}"
```

```
            f"{atom['z']:8.3f}"
            f"  1.00  0.00          "
            f"{symbol:>2s}"
        )

    adjacency = build_adjacency(atoms, bonds)
    for atom_index in sorted(adjacency):
        if not adjacency[atom_index]:
            continue
        conect_targets = "".join(f"{target:5d}" for target in adjacency[atom_index])
        lines.append(f"CONECT{atom_index:5d}{conect_targets}")

    lines.append("END")
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_cml_file(path, atoms, bonds, metadata):
    lines = []
    lines.append('<?xml version="1.0" encoding="UTF-8"?>')
    lines.append('<cml xmlns="http://www.xml-cml.org/schema">')
    lines.append(f'  <molecule id="{escape(safe_filename(metadata["Nama molekul"]))}">')
    lines.append("    <metadataList>")
    for key, value in metadata.items():
        lines.append(f'      <metadata name="{escape(str(key))}"
content="{escape(str(value))}"/>')
    lines.append("    </metadataList>")
    lines.append("    <atomArray>")
    for atom in atoms:
        lines.append(
            f'      <atom id="a{atom["index"]}" '
            f'elementType="{escape(atom["symbol"])}" '
            f'x3="{atom["x"]:.8f}" '
            f'y3="{atom["y"]:.8f}" '
            f'z3="{atom["z"]:.8f}"/>'
        )
    lines.append("    </atomArray>")
    lines.append("    <bondArray>")
    for bond in bonds:
        lines.append(
            f'      <bond atomRefs2="a{bond["atom1"]} a{bond["atom2"]}" '
            f'order="{int(bond.get("order", 1))}"/>'
        )
    lines.append("    </bondArray>")
    lines.append("  </molecule>")
    lines.append("</cml>")
    Path(path).write_text("\n".join(lines) + "\n", encoding="utf-8")


def write_json_structure_file(path, atoms, bonds, angles, dihedrals, distance_matrix,
metadata):
```

```
    structure_data = {
        "metadata": metadata,
        "atoms": atoms,
        "bonds": bonds,
        "angles": angles,
        "dihedrals": dihedrals,
        "distance_matrix_angstrom": distance_matrix,
        "notes": {
            "bond_inference": (
                "Ikatan diinferensi dari jarak antaratom dan jari-jari kovalen. "
                "Orde ikatan ditulis 1 untuk tujuan visualisasi struktur 3D."
            ),
            "indexing": "Semua indeks atom memakai sistem 1-based.",
        },
    }
    Path(path).write_text(
            json.dumps(structure_data, indent=2, ensure_ascii=False),
        encoding="utf-8",
    )

def write_log_file(path, atoms, bonds, angles, dihedrals, distance_matrix, metadata,
output_files):
    lines = identity_header_lines() + [""]
        lines.append("=" * 80)
        lines.append("LAPORAN AKHIR PERHITUNGAN KIMIA KOMPUTASI PySCF")
        lines.append("=" * 80)
        for key, value in metadata.items():
            lines.append(f"{key:28s}: {value}")

        lines.append("")
        lines.append("=" * 80)
        lines.append("KOORDINAT AKHIR MOLEKUL")
        lines.append("=" * 80)
        lines.append("No.  Atom            X             Y             Z       Satuan")
        lines.append("-" * 80)
        for atom in atoms:
            lines.append(
            f"{atom['index']:3d}  {atom['symbol']:2s}  "
            f"{atom['x']:14.8f}  {atom['y']:14.8f}  {atom['z']:14.8f}  Angstrom"
        )

        lines.append("")
        lines.append("=" * 80)
        lines.append("DAFTAR IKATAN HASIL INFERENSI")
        lines.append("=" * 80)
        lines.append("No.  Atom-1       Atom-2       Orde    Jarak (Angstrom)")
```

```
    lines.append("-" * 80)
    if bonds:
        for number, bond in enumerate(bonds, start=1):
            lines.append(
          f"{number:3d}  "
          f"{bond['atom1']:3d}-{bond['symbol1']:<2s}     "
          f"{bond['atom2']:3d}-{bond['symbol2']:<2s}     "
          f"{int(bond.get('order', 1)):3d}     "
          f"{bond['distance']:12.6f}"
        )
    else:
        lines.append("Tidak ada ikatan terdeteksi. Periksa --bond-scale atau --bond-
tolerance.")
```

```
    lines.append("")
    lines.append("=" * 80)
    lines.append("DAFTAR SUDUT IKATAN")
    lines.append("=" * 80)
    lines.append("No.  Atom-1       Atom-2/Pusat Atom-3       Sudut (derajat)")
    lines.append("-" * 80)
    if angles:
        for number, angle in enumerate(angles, start=1):
            lines.append(
          f"{number:3d}  "
          f"{angle['atom1']:3d}-{angle['symbol1']:<2s}     "
          f"{angle['atom2']:3d}-{angle['symbol2']:<2s}      "
          f"{angle['atom3']:3d}-{angle['symbol3']:<2s}     "
          f"{angle['angle_degrees']:12.6f}"
        )
    else:
        lines.append("Tidak ada sudut ikatan terdeteksi.")
```

```
    lines.append("")
    lines.append("=" * 80)
    lines.append("DAFTAR SUDUT DIHEDRAL")
    lines.append("=" * 80)
    lines.append("No.  Atom-1       Atom-2       Atom-3       Atom-4       Dihedral
(derajat)")
    lines.append("-" * 80)
    if dihedrals:
        for number, dihedral in enumerate(dihedrals, start=1):
            lines.append(
          f"{number:3d}  "
          f"{dihedral['atom1']:3d}-{dihedral['symbol1']:<2s}     "
          f"{dihedral['atom2']:3d}-{dihedral['symbol2']:<2s}     "
          f"{dihedral['atom3']:3d}-{dihedral['symbol3']:<2s}     "
          f"{dihedral['atom4']:3d}-{dihedral['symbol4']:<2s}     "
          f"{dihedral['dihedral_degrees']:12.6f}"
        )
    else:
```

```
        lines.append("Tidak ada sudut dihedral terdeteksi.")

    lines.append("")
    lines.append("=" * 80)
    lines.append("MATRIKS JARAK ANTARATOM (Angstrom)")
    lines.append("=" * 80)
```

header = "Atom ".ljust(8) + "".join(f"{atom['index']:>10d}" for atom in atoms)

```
    lines.append(header)
    lines.append("-" * max(80, len(header)))
    for atom, row in zip(atoms, distance_matrix):
```

row_text = f"{atom['index']:3d}-{atom['symbol']:<2s} ".ljust(8)

row_text += "".join(f"{value:10.4f}" for value in row)

```
        lines.append(row_text)

    lines.append("")
    lines.append("=" * 80)
    lines.append("FILE HASIL YANG DIBUAT")
    lines.append("=" * 80)
    for label, filename in output_files.items():
        lines.append(f"{label:18s}: {filename}")

    lines.append("")
    lines.append("CATATAN")
    lines.append("- Format .xyz hanya menyimpan koordinat, bukan konektivitas ikatan.")
    lines.append("- Format .mol, .sdf, .pdb, dan .cml lebih tepat untuk visualisasi
struktur 3D.")
    lines.append("- Ikatan diinferensi dari jarak antaratom dan jari-jari kovalen.")
    lines.append("- Jika konektivitas visual belum sesuai, ubah --bond-scale atau --bond-
tolerance.")
    lines.append("")
```

Path(path).write_text("\n".join(lines), encoding="utf-8")

```
def save_final_structure_outputs(mol, mf, energy, method_name, method_info, basis_name,
pyscf_basis, args, output_dir, running_output_log_path=None):
```

atoms = get_atom_records(mol)

bonds = infer_bonds(atoms, bond_scale=args.bond_scale, bond_tolerance=args.bond_tolerance)

angles = infer_angles(atoms, bonds)

dihedrals = infer_dihedrals(atoms, bonds)

distance_matrix = calculate_distance_matrix(atoms)

prefix = make_output_prefix(method_name, basis_name, args.job)

metadata = {

"author": AUTHOR_DEFAULT,

"Telp.": AUTHOR_PHONE_DEFAULT,

"Nama molekul": MOLECULE_NAME,

"Rumus molekul": MOLECULE_FORMULA,

"SMILES": SMILES_DEFAULT,
"IUPAC-like name": IUPAC_NAME_DEFAULT,

"Catatan riset": RESEARCH_NOTE_DEFAULT,

```
        "Jumlah atom": mol.natm,
        "Jumlah elektron": mol.nelectron,
        "Muatan total": args.charge,
        "Spin PySCF": args.spin,
        "Metode": method_name,
        "Family metode": method_info["family"],
        "XC functional": method_info.get("xc"),
        "Dispersion": method_info.get("disp"),
        "Basis set": basis_name,
        "Basis PySCF": pyscf_basis,
        "Jenis perhitungan": args.job,
        "Energi total Hartree": f"{energy:.12f}",
        "SCF konvergen": bool(getattr(mf, "converged", False)),
        "File checkpoint": getattr(mf, "chkfile", ""),
        "Satuan koordinat": "Angstrom",
        "Bond scale": args.bond_scale,
        "Bond tolerance": args.bond_tolerance,
        "Waktu simpan": datetime.now().strftime("%Y-%m-%d %H:%M:%S"),
    }

    output_files = {
        "txt_coordinates": str(output_dir / f"{prefix}_koordinat_akhir.txt"),
        "xyz_structure": str(output_dir / f"{prefix}.xyz"),
        "mol_structure": str(output_dir / f"{prefix}.mol"),
        "sdf_structure": str(output_dir / f"{prefix}.sdf"),
        "pdb_structure": str(output_dir / f"{prefix}.pdb"),
        "cml_structure": str(output_dir / f"{prefix}.cml"),
        "json_structure": str(output_dir / f"{prefix}_struktur_lengkap.json"),
        "log_report": str(output_dir / f"{prefix}.log"),
    }

    if running_output_log_path:
        output_files["running_output_log"] = str(running_output_log_path)

    write_txt_coordinates(output_files["txt_coordinates"], atoms, metadata)
```

write_xyz_file(output_files["xyz_structure"], atoms, metadata)

write_mol_file(output_files["mol_structure"], atoms, bonds, title=MOLECULE_NAME)

write_sdf_file(output_files["sdf_structure"], atoms, bonds, metadata)

write_pdb_file(output_files["pdb_structure"], atoms, bonds, metadata)

write_cml_file(output_files["cml_structure"], atoms, bonds, metadata)

write_json_structure_file(output_files["json_structure"], atoms, bonds, angles, dihedrals, distance_matrix, metadata)

write_log_file(output_files["log_report"], atoms, bonds, angles, dihedrals, distance_matrix, metadata, output_files)

```
    return output_files, {
```

"atoms": atoms,

"bonds": bonds,

"angles": angles,

"dihedrals": dihedrals,

}

```
# ============================================================
```

14. INFORMASI ORBITAL

Subbab ini menguraikan # 14. INFORMASI ORBITAL. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================

def print_orbital_info(mf):
```

mo_energy = getattr(mf, "mo_energy", None)

mo_occ = getattr(mf, "mo_occ", None)

```
    if mo_energy is None or mo_occ is None:
        print("\nInformasi orbital tidak tersedia.")
```

return

```
    if isinstance(mo_energy, tuple) or isinstance(mo_occ, tuple):
        print("\nSistem terbuka-shell terdeteksi.")
        print("Pelaporan HOMO-LUMO sederhana dilewati.")
```

return

occupied = [i for i, occ in enumerate(mo_occ) if occ > 0]

virtual = [i for i, occ in enumerate(mo_occ) if occ == 0]

```
    if not occupied or not virtual:
        print("\nHOMO/LUMO tidak dapat ditentukan dari okupansi orbital.")
```

return

homo_index = max(occupied)

lumo_index = min(virtual)

homo_hartree = mo_energy[homo_index]

lumo_hartree = mo_energy[lumo_index]

gap_hartree = lumo_hartree - homo_hartree

hartree_to_ev = 27.211386245988

homo_ev = homo_hartree * hartree_to_ev

lumo_ev = lumo_hartree * hartree_to_ev

gap_ev = gap_hartree * hartree_to_ev

```
    print("\n============================================================")
    print("INFORMASI ORBITAL SEDERHANA")
    print("============================================================")
    print(f"HOMO index          : {homo_index}")
    print(f"LUMO index          : {lumo_index}")
    print(f"Energi HOMO         : {homo_hartree: .10f} Hartree")
    print(f"Energi LUMO         : {lumo_hartree: .10f} Hartree")
    print(f"Gap HOMO-LUMO       : {gap_hartree: .10f} Hartree")
    print(f"Energi HOMO         : {homo_ev: .6f} eV")
    print(f"Energi LUMO         : {lumo_ev: .6f} eV")
    print(f"Gap HOMO-LUMO       : {gap_ev: .6f} eV")
    print("============================================================")
```

```
# ============================================================
```

15. ARGUMEN TERMINAL

Subbab ini menguraikan # 15. ARGUMEN TERMINAL. Pembahasan difokuskan pada gagasan utama, alasan metodologis, langkah kerja, serta konsekuensi pembacaannya dalam konteks kimia komputasi.

```
# ============================================================
```

```
def parse_arguments():
```

parser = argparse.ArgumentParser(

description="Template generik perhitungan kimia komputasi molekul dengan PySCF. Default: OTH-01 Olaparib-Talazoparib Hybrid."

)

```
    parser.add_argument("--name", default=MOLECULE_NAME_DEFAULT, help="Nama molekul.")
    parser.add_argument("--formula", default=FORMULA_DEFAULT, help="Rumus molekul
opsional.")
```

```
    parser.add_argument("--xyz-file", default=None, help="File XYZ standar atau blok
koordinat XYZ.")
    parser.add_argument("--coord-file", default=None, help="File teks koordinat:
SimbolAtom X Y Z.")
```

```
    parser.add_argument("--method", default=METHOD_DEFAULT, help=f"Metode komputasi.
Default: {METHOD_DEFAULT}")
```

```
    parser.add_argument("--xc", default=None, help="Functional PySCF untuk --method
CUSTOM, contoh: 'm06-2x'.")
    parser.add_argument("--disp", default=None, help="Koreksi dispersi manual, contoh:
d3bj, d3zero, d4.")

    parser.add_argument("--basis", default=BASIS_DEFAULT, help=f"Basis set. Default:
{BASIS_DEFAULT}")
    parser.add_argument("--basis-custom", default=None, help="Nama basis PySCF untuk --
basis CUSTOM.")

    parser.add_argument("--charge", type=int, default=CHARGE_DEFAULT, help=f"Muatan total
molekul. Default: {CHARGE_DEFAULT}")
    parser.add_argument(
        "--spin",
        type=int,
        default=SPIN_DEFAULT,
        help=f"Spin PySCF = jumlah elektron alfa - beta. Default: {SPIN_DEFAULT}",
    )

    parser.add_argument(
        "--job",
        choices=["singlepoint", "optimize"],
        default=JOB_DEFAULT,
        help=f"Jenis perhitungan: singlepoint atau optimize. Default: {JOB_DEFAULT}",
    )
    parser.add_argument("--optimize", action="store_true", help="Alias lama untuk --job
optimize.")

    parser.add_argument("--grid", type=int, default=DFT_GRID_LEVEL_DEFAULT, help=f"Level
grid DFT. Default: {DFT_GRID_LEVEL_DEFAULT}")
    parser.add_argument("--conv-tol", type=float, default=CONV_TOL_DEFAULT,
help=f"Toleransi konvergensi SCF. Default: {CONV_TOL_DEFAULT}")
    parser.add_argument("--max-cycle", type=int, default=MAX_CYCLE_DEFAULT,
help=f"Maksimum iterasi SCF. Default: {MAX_CYCLE_DEFAULT}")
    parser.add_argument("--verbose", type=int, default=VERBOSE_DEFAULT, help=f"Level
output PySCF. Default: {VERBOSE_DEFAULT}")
    parser.add_argument("--density-fit", action="store_true",
default=USE_DENSITY_FITTING_DEFAULT, help="Aktifkan density fitting.")

    parser.add_argument("--output-dir", default=None, help="Direktori output. Jika kosong,
dibuat otomatis di hasil_pyscf/.")
    parser.add_argument("--bond-scale", type=float, default=BOND_SCALE_DEFAULT,
help=f"Faktor skala inferensi ikatan. Default: {BOND_SCALE_DEFAULT}")
    parser.add_argument("--bond-tolerance", type=float, default=BOND_TOLERANCE_DEFAULT,
help=f"Toleransi inferensi ikatan dalam Angstrom. Default: {BOND_TOLERANCE_DEFAULT}")

    parser.add_argument("--list", action="store_true", help="Tampilkan daftar metode,
basis set, dan contoh pemakaian.")

    return parser.parse_args()


# ============================================================
# 16. PROGRAM UTAMA
```

Bagian ini memuat # 16. PROGRAM UTAMA sebagai contoh implementasi. Pembaca sebaiknya membaca tujuan blok kerja terlebih dahulu sebelum menjalankan kode, kemudian membandingkan output yang diperoleh dengan parameter metode yang digunakan.

```
# ============================================================

def main():
    global MOLECULE_NAME, MOLECULE_FORMULA, BASIS_LABEL

    args = parse_arguments()

    if args.list:
        print_available_options()
        return

    if args.optimize:
        args.job = "optimize"

    validate_nonnegative_int(args.grid, "grid")
    validate_positive_float(args.conv_tol, "conv_tol")
    validate_positive_int(args.max_cycle, "max_cycle")
    validate_positive_float(args.bond_scale, "bond_scale")
    validate_nonnegative_int(args.verbose, "verbose")

    MOLECULE_NAME = args.name.strip() if args.name else MOLECULE_NAME_DEFAULT
    MOLECULE_FORMULA = args.formula.strip() if args.formula else "-"

    method_name, method_info = resolve_method(args)
    basis_name, pyscf_basis = resolve_basis(args)

    BASIS_LABEL = basis_name

    # Direktori output dan file running log dibuat sebelum PySCF mulai bekerja,
    # sehingga pesan proses SCF/DFT dan pesan error tetap tersimpan walaupun perhitungan gagal.
    output_dir = make_output_dir(args, method_name, basis_name)
    running_log_prefix = make_output_prefix(method_name, basis_name, args.job)
    running_output_log_path = output_dir / f"{running_log_prefix}_running_output.log"

    original_stdout = sys.stdout
    original_stderr = sys.stderr
    status_akhir = "SELESAI"

    with open(running_output_log_path, "w", encoding="utf-8", buffering=1) as running_log_file:
        sys.stdout = TeeOutput(original_stdout, running_log_file)
        sys.stderr = TeeOutput(original_stderr, running_log_file)
```

```
        try:
    write_running_log_header(
        log_path=running_output_log_path,
        args=args,
        method_name=method_name,
        basis_name=basis_name,
        pyscf_basis=pyscf_basis,
    )

    coordinate_atoms, coordinate_source = get_coordinate_atoms(args)
    atom_block = format_atom_block(coordinate_atoms)

    mol = build_molecule(
        atom_block=atom_block,
        pyscf_basis=pyscf_basis,
        charge=args.charge,
        spin=args.spin,
        verbose=args.verbose,
    )

    print_molecule_info(
        mol=mol,
        method_name=method_name,
        method_info=method_info,
        basis_name=basis_name,
        pyscf_basis=pyscf_basis,
        charge=args.charge,
        spin=args.spin,
        job=args.job,
        coordinate_source=coordinate_source,
    )

    print_coordinates(mol, "KOORDINAT AWAL MOLEKUL")

    mf = build_calculator(
        mol=mol,
```

```
        method_name=method_name,
        method_info=method_info,
        grid_level=args.grid,
        conv_tol=args.conv_tol,
        max_cycle=args.max_cycle,
        use_density_fitting=args.density_fit,
    )

    if args.job == "optimize":
        mol_opt = optimize_geometry(mf)
        print_coordinates(mol_opt, "KOORDINAT HASIL OPTIMASI GEOMETRI")
        mf = build_calculator(
            mol=mol_opt,
            method_name=method_name,
            method_info=method_info,
            grid_level=args.grid,
            conv_tol=args.conv_tol,
            max_cycle=args.max_cycle,
            use_density_fitting=args.density_fit,
        )

    print("\n============================================================")
    print("PERHITUNGAN ENERGI DIMULAI")
    print("============================================================")
    print(f"Nama molekul : {MOLECULE_NAME}")
    print(f"Metode       : {method_name}")
    print(f"XC           : {method_info.get('xc')}")
    print(f"Dispersion   : {method_info.get('disp')}")
    print(f"Basis set    : {basis_name} ({pyscf_basis})")
    print(f"Grid DFT     : {args.grid}")
    print(f"Conv tol     : {args.conv_tol}")
    print(f"Max cycle    : {args.max_cycle}")
    print(f"Density fit  : {args.density_fit}")
    print("============================================================\n")

    try:
        energy = mf.kernel()
    except Exception as exc:
        status_akhir = "GAGAL"
        print("\nERROR: Perhitungan PySCF gagal.")
        print(str(exc))
        if method_info.get("disp"):
            print("\nKemungkinan penyebab: koreksi dispersi belum tersedia pada
instalasi PySCF Anda.")
            print("Coba salah satu opsi berikut:")
            print("1. Jalankan tanpa dispersi:")
            print("   python oth01-olaparib-talazoparib-hybrid.py --method B3LYP -
-basis def2-SVP")
```

```
                    print("2. Instal dukungan dispersi sesuai lingkungan Python Anda,
misalnya:")
                    print("   pip install pyscf-dispersion simple-dftd3")
                print("\nPENTING:")
                print(f"Teks running/error sudah otomatis tersimpan di:
{running_output_log_path}")
```

sys.exit(1)

```
            print("\n============================================================")
            print("HASIL PERHITUNGAN")
            print("============================================================")
            print(f"Nama molekul       : {MOLECULE_NAME}")
            print(f"Rumus molekul      : {MOLECULE_FORMULA}")
            print(f"Metode             : {method_name}")
            print(f"Basis set          : {basis_name}")
            print(f"Energi total       : {energy: .12f} Hartree")
            print(f"SCF konvergen      : {mf.converged}")
            print(f"File checkpoint    : {mf.chkfile}")
```

```
            if not mf.converged:
                print("\nPERINGATAN:")
                print("SCF belum konvergen.")
                print("Coba salah satu langkah berikut:")
                print("1. Naikkan --max-cycle, misalnya --max-cycle 250")
                print("2. Longgarkan sementara --conv-tol, misalnya --conv-tol 1e-7")
                print("3. Gunakan basis lebih kecil untuk uji awal, misalnya STO-3G")
                print("4. Optimasi geometri awal di Avogadro atau jalankan --job
optimize")
```

```
            print("============================================================")
```

output_files, structure_summary = save_final_structure_outputs(

mol=mf.mol,

mf=mf,

energy=energy,

method_name=method_name,

method_info=method_info,

basis_name=basis_name,

pyscf_basis=pyscf_basis,

args=args,

output_dir=output_dir,

running_output_log_path=running_output_log_path,

)

```
            print("\n============================================================")
            print("FILE HASIL AKHIR BERHASIL DISIMPAN")
            print("============================================================")
            print(f"Direktori output            : {output_dir}")
            print(f"Jumlah ikatan terdeteksi    : {len(structure_summary['bonds'])}")
            print(f"Jumlah sudut terdeteksi     : {len(structure_summary['angles'])}")
            print(f"Jumlah dihedral terdeteksi  : {len(structure_summary['dihedrals'])}")
            print("")
            for label, filename in output_files.items():
                print(f"{label:18s}: {filename}")
            print("============================================================")
            print("Catatan:")
```

```
            print("- .txt berisi koordinat akhir molekul.")
            print("- .xyz berisi koordinat 3D standar.")
            print("- .mol/.sdf/.pdb/.cml dapat dipakai untuk visualisasi struktur 3D.")
            print("- .json dan .log menyimpan data ikatan, sudut, dihedral, jarak, dan
teks running.")
            print("================================================================")
```

print_orbital_info(mf)

```
            print("\nSELESAI.")

        except SystemExit:
            if status_akhir != "GAGAL":
```

status_akhir = "DIHENTIKAN"

raise

```
        except Exception:
```

status_akhir = "GAGAL"

```
            print("\nERROR TAK TERDUGA:", file=sys.stderr)
            import traceback
```

traceback.print_exc()

```
            print("\nPENTING:")
            print(f"Teks running/error sudah otomatis tersimpan di:
{running_output_log_path}")
```

raise

finally:

```
            try:
```

write_running_log_footer(running_output_log_path, status=status_akhir)

finally:

sys.stdout = original_stdout

sys.stderr = original_stderr

```
if __name__ == "__main__":
```

main()

BAB 8

PANDUAN PELAKSANAAN RISET S1 KIMIA KOMPUTASI

Pengantar Bab

Bab ini memberi panduan operasional untuk menjalankan riset kimia komputasi pada level S1. Materi diarahkan agar mahasiswa mampu menyusun alur kerja yang realistis, mulai dari formulasi masalah, persiapan struktur, optimasi awal, perhitungan DFT, docking, prediksi ADMET, hingga penulisan laporan.

Bagian ini penting karena keberhasilan riset tidak hanya ditentukan oleh software, tetapi oleh disiplin kerja: penamaan file, pencatatan log, pengujian input, validasi output, pengendalian kesalahan, dan konsistensi dalam menafsirkan data.

Setelah pengantar ini, pembahasan dilanjutkan dengan orientasi bab, peta kompetensi, uraian materi inti, contoh kerja, dan catatan metodologis yang diperlukan untuk menuntun pembaca dari konsep menuju praktik.

8.1 Prinsip Umum Riset Komputasi untuk Mahasiswa S1

Riset kimia komputasi untuk mahasiswa S1 perlu dirancang dengan keseimbangan antara kedalaman ilmiah dan keterjangkauan teknis. Topik yang terlalu ambisius sering kali terlihat menarik pada judul, tetapi sulit diselesaikan karena memerlukan server besar, validasi kompleks, atau dataset yang belum tersedia. Sebaliknya, topik yang terlalu sederhana dapat kehilangan nilai ilmiah karena hanya menjadi latihan menjalankan software.

Kriteria topik yang ideal adalah jelas targetnya, jelas senyawanya, jelas metode komputasinya, jelas parameter yang diukur, dan jelas kriteria keberhasilan. Misalnya, studi DFT dan docking pada satu kelompok senyawa hibrida lebih realistis daripada screening ribuan senyawa tanpa pipeline otomatis yang matang.

Mahasiswa perlu memahami bahwa keberhasilan riset komputasi tidak diukur dari banyaknya angka yang dihasilkan, tetapi dari kemampuan menjelaskan hubungan antara struktur molekul, sifat elektronik, interaksi target, dan kelayakan kandidat berdasarkan batasan metode.

8.2 Alur Kerja Standar

Alur kerja standar dimulai dengan formulasi masalah. Pertanyaan penelitian harus dapat dijawab oleh metode komputasi. Contoh pertanyaan yang baik adalah: apakah modifikasi substituen tertentu menurunkan celah HOMO-LUMO, meningkatkan momen dipol, memperbaiki interaksi hidrogen dengan residu target, atau menghasilkan profil drug-likeness yang lebih baik.

Setelah masalah jelas, mahasiswa menyiapkan struktur molekul. Struktur dapat bersumber dari SMILES, PubChem, atau desain manual, tetapi tetap harus distandarkan: valensi benar, muatan jelas, stereokimia diperiksa, protonasi sesuai konteks, dan konformer awal masuk akal.

Tahap berikutnya adalah optimasi awal. Untuk molekul organik menengah, optimasi awal dapat memakai RDKit, MMFF94, UFF, atau xTB. Setelah itu barulah dilakukan optimasi DFT dengan basis set yang realistis. Pendekatan bertahap ini menghemat waktu dan mengurangi risiko gagal konvergensi.

Jika penelitian melibatkan docking, protein perlu dipreparasi dengan cermat. Air kristal, ion, kofaktor, rantai protein, ligan asli, hidrogen, muatan, dan grid box harus dijelaskan. Validasi docking melalui redocking ligan ko-kristal sangat dianjurkan sebelum docking kandidat baru.

Analisis akhir menyatukan data. Energi DFT, parameter orbital, docking score, interaksi residu, descriptor, dan ADMET tidak boleh dibaca secara terpisah. Kandidat terbaik adalah kandidat yang konsisten pada beberapa indikator, bukan hanya kandidat dengan satu angka docking paling rendah.

8.3 Pembagian Waktu Enam Bulan

Bulan pertama digunakan untuk studi pustaka, pemilihan target, pemilihan senyawa induk, dan penyusunan proposal. Pada tahap ini mahasiswa harus sudah memiliki rumusan masalah, daftar kandidat awal, dan daftar software yang akan dipakai.

Bulan kedua difokuskan pada persiapan struktur molekul, standardisasi SMILES, generasi konformer, optimasi awal, dan dokumentasi file input. Semua file harus diberi nama sistematis agar mudah ditelusuri.

Bulan ketiga digunakan untuk perhitungan DFT atau semiempiris. Jika sumber daya terbatas, gunakan basis set menengah untuk screening dan basis set lebih tinggi untuk kandidat terbaik. Catat semua kegagalan konvergensi karena kegagalan juga bagian dari data metodologis.

Bulan keempat digunakan untuk docking, validasi redocking, analisis interaksi protein-ligan, serta visualisasi residu penting. Jangan mengubah parameter docking di tengah jalan tanpa alasan ilmiah yang terdokumentasi.

Bulan kelima digunakan untuk prediksi descriptor, drug-likeness, ADMET, integrasi data, pembuatan tabel hasil, dan interpretasi. Buat matriks keputusan agar pemilihan kandidat terbaik tidak subjektif.

Bulan keenam digunakan untuk penulisan skripsi, revisi, pembuatan lampiran kode, finalisasi gambar, pemeriksaan daftar pustaka, dan persiapan seminar hasil.

8.4 Format Logbook Komputasi

Logbook komputasi adalah catatan harian yang menjamin reproducibility. Setiap run perlu mencatat tanggal, nama file, environment, versi Python, library utama, metode, basis set, input, output, status run, durasi, dan catatan error.

Nama file sebaiknya konsisten. Misalnya, LAH01_B3LYP_def2SVP_opt.log untuk hibrida letrozole-anastrozole kandidat 01 dengan metode B3LYP dan basis def2-SVP. Nama yang konsisten memudahkan pencarian hasil ketika jumlah file bertambah.

Logbook juga perlu mencatat keputusan ilmiah. Jika kandidat dieliminasi karena struktur tidak stabil, docking gagal, atau ADMET buruk, alasan tersebut harus dicatat. Dengan cara ini, skripsi tidak hanya melaporkan hasil sukses, tetapi juga menunjukkan proses seleksi yang transparan.

8.5 Kesalahan Umum yang Harus Dihindari

Kesalahan pertama adalah menganggap docking score sebagai bukti aktivitas biologis. Docking hanya memberi prediksi pose dan skor berbasis fungsi penilaian tertentu. Aktivitas biologis tetap memerlukan validasi eksperimental.

Kesalahan kedua adalah membandingkan energi total DFT antar molekul yang ukuran dan komposisinya berbeda tanpa konteks. Energi total harus ditafsirkan hati-hati; untuk perbandingan struktur sejenis, energi relatif lebih bermakna.

Kesalahan ketiga adalah tidak melaporkan parameter komputasi. Tanpa functional, basis set, grid, kriteria konvergensi, dan versi software, penelitian sulit direplikasi.

Kesalahan keempat adalah memakai struktur 3D yang belum dioptimasi atau belum diperiksa valensinya. Input yang salah akan menghasilkan output yang tampak ilmiah tetapi sebenarnya tidak dapat dipercaya.

Kesalahan kelima adalah memasukkan terlalu banyak kandidat tanpa strategi seleksi. Untuk S1, lebih baik sedikit kandidat tetapi dianalisis mendalam daripada banyak kandidat tetapi hanya ditampilkan sebagai tabel angka.

BAB 9

STANDAR MUTU DATA, REPRODUCIBILITY, DAN PELAPORAN

Pengantar Bab

Bab ini menegaskan standar mutu data, reproducibility, dan pelaporan dalam riset kimia komputasi. Pembaca diarahkan untuk memahami bahwa angka hasil komputasi harus selalu dapat ditelusuri kembali ke input, metode, parameter, versi software, dan file keluaran yang digunakan.

Standar pelaporan yang baik membantu pembaca lain mengulang perhitungan, menilai kelayakan metode, serta membedakan hasil yang kuat dari klaim yang berlebihan. Karena itu, bab ini menjadi bagian pengaman akademik bagi seluruh rancangan riset dalam buku.

Setelah pengantar ini, pembahasan dilanjutkan dengan orientasi bab, peta kompetensi, uraian materi inti, contoh kerja, dan catatan metodologis yang diperlukan untuk menuntun pembaca dari konsep menuju praktik.

9.1 Reproducibility sebagai Syarat Ilmiah

Reproducibility berarti orang lain dapat mengulangi perhitungan dengan input, metode, dan parameter yang sama untuk memperoleh hasil yang sebanding. Dalam kimia komputasi, reproducibility sangat bergantung pada kelengkapan file input, versi software, struktur awal, pengaturan metode, dan dokumentasi output.

Setiap tabel hasil sebaiknya dapat ditelusuri ke file asal. Jika ada tabel HOMO-LUMO, maka nama file output, metode, basis set, dan status konvergensi harus jelas. Jika ada gambar pose docking, maka protein, ligan, grid box, dan parameter docking harus dapat dilacak.

Buku ini menempatkan kode program sebagai bagian dari naskah, bukan sekadar lampiran teknis. Hal ini penting karena skrip adalah metode operasional penelitian. Dengan membaca skrip, pembaca dapat mengetahui bagaimana data dihasilkan.

9.2 Standar Minimal Pelaporan DFT

Pelaporan DFT minimal mencakup nama molekul, rumus molekul, koordinat atau sumber struktur, functional, koreksi dispersi jika digunakan, basis set, muatan, spin, kriteria konvergensi, grid, jenis perhitungan, status optimasi, energi akhir, dan parameter elektronik yang dianalisis.

Jika optimasi geometri dilakukan, laporan perlu mencantumkan apakah struktur merupakan minimum. Idealnya, verifikasi minimum dilakukan melalui analisis frekuensi untuk memastikan tidak ada frekuensi imajiner. Jika analisis frekuensi tidak dilakukan karena keterbatasan sumber daya, keterbatasan tersebut harus dinyatakan.

Parameter HOMO, LUMO, dan gap dapat digunakan sebagai indikator reaktivitas, tetapi tidak boleh ditafsirkan secara berlebihan. Hubungan dengan aktivitas biologis harus dijelaskan sebagai dugaan struktur-elektronik, bukan bukti langsung.

9.3 Standar Minimal Pelaporan Docking

Pelaporan docking minimal mencakup identitas protein, sumber struktur, kode PDB jika ada, resolusi struktur, ligan asli, persiapan protein, pengaturan protonasi, ukuran dan pusat grid box, software docking, jumlah run, parameter scoring, dan validasi redocking.

Docking yang baik tidak hanya menampilkan skor. Interaksi hidrogen, hidrofobik, π-π, kation-π, garam, dan kontak residu penting harus dibahas. Visualisasi 2D dan 3D membantu pembaca memahami apakah pose masuk akal secara kimia.

Perbandingan antar kandidat harus konsisten. Jika kandidat A dan B dibandingkan, keduanya harus memakai protein, grid, parameter, dan protokol yang sama. Perubahan parameter harus dijelaskan.

9.4 Integrasi DFT, Docking, dan ADMET

Integrasi data adalah bagian paling menentukan dalam riset hibrida. DFT menjelaskan aspek elektronik dan kestabilan struktur. Docking menjelaskan kecenderungan interaksi dengan target. ADMET dan drug-likeness memberi gambaran kelayakan farmakokinetik awal. Ketiganya harus dibaca bersama.

Kandidat dengan docking score bagus tetapi ADMET buruk tidak otomatis menjadi kandidat terbaik. Kandidat dengan gap elektronik kecil tetapi struktur tidak stabil juga harus dikaji ulang. Karena itu, disarankan membuat skor keputusan bertingkat yang mempertimbangkan beberapa parameter.

Dalam skripsi, integrasi data dapat disajikan melalui tabel matriks. Kolom dapat mencakup kode senyawa, energi, HOMO, LUMO, gap, momen dipol, docking score, jumlah ikatan hidrogen, residu kunci, logP, TPSA, HBD, HBA, pelanggaran Lipinski, dan status rekomendasi.

9.5 Etika Interpretasi Hasil

Riset komputasi harus jujur terhadap keterbatasan. Prediksi in silico tidak sama dengan aktivitas in vitro atau in vivo. Oleh karena itu, kalimat seperti “senyawa ini terbukti sebagai obat” harus dihindari jika belum ada data eksperimental.

Bahasa yang lebih tepat adalah “diprediksi memiliki potensi”, “menunjukkan afinitas komputasi yang lebih baik”, “layak dipertimbangkan untuk validasi lebih lanjut”, atau “menjadi kandidat awal berdasarkan indikator komputasi”.

Etika interpretasi juga mencakup keterbukaan terhadap hasil negatif. Kandidat yang gagal konvergensi, memiliki pose docking tidak masuk akal, atau melanggar banyak parameter ADMET tetap bernilai sebagai data seleksi.

BAB 10

PANDUAN PENULISAN SKRIPSI BERBASIS BUKU INI

Pengantar Bab

Bab ini memberikan panduan penulisan skripsi berbasis materi buku. Fokusnya adalah bagaimana data komputasi disusun menjadi argumentasi ilmiah yang runtut, mulai dari pendahuluan, tinjauan pustaka, metode, hasil, pembahasan, kesimpulan, hingga lampiran teknis.

Mahasiswa perlu memastikan bahwa pembahasan tidak sekadar memindahkan tabel hasil, tetapi menjelaskan makna kimianya. Setiap klaim harus merujuk pada data, metode, keterbatasan, dan hubungan antara struktur molekul, parameter elektronik, docking, serta prediksi sifat obat.

Setelah pengantar ini, pembahasan dilanjutkan dengan orientasi bab, peta kompetensi, uraian materi inti, contoh kerja, dan catatan metodologis yang diperlukan untuk menuntun pembaca dari konsep menuju praktik.

10.1 Struktur Bab Skripsi

Bab pendahuluan skripsi perlu menunjukkan masalah penelitian, urgensi komputasi, target kanker payudara, alasan pemilihan senyawa induk, dan celah penelitian. Hindari latar belakang yang terlalu umum. Setiap paragraf sebaiknya mengarah pada kebutuhan penelitian yang dilakukan.

Bab tinjauan pustaka menjelaskan kanker payudara sesuai konteks target molekuler, senyawa induk, prinsip DFT, docking, ADMET, serta penelitian terdahulu. Tinjauan pustaka bukan kumpulan definisi, melainkan landasan untuk membangun hipotesis penelitian.

Bab metode menjelaskan workflow secara operasional. Pembaca harus dapat mengulang penelitian hanya dengan membaca bab metode dan lampiran. Semua software, parameter, dan kriteria seleksi harus disebutkan.

Bab hasil dan pembahasan menyajikan data secara bertahap: struktur kandidat, hasil optimasi, parameter elektronik, docking, interaksi target, ADMET, dan integrasi kandidat terbaik. Setiap tabel perlu dibahas, bukan hanya ditempel.

Bab kesimpulan menjawab rumusan masalah. Kesimpulan harus singkat, berbasis data, dan tidak melebihi cakupan penelitian. Saran dapat diarahkan pada validasi eksperimental, MD lebih panjang, perhitungan free energy, atau pengembangan dataset.

10.2 Pola Paragraf Pembahasan

Paragraf pembahasan yang baik biasanya memiliki empat unsur: pernyataan hasil, penjelasan kimia, perbandingan dengan kandidat lain atau literatur, dan implikasi terhadap hipotesis. Pola ini membantu mahasiswa menghindari pembahasan yang hanya mengulang angka tabel.

Contoh pola: “Kandidat LAH-03 memiliki gap HOMO-LUMO lebih kecil dibanding LAH-01, yang menunjukkan kecenderungan reaktivitas elektronik lebih tinggi. Penurunan gap ini dapat berkaitan dengan perluasan konjugasi pada fragmen aromatik. Namun, nilai docking LAH-03 perlu dibaca bersama profil ADMET karena peningkatan lipofilisitas dapat memengaruhi kelayakan farmakokinetik.”

Dengan pola tersebut, pembahasan menjadi lebih argumentatif. Mahasiswa tidak hanya menyebut hasil, tetapi menjelaskan makna dan batasannya.

10.3 Penyajian Tabel dan Gambar

Tabel harus diberi judul yang informatif. Hindari judul seperti “Tabel Hasil” tanpa konteks. Gunakan judul seperti “Parameter elektronik kandidat hibrida hasil optimasi B3LYP-D3BJ/def2-SVP”.

Gambar struktur molekul harus jelas, beresolusi memadai, dan diberi keterangan. Jika gambar pose docking digunakan, residu penting dan interaksi perlu ditampilkan. Jika spektrum atau grafik dibuat, sumbu, satuan, dan legenda harus terbaca.

Setiap tabel dan gambar harus dirujuk dalam teks. Jangan memasukkan tabel atau gambar tanpa pembahasan. Fungsi visual dalam skripsi adalah memperjelas argumen, bukan sekadar memperbanyak halaman.

10.4 Lampiran Kode Program

Kode program sebaiknya dimasukkan sebagai lampiran atau repositori pendamping. Jika skripsi dicetak, lampiran kode perlu diberi nomor, nama file, dan penjelasan fungsi. Jika kode terlalu panjang, bagian inti dapat ditampilkan di lampiran dan file lengkap disediakan secara digital.

Kode harus dibersihkan dari path lokal yang tidak relevan, token pribadi, atau data sensitif. Komentar di dalam kode perlu membantu pembaca memahami fungsi bagian program.

Versi final kode harus sama dengan versi yang menghasilkan data dalam skripsi. Jika ada perubahan setelah data dihitung, perbedaan versi harus dicatat.

BAB 11

MODUL PRAKTIKUM TERSTRUKTUR BERBASIS BUKU INI

Pengantar Bab

Bab ini menyusun modul praktikum terstruktur agar buku dapat langsung digunakan dalam perkuliahan, bimbingan skripsi, atau pelatihan kimia komputasi. Setiap pertemuan dirancang menghasilkan produk kerja yang konkret dan dapat dinilai.

Prinsip modul ini adalah bertahap. Mahasiswa tidak langsung diarahkan ke perhitungan besar, tetapi terlebih dahulu memahami ekosistem software, membangun struktur, memvalidasi input, menjalankan perhitungan kecil, membaca output, lalu menyusun laporan ilmiah.

Setelah pengantar ini, pembahasan dilanjutkan dengan orientasi bab, peta kompetensi, uraian materi inti, contoh kerja, dan catatan metodologis yang diperlukan untuk menuntun pembaca dari konsep menuju praktik.

Modul praktikum ini ditambahkan agar buku dapat langsung digunakan dalam perkuliahan, lokakarya, atau bimbingan skripsi. Setiap pertemuan dirancang untuk menghubungkan konsep kimia, keterampilan Python, dan keluaran riset yang konkret. Dosen dapat memilih seluruh modul atau hanya mengambil bagian yang sesuai dengan fasilitas komputasi yang tersedia.

Prinsip penyusunan modul adalah bertahap. Mahasiswa tidak langsung diminta menjalankan perhitungan besar, tetapi terlebih dahulu belajar memahami struktur molekul, format file, environment Python, validasi input, dan cara membaca output. Setelah itu mahasiswa masuk ke DFT, docking, descriptor, ADMET, integrasi data, dan penulisan laporan.

Setiap praktikum sebaiknya menghasilkan produk kecil. Produk tersebut dapat berupa tabel library, struktur 3D, file XYZ, hasil optimasi, log running, grafik parameter elektronik, tabel docking, matriks keputusan kandidat, atau paragraf pembahasan. Produk kecil ini kemudian digabungkan menjadi laporan akhir.

Pertemuan 1: Orientasi Riset Kimia Komputasi dan Ekosistem Python

Tujuan: mahasiswa mampu memahami dan mempraktikkan bagian spesifik dari workflow riset komputasi sesuai urutan pembelajaran.

Kegiatan: Mahasiswa mengenali posisi kimia komputasi dalam riset modern, memahami hubungan antara struktur molekul dan data numerik, serta membaca daftar library dalam Bab 1. Kegiatan utama adalah mengelompokkan library berdasarkan bidang: kimia kuantum, cheminformatics, dinamika molekul, material, visualisasi, machine learning, dan workflow. Luaran pertemuan berupa tabel ringkas pilihan library minimal untuk proyek S1.

Produk akhir: catatan praktikum, file kerja, tabel hasil, dan refleksi singkat yang dapat digunakan sebagai bahan penyusunan skripsi.

Catatan penguatan: dosen perlu menekankan validasi, kerapian nama file, kelengkapan logbook, dan kehati-hatian dalam menafsirkan hasil komputasi.

Pertemuan 2: Instalasi Environment dan Manajemen Dependency

Tujuan: mahasiswa mampu memahami dan mempraktikkan bagian spesifik dari workflow riset komputasi sesuai urutan pembelajaran.

Kegiatan: Mahasiswa mempraktikkan konsep environment terpisah. Dosen menjelaskan mengapa semua paket berat tidak ideal dimasukkan ke satu environment. Kegiatan meliputi membaca skrip instalasi, memahami fungsi Conda, mamba, pip, dan kernel Jupyter. Luaran berupa catatan instalasi, daftar environment, dan bukti import library utama.

Produk akhir: catatan praktikum, file kerja, tabel hasil, dan refleksi singkat yang dapat digunakan sebagai bahan penyusunan skripsi.

Catatan penguatan: dosen perlu menekankan validasi, kerapian nama file, kelengkapan logbook, dan kehati-hatian dalam menafsirkan hasil komputasi.

Pertemuan 3: Representasi Molekul dengan SMILES, XYZ, SDF, dan PDB

Tujuan: mahasiswa mampu memahami dan mempraktikkan bagian spesifik dari workflow riset komputasi sesuai urutan pembelajaran.

Kegiatan: Mahasiswa membaca daftar senyawa anti-kanker payudara dan mempelajari perbedaan representasi molekul. SMILES digunakan untuk input ringkas, XYZ untuk koordinat 3D, SDF untuk struktur dengan metadata, dan PDB untuk biomolekul. Luaran berupa konversi satu senyawa dari SMILES menjadi struktur 3D serta catatan potensi masalah stereokimia atau protonasi.

Produk akhir: catatan praktikum, file kerja, tabel hasil, dan refleksi singkat yang dapat digunakan sebagai bahan penyusunan skripsi.

Catatan penguatan: dosen perlu menekankan validasi, kerapian nama file, kelengkapan logbook, dan kehati-hatian dalam menafsirkan hasil komputasi.

Pertemuan 4: Generasi Konformer dan Optimasi Awal

Tujuan: mahasiswa mampu memahami dan mempraktikkan bagian spesifik dari workflow riset komputasi sesuai urutan pembelajaran.

Kegiatan: Mahasiswa mempelajari bahwa struktur 3D awal sangat memengaruhi hasil komputasi. Kegiatan meliputi pembuatan konformer, minimisasi dengan UFF atau MMFF94, dan pemilihan struktur awal yang paling masuk akal. Luaran berupa file XYZ kandidat dan tabel energi relatif optimasi awal.

Produk akhir: catatan praktikum, file kerja, tabel hasil, dan refleksi singkat yang dapat digunakan sebagai bahan penyusunan skripsi.

Catatan penguatan: dosen perlu menekankan validasi, kerapian nama file, kelengkapan logbook, dan kehati-hatian dalam menafsirkan hasil komputasi.

Pertemuan 5: Dasar Perhitungan DFT dengan PySCF

Tujuan: mahasiswa mampu memahami dan mempraktikkan bagian spesifik dari workflow riset komputasi sesuai urutan pembelajaran.

Kegiatan: Mahasiswa membaca bagian metode dan skrip PySCF. Dosen menjelaskan konsep molekul, basis set, muatan, spin, SCF, energi elektronik, dan kriteria konvergensi. Luaran berupa

perhitungan single-point sederhana pada molekul kecil sebelum mahasiswa berpindah ke senyawa hibrida yang lebih besar.

Produk akhir: catatan praktikum, file kerja, tabel hasil, dan refleksi singkat yang dapat digunakan sebagai bahan penyusunan skripsi.

Catatan penguatan: dosen perlu menekankan validasi, kerapian nama file, kelengkapan logbook, dan kehati-hatian dalam menafsirkan hasil komputasi.

Pertemuan 6: Optimasi Geometri dan Validasi Struktur

Tujuan: mahasiswa mampu memahami dan mempraktikkan bagian spesifik dari workflow riset komputasi sesuai urutan pembelajaran.

Kegiatan: Mahasiswa mempelajari perbedaan antara single-point energy dan optimasi geometri. Kegiatan meliputi menjalankan optimasi, membaca status konvergensi, memeriksa perubahan panjang ikatan, dan menyimpan struktur akhir. Luaran berupa struktur teroptimasi, log running, dan ringkasan perubahan geometri.

Produk akhir: catatan praktikum, file kerja, tabel hasil, dan refleksi singkat yang dapat digunakan sebagai bahan penyusunan skripsi.

Catatan penguatan: dosen perlu menekankan validasi, kerapian nama file, kelengkapan logbook, dan kehati-hatian dalam menafsirkan hasil komputasi.

Pertemuan 7: Analisis Orbital, HOMO-LUMO Gap, dan Momen Dipol

Tujuan: mahasiswa mampu memahami dan mempraktikkan bagian spesifik dari workflow riset komputasi sesuai urutan pembelajaran.

Kegiatan: Mahasiswa menghubungkan data orbital dengan interpretasi reaktivitas elektronik. Kegiatan meliputi mengekstrak energi HOMO, LUMO, gap, dan momen dipol. Luaran berupa tabel parameter elektronik dan paragraf pembahasan singkat yang membedakan data komputasi dari klaim biologis.

Produk akhir: catatan praktikum, file kerja, tabel hasil, dan refleksi singkat yang dapat digunakan sebagai bahan penyusunan skripsi.

Catatan penguatan: dosen perlu menekankan validasi, kerapian nama file, kelengkapan logbook, dan kehati-hatian dalam menafsirkan hasil komputasi.

Pertemuan 8: Preparasi Protein dan Validasi Docking

Tujuan: mahasiswa mampu memahami dan mempraktikkan bagian spesifik dari workflow riset komputasi sesuai urutan pembelajaran.

Kegiatan: Mahasiswa mempelajari bahwa docking yang baik dimulai dari protein yang benar. Kegiatan meliputi membaca target, memahami ligan ko-kristal, menentukan grid box, menambahkan hidrogen, dan melakukan redocking jika memungkinkan. Luaran berupa protokol preparasi protein dan catatan validasi docking.

Produk akhir: catatan praktikum, file kerja, tabel hasil, dan refleksi singkat yang dapat digunakan sebagai bahan penyusunan skripsi.

Catatan penguatan: dosen perlu menekankan validasi, kerapian nama file, kelengkapan logbook, dan kehati-hatian dalam menafsirkan hasil komputasi.

Pertemuan 9: Molecular Docking Kandidat Hibrida

Tujuan: mahasiswa mampu memahami dan mempraktikkan bagian spesifik dari workflow riset komputasi sesuai urutan pembelajaran.

Kegiatan: Mahasiswa menjalankan docking pada kandidat terpilih. Dosen menekankan konsistensi parameter dan pentingnya memeriksa pose, bukan hanya skor. Luaran berupa tabel docking score, pose terbaik, interaksi residu, dan visualisasi 2D/3D.

Produk akhir: catatan praktikum, file kerja, tabel hasil, dan refleksi singkat yang dapat digunakan sebagai bahan penyusunan skripsi.

Catatan penguatan: dosen perlu menekankan validasi, kerapian nama file, kelengkapan logbook, dan kehati-hatian dalam menafsirkan hasil komputasi.

Pertemuan 10: Descriptor, Drug-Likeness, dan ADMET Awal

Tujuan: mahasiswa mampu memahami dan mempraktikkan bagian spesifik dari workflow riset komputasi sesuai urutan pembelajaran.

Kegiatan: Mahasiswa menghitung descriptor sederhana seperti berat molekul, logP, HBD, HBA, TPSA, rotatable bond, dan aturan Lipinski. Kegiatan dilanjutkan dengan membaca prediksi ADMET awal secara hati-hati. Luaran berupa tabel descriptor dan catatan kandidat yang perlu dieliminasi.

Produk akhir: catatan praktikum, file kerja, tabel hasil, dan refleksi singkat yang dapat digunakan sebagai bahan penyusunan skripsi.

Catatan penguatan: dosen perlu menekankan validasi, kerapian nama file, kelengkapan logbook, dan kehati-hatian dalam menafsirkan hasil komputasi.

Pertemuan 11: Integrasi Data dan Matriks Keputusan

Tujuan: mahasiswa mampu memahami dan mempraktikkan bagian spesifik dari workflow riset komputasi sesuai urutan pembelajaran.

Kegiatan: Mahasiswa menggabungkan hasil DFT, docking, dan ADMET. Setiap parameter diberi bobot sesuai tujuan penelitian. Dosen mengarahkan agar mahasiswa tidak memilih kandidat terbaik hanya berdasarkan satu angka. Luaran berupa matriks keputusan dan rekomendasi kandidat prioritas.

Produk akhir: catatan praktikum, file kerja, tabel hasil, dan refleksi singkat yang dapat digunakan sebagai bahan penyusunan skripsi.

Catatan penguatan: dosen perlu menekankan validasi, kerapian nama file, kelengkapan logbook, dan kehati-hatian dalam menafsirkan hasil komputasi.

Pertemuan 12: Visualisasi Data dan Pembuatan Gambar Publikasi

Tujuan: mahasiswa mampu memahami dan mempraktikkan bagian spesifik dari workflow riset komputasi sesuai urutan pembelajaran.

Kegiatan: Mahasiswa membuat grafik HOMO-LUMO gap, diagram perbandingan docking score, visualisasi struktur, dan skema alur workflow. Luaran berupa gambar siap laporan dengan caption yang informatif.

Produk akhir: catatan praktikum, file kerja, tabel hasil, dan refleksi singkat yang dapat digunakan sebagai bahan penyusunan skripsi.

Catatan penguatan: dosen perlu menekankan validasi, kerapian nama file, kelengkapan logbook, dan kehati-hatian dalam menafsirkan hasil komputasi.

Pertemuan 13: Penyusunan Bab Metode dan Lampiran Kode

Tujuan: mahasiswa mampu memahami dan mempraktikkan bagian spesifik dari workflow riset komputasi sesuai urutan pembelajaran.

Kegiatan: Mahasiswa menerjemahkan workflow menjadi uraian metode skripsi. Kegiatan meliputi menulis parameter komputasi, menjelaskan software, membuat tabel input-output, dan menyiapkan lampiran kode. Luaran berupa draf Bab Metode dan daftar lampiran.

Produk akhir: catatan praktikum, file kerja, tabel hasil, dan refleksi singkat yang dapat digunakan sebagai bahan penyusunan skripsi.

Catatan penguatan: dosen perlu menekankan validasi, kerapian nama file, kelengkapan logbook, dan kehati-hatian dalam menafsirkan hasil komputasi.

Pertemuan 14: Pembahasan Hasil dan Simulasi Seminar

Tujuan: mahasiswa mampu memahami dan mempraktikkan bagian spesifik dari workflow riset komputasi sesuai urutan pembelajaran.

Kegiatan: Mahasiswa menyusun pembahasan berbasis data dan berlatih menjawab pertanyaan penguji. Kegiatan meliputi membaca tabel, menjelaskan keterbatasan, menyusun kesimpulan, dan membuat slide ringkas. Luaran berupa draf pembahasan dan bahan presentasi.

Produk akhir: catatan praktikum, file kerja, tabel hasil, dan refleksi singkat yang dapat digunakan sebagai bahan penyusunan skripsi.

Catatan penguatan: dosen perlu menekankan validasi, kerapian nama file, kelengkapan logbook, dan kehati-hatian dalam menafsirkan hasil komputasi.

BAB 12

BANK TUGAS, PERTANYAAN DISKUSI, DAN LATIHAN ANALISIS

Pengantar Bab

Bab ini menyediakan bank tugas, pertanyaan diskusi, dan latihan analisis untuk memperkuat pemahaman. Materi dapat dipakai sebagai kuis, tugas individu, diskusi kelompok, atau simulasi pertanyaan penguji skripsi.

Latihan dalam bab ini menekankan kemampuan menjelaskan alasan ilmiah, bukan sekadar menyebut jawaban singkat. Mahasiswa didorong menghubungkan konsep kimia, pemilihan metode, keterbatasan komputasi, dan etika interpretasi hasil.

Setelah pengantar ini, pembahasan dilanjutkan dengan orientasi bab, peta kompetensi, uraian materi inti, contoh kerja, dan catatan metodologis yang diperlukan untuk menuntun pembaca dari konsep menuju praktik.

Bab ini menyediakan bahan evaluasi yang dapat digunakan untuk tugas individu, diskusi kelompok, kuis, ujian praktikum, atau bimbingan skripsi. Pertanyaan disusun agar mahasiswa tidak hanya menghafal istilah, tetapi juga mampu menjelaskan alasan metodologis dan menilai kualitas data komputasi.

12.1 Tugas Pemahaman Library Python

1. Jelaskan perbedaan fungsi RDKit, Open Babel, PySCF, Psi4, ASE, OpenMM, MDAnalysis, pymatgen, cclib, dan py3Dmol dalam satu workflow riset molekul.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

2. Buatlah tabel minimal lima library yang paling diperlukan untuk riset senyawa hibrida anti-kanker payudara tingkat S1, lalu jelaskan alasan pemilihannya.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

3. Mengapa environment Conda sebaiknya dipisah antara kimia kuantum, molecular dynamics, material, dan machine learning?

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

4. Jelaskan risiko dependency conflict jika semua library berat dipasang dalam satu environment tunggal.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

5. Pilih satu library visualisasi molekul dan jelaskan bagaimana library tersebut membantu validasi struktur input.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

12.2 Tugas Representasi Molekul

6. Pilih satu SMILES dari daftar senyawa kecil, kemudian jelaskan gugus fungsi utama yang terlihat dari struktur tersebut.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

7. Bandingkan kelebihan dan kekurangan SMILES, XYZ, SDF, dan PDB sebagai format input komputasi.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

8. Mengapa stereokimia penting ketika molekul akan dipakai untuk docking atau DFT?

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

9. Jelaskan bagaimana kesalahan protonasi dapat mengubah hasil docking.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

10. Buat alur kerja sederhana dari nama molekul menuju struktur 3D siap optimasi.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

12.3 Tugas DFT dan Analisis Elektronik

11. Jelaskan perbedaan single-point energy dan optimasi geometri.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

12. Mengapa basis set memengaruhi akurasi dan waktu komputasi?

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

13. Bagaimana cara membaca hasil SCF yang konvergen dan tidak konvergen?

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

14. Jelaskan makna HOMO, LUMO, dan HOMO-LUMO gap dalam konteks reaktivitas molekul.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

15. Mengapa energi total tidak boleh dibandingkan sembarangan antara molekul dengan komposisi atom berbeda?

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

16. Jelaskan apa yang dimaksud dengan momen dipol dan bagaimana parameter ini berkaitan dengan polaritas molekul.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

17. Buat contoh paragraf pembahasan yang menghubungkan gap elektronik dan kemungkinan reaktivitas kandidat hibrida.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

12.4 Tugas Docking dan Interaksi Protein-Ligan

18. Mengapa validasi redocking penting sebelum docking kandidat baru?

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

19. Jelaskan arti grid box dan akibat jika grid box terlalu kecil atau tidak tepat pada sisi aktif.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

20. Mengapa docking score tidak dapat dianggap sebagai bukti aktivitas biologis?

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

21. Jelaskan jenis interaksi protein-ligan yang perlu dibahas selain skor docking.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

22. Buat tabel interpretasi yang berisi docking score, residu kunci, jenis interaksi, dan catatan kelayakan pose.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

12.5 Tugas Integrasi Data dan Kesimpulan

23. Buat matriks keputusan kandidat terbaik dengan menggabungkan hasil DFT, docking, dan ADMET.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

24. Jelaskan bagaimana menangani kandidat dengan docking score baik tetapi profil ADMET buruk.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

25. Mengapa kesimpulan penelitian harus dibatasi pada cakupan in silico?

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

26. Susun tiga kalimat kesimpulan yang benar secara ilmiah dan tidak berlebihan.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

27. Tuliskan lima keterbatasan umum riset komputasi yang perlu dicantumkan dalam skripsi.

Petunjuk jawaban: berikan alasan ilmiah, sebutkan parameter yang relevan, dan hubungkan jawaban dengan workflow penelitian di buku ini.

LAMPIRAN A
CHECK-LIST PROPOSAL RISET KIMIA KOMPUTASI

1. Judul mencantumkan objek molekul, metode utama, dan target penelitian.
2. Latar belakang menjelaskan masalah kanker payudara sesuai target molekuler yang dipilih.
3. Rumusan masalah dapat dijawab dengan metode komputasi yang tersedia.
4. Tujuan penelitian sejajar dengan rumusan masalah.
5. Hipotesis tidak melampaui cakupan prediksi in silico.
6. Kebaruan dijelaskan sebagai kombinasi scaffold, target, metode, atau workflow yang spesifik.
7. Batasan penelitian menyebut jumlah kandidat, metode, basis set, target protein, dan jenis prediksi ADMET.
8. Metode menjelaskan preparasi ligan dan protein secara operasional.
9. Parameter DFT dan docking disebutkan lengkap.
10. Kriteria kandidat terbaik dijelaskan sebelum hasil dianalisis.
11. Risiko penelitian dan strategi mitigasi ditulis realistis.
12. Luaran penelitian dapat dicapai dalam durasi S1.

LAMPIRAN B

FORMAT LOGBOOK HARIAN KOMPUTASI

Tanggal: .....

Nama peneliti: .....

Nama file input: .....

Nama file output: .....

Environment Conda: .....

Versi Python: .....

Library utama: .....

Metode: .....

Basis set: .....

Muatan dan spin: .....

Status perhitungan: .....

Durasi: .....

Energi akhir: .....

Parameter utama: .....

Catatan error: .....

Keputusan tindak lanjut: .....

LAMPIRAN C

RUBRIK PENILAIAN MINI RISET KIMIA KOMPUTASI

Perumusan masalah: Masalah jelas, spesifik, dan dapat dijawab dengan komputasi. Bobot 20%.

Kesesuaian metode: Metode, basis set, parameter docking, dan ADMET sesuai tujuan. Bobot 20%.

Kualitas data: Input benar, output lengkap, logbook tertata, dan status konvergensi jelas. Bobot 20%.

Analisis kimia: Pembahasan menghubungkan struktur, sifat elektronik, interaksi, dan kelayakan kandidat. Bobot 25%.

Pelaporan: Tabel, gambar, kode, dan kesimpulan disajikan rapi serta dapat direplikasi. Bobot 15%.

LAMPIRAN D

GLOSARIUM

A

Ab initio
Metode kimia komputasi yang menghitung sifat molekul berdasarkan prinsip dasar mekanika kuantum tanpa bergantung sepenuhnya pada parameter empiris. Contohnya Hartree–Fock, MP2, dan beberapa metode pasca-HF.

ADMET
Singkatan dari *Absorption, Distribution, Metabolism, Excretion, and Toxicity*. ADMET digunakan untuk memperkirakan sifat farmakokinetik dan toksisitas kandidat senyawa obat secara komputasi.

Adsorpsi
Proses melekatnya atom, ion, atau molekul pada permukaan suatu material. Dalam kimia komputasi, adsorpsi sering dikaji pada permukaan katalis, material, atau nanopartikel.

Afiliasi molekuler
Kecenderungan suatu molekul untuk berinteraksi dengan molekul lain, permukaan, atau target biologis tertentu.

Algoritma
Urutan langkah logis dan sistematis untuk menyelesaikan masalah. Dalam riset kimia komputasi, algoritma digunakan untuk optimasi geometri, pencarian konformer, docking, machine learning, dan analisis data.

Alur kerja komputasi
Rangkaian prosedur penelitian komputasi, mulai dari penyiapan struktur, pemilihan metode, perhitungan, analisis, validasi, hingga pelaporan.

Anastrozole
Senyawa inhibitor aromatase yang digunakan sebagai model dalam studi anti-kanker payudara. Dalam buku ini, anastrozole dapat dijadikan senyawa induk untuk desain hibrida dan analisis komputasi.

Anti-kanker payudara
Istilah yang merujuk pada senyawa, mekanisme, target, atau pendekatan terapi yang berkaitan dengan penghambatan perkembangan kanker payudara. Dalam riset komputasi, istilah ini harus dipahami sebagai potensi awal yang memerlukan validasi eksperimental.

Aromatase
Enzim yang berperan dalam biosintesis estrogen. Aromatase menjadi salah satu target penting dalam terapi kanker payudara hormon-positif.

ASE
Singkatan dari *Atomic Simulation Environment*. Library Python untuk membangun struktur atomistik, menjalankan kalkulasi, mengelola optimasi geometri, dan menghubungkan berbagai kalkulator kimia komputasi.

B

Basis set
Kumpulan fungsi matematika yang digunakan untuk merepresentasikan orbital atom atau orbital molekul dalam perhitungan kimia kuantum.

Basis Set Exchange
Sumber dan library untuk memperoleh, mengonversi, dan memformat basis set bagi berbagai program kimia kuantum.

Binding affinity
Ukuran kecenderungan ligan berikatan dengan target protein. Dalam docking, sering dinyatakan melalui skor ikatan atau energi ikatan prediktif.

Binding energy
Energi yang diasosiasikan dengan interaksi antara ligan dan reseptor. Nilai lebih negatif sering ditafsirkan sebagai interaksi yang lebih stabil, tetapi tetap harus dianalisis secara hati-hati.

Binding pocket
Rongga atau situs pada protein tempat ligan berikatan. Binding pocket menjadi pusat analisis dalam molecular docking.

Bioavailabilitas
Derajat dan kecepatan suatu senyawa mencapai sirkulasi sistemik setelah masuk ke tubuh. Dalam studi ADMET, bioavailabilitas sering diprediksi dari sifat fisikokimia molekul.

Bioinformatika struktural
Bidang yang mempelajari struktur biomolekul seperti protein, DNA, dan kompleks protein-ligan dengan pendekatan komputasi.

Biopython
Library Python untuk analisis data biologis, termasuk pembacaan struktur protein melalui modul Bio.PDB.

Boltzmann distribution
Distribusi probabilitas keadaan energi dalam sistem termal. Dalam kimia komputasi, digunakan untuk memahami populasi konformer dan energi relatif.

Breast cancer
Istilah bahasa Inggris untuk kanker payudara. Dalam buku ini digunakan sebagai konteks studi senyawa kecil, target molekuler, docking, descriptor, dan ADMET.

C

CADD
Singkatan dari *Computer-Aided Drug Design*. Pendekatan desain obat berbantuan komputer yang mencakup docking, QSAR, pharmacophore modeling, molecular dynamics, dan virtual screening.

cclib
Library Python untuk membaca dan mengekstrak data dari output program kimia kuantum seperti Gaussian, ORCA, GAMESS, NWChem, Psi4, dan lainnya.

Cheminformatics
Bidang yang menggabungkan kimia, informatika, statistik, dan komputasi untuk menyimpan, mengolah, menganalisis, dan memodelkan data molekul.

Conda
Manajer paket dan environment yang banyak digunakan untuk mengelola library Python ilmiah agar tidak terjadi konflik dependency.

Conformer
Bentuk tiga dimensi berbeda dari molekul yang muncul akibat rotasi ikatan tunggal. Dalam kimia komputasi, pencarian konformer penting sebelum optimasi dan docking.

Computational chemistry
Kimia komputasi, yaitu bidang kimia yang menggunakan metode matematika, fisika, dan komputasi untuk mempelajari struktur, energi, sifat, dan reaktivitas molekul.

Coordinate XYZ
Format koordinat molekul yang menyatakan posisi atom dalam sumbu x, y, dan z. Format XYZ sering digunakan sebagai input struktur pada perhitungan kimia komputasi.

Coulomb interaction
Interaksi elektrostatik antara partikel bermuatan. Dalam molekul, interaksi ini berpengaruh terhadap energi, distribusi muatan, dan stabilitas sistem.

Cytotoxicity
Kemampuan suatu senyawa untuk merusak atau membunuh sel. Dalam konteks anti-kanker, prediksi sitotoksisitas perlu diperlakukan hati-hati dan tidak boleh dianggap sebagai bukti terapi tanpa validasi biologis.

D

Data descriptor
Representasi numerik dari sifat molekul, seperti massa molekul, logP, jumlah donor/akseptor hidrogen, polar surface area, atau fingerprint.

DeepChem
Library Python untuk machine learning dan deep learning dalam kimia, biologi, dan desain obat.

Dependency
Paket atau library lain yang dibutuhkan agar suatu program dapat berjalan. Konflik dependency sering terjadi dalam environment kimia komputasi yang kompleks.

Descriptor molekul
Angka atau fitur yang menggambarkan sifat molekul. Descriptor digunakan dalam QSAR, machine learning, klasifikasi senyawa, dan analisis struktur-aktivitas.

DFT
Singkatan dari *Density Functional Theory*. Metode kimia kuantum yang menghitung sifat elektronik berdasarkan kerapatan elektron, bukan fungsi gelombang penuh.

DFTB
Singkatan dari *Density Functional Tight Binding*. Metode semiempiris berbasis DFT yang lebih cepat daripada DFT penuh dan cocok untuk sistem besar atau screening awal.

Dinamika molekul
Simulasi gerakan atom dan molekul terhadap waktu berdasarkan gaya antaratom. Digunakan untuk mempelajari kestabilan protein-ligan, pelarut, membran, dan sistem biomolekuler.

Docking
Metode komputasi untuk memprediksi posisi, orientasi, dan kekuatan interaksi ligan terhadap situs aktif protein.

Drug-likeness
Kesesuaian sifat molekul dengan karakteristik umum senyawa obat, misalnya berdasarkan aturan Lipinski, berat molekul, logP, donor/akseptor hidrogen, dan polar surface area.

E

Efek elektronik
Pengaruh distribusi elektron terhadap sifat molekul, seperti polaritas, reaktivitas, energi orbital, dan interaksi antar molekul.

Energi total
Nilai energi keseluruhan sistem molekul hasil perhitungan komputasi. Dalam DFT atau HF, energi total sering digunakan untuk membandingkan stabilitas relatif.

Energy minimization
Proses mencari struktur dengan energi paling rendah melalui perubahan koordinat atom. Proses ini disebut juga optimasi geometri.

Environment Python
Lingkungan kerja Python yang berisi versi Python dan library tertentu. Environment terpisah membantu menjaga stabilitas proyek komputasi.

Enzim target
Protein atau enzim yang menjadi sasaran interaksi senyawa kandidat. Dalam riset anti-kanker payudara, target dapat berupa aromatase, PARP, reseptor estrogen, atau protein lain yang relevan.

ER-positive breast cancer
Jenis kanker payudara yang pertumbuhannya dipengaruhi oleh reseptor estrogen. Senyawa inhibitor aromatase sering dikaji dalam konteks ini.

Exchange-correlation functional
Komponen penting dalam DFT yang merepresentasikan energi pertukaran dan korelasi elektron. Contoh functional: LDA, GGA, PBE, B3LYP, M06.

Experimental validation
Validasi melalui eksperimen laboratorium. Dalam riset in silico, validasi eksperimental diperlukan sebelum menyatakan klaim biologis yang kuat.

F

Farmakokinetik
Kajian tentang perjalanan senyawa dalam tubuh, meliputi absorpsi, distribusi, metabolisme, dan ekskresi.

Fingerprint molekul
Representasi biner atau numerik dari struktur molekul. Fingerprint digunakan untuk similarity search, clustering, QSAR, dan machine learning.

Force field
Model matematis yang menggambarkan energi potensial molekul berdasarkan parameter ikatan, sudut, torsi, dan interaksi non-ikatan.

Format file molekul
Format penyimpanan data struktur molekul, seperti XYZ, MOL, SDF, PDB, SMILES, InChI, dan CIF.

Frontier Molecular Orbital
Orbital molekul terluar yang paling berperan dalam reaktivitas, terutama HOMO dan LUMO.

Fungsi basis
Fungsi matematika penyusun basis set yang digunakan untuk mendeskripsikan orbital atom atau orbital molekul.

Functional
Dalam DFT, functional adalah pendekatan matematis untuk menghitung energi sistem dari kerapatan elektron.

Fukui function
Parameter reaktivitas lokal yang digunakan untuk memprediksi lokasi molekul yang rentan terhadap serangan nukleofilik, elektrofilik, atau radikal.

G

Gaussian basis function
Fungsi matematika berbentuk Gaussian yang banyak digunakan dalam basis set kimia kuantum.

Gaya antaratom
Gaya yang muncul akibat interaksi antaratom dalam sistem molekul. Gaya ini digunakan dalam optimasi geometri dan dinamika molekul.

Gaya van der Waals
Interaksi lemah antar molekul atau antar bagian molekul yang muncul akibat fluktuasi distribusi elektron.

Gibbs free energy
Energi bebas Gibbs, digunakan untuk menilai spontanitas proses pada suhu dan tekanan tertentu.

Geometry optimization
Proses mencari susunan atom paling stabil berdasarkan energi minimum. Dalam bahasa Indonesia disebut optimasi geometri.

GPAW
Program dan library untuk DFT berbasis metode PAW, grid real-space, dan plane wave, sering digunakan dalam studi material dan molekul.

Grid komputasi
Representasi ruang dalam bentuk titik-titik numerik untuk menghitung kerapatan elektron, potensial, atau permukaan molekul.

GUI kimia komputasi
Antarmuka grafis yang memudahkan pengguna menjalankan program kimia komputasi tanpa menulis seluruh perintah di terminal.

H

Hartree
Satuan energi atomik yang umum digunakan dalam kimia kuantum. Satu Hartree kira-kira setara dengan 627,5 kcal/mol.

Hartree–Fock
Metode ab initio dasar yang menghitung fungsi gelombang elektronik dengan pendekatan medan rata-rata.

Hibrida molekul
Senyawa rancangan yang menggabungkan dua atau lebih fragmen molekul aktif untuk memperoleh sifat baru atau meningkatkan potensi interaksi dengan target.

High-throughput screening
Penyaringan banyak senyawa secara cepat menggunakan alur komputasi otomatis.

HOMO
Singkatan dari *Highest Occupied Molecular Orbital*. HOMO adalah orbital molekul berenergi tertinggi yang masih terisi elektron.

HOMO-LUMO gap
Selisih energi antara HOMO dan LUMO. Gap ini sering digunakan untuk memperkirakan stabilitas elektronik dan reaktivitas molekul.

Hydrogen bond
Ikatan hidrogen, yaitu interaksi antara donor hidrogen dan akseptor elektronegatif seperti O, N, atau F. Ikatan ini penting dalam interaksi protein-ligan.

Hyperparameter
Parameter pengaturan model atau algoritma, misalnya dalam machine learning atau pencarian konformer, yang memengaruhi hasil akhir.

I

In silico
Pendekatan penelitian berbasis komputer. Dalam buku ini, riset senyawa anti-kanker payudara dilakukan secara in silico melalui DFT, docking, descriptor, dan ADMET.

In vitro
Penelitian yang dilakukan di luar organisme hidup, misalnya pada kultur sel atau tabung reaksi.

In vivo
Penelitian yang dilakukan pada organisme hidup.

InChI
Singkatan dari *International Chemical Identifier*. Format standar untuk merepresentasikan identitas kimia suatu molekul.

Input file
File masukan yang berisi struktur molekul, metode, basis set, muatan, spin, dan parameter lain untuk menjalankan perhitungan.

Interaksi elektrostatik
Interaksi antara muatan positif dan negatif dalam molekul atau kompleks protein-ligan.

Interaksi hidrofobik
Interaksi antara bagian molekul nonpolar dengan lingkungan nonpolar. Dalam docking, interaksi hidrofobik dapat memperkuat stabilitas kompleks.

Interpretasi hasil
Proses menjelaskan makna ilmiah dari data komputasi, termasuk energi, geometri, orbital, docking score, descriptor, dan prediksi ADMET.

J

Jadwal riset
Rencana waktu pelaksanaan penelitian, misalnya enam bulan untuk skripsi S1, yang mencakup studi literatur, preparasi data, perhitungan, analisis, dan penulisan.

Jari-jari atom
Ukuran perkiraan atom yang memengaruhi jarak ikatan, sterik, packing, dan interaksi molekul.

Jembatan hidrogen
Istilah lain untuk ikatan hidrogen, terutama ketika interaksi tersebut menghubungkan dua bagian molekul atau ligan dengan residu protein.

Jupyter Notebook
Lingkungan kerja interaktif untuk menulis kode Python, menjalankan analisis, menampilkan visualisasi, dan menyusun catatan eksperimen komputasi.

Justifikasi metode
Alasan ilmiah pemilihan metode komputasi, basis set, target protein, senyawa, parameter docking, atau model analisis.

Jalur reaksi
Rangkaian perubahan struktur dan energi dari reaktan menuju produk. Dalam komputasi, jalur reaksi dapat dipelajari melalui transition state dan profil energi.

K

Kanker payudara
Penyakit yang ditandai oleh pertumbuhan sel abnormal pada jaringan payudara. Dalam buku ini, kanker payudara menjadi konteks riset senyawa kecil dan kandidat inhibitor berbasis komputasi.

Kandidat senyawa
Molekul yang dipilih untuk dikaji lebih lanjut karena memiliki sifat komputasi yang menjanjikan, misalnya docking score baik, ADMET memadai, dan descriptor mendukung.

Kerapatan elektron
Distribusi elektron dalam ruang molekul. Kerapatan elektron menjadi konsep utama dalam DFT.

Kimia komputasi
Cabang kimia yang menggunakan model matematika dan komputer untuk menghitung struktur, energi, sifat, dan reaktivitas molekul.

Kimia kuantum
Cabang kimia yang menerapkan mekanika kuantum untuk menjelaskan struktur elektronik atom dan molekul.

Kinetika
Kajian tentang laju reaksi dan faktor yang memengaruhinya. Dalam komputasi, kinetika dapat dikaitkan dengan energi aktivasi dan jalur reaksi.

Kode program
Instruksi tertulis dalam bahasa pemrograman, misalnya Python, untuk menjalankan perhitungan, analisis, atau visualisasi data.

Konformer
Susunan ruang berbeda dari molekul yang muncul karena rotasi ikatan. Konformer dengan energi rendah sering dipilih untuk docking atau DFT.

L

Letrozole
Inhibitor aromatase yang digunakan dalam terapi kanker payudara hormon-positif. Dalam buku ini dapat menjadi contoh senyawa induk untuk desain hibrida.

Library Python
Paket atau pustaka Python yang menyediakan fungsi tertentu, misalnya PySCF untuk kimia kuantum dan RDKit untuk cheminformatics.

Ligan
Molekul kecil yang berikatan dengan protein atau reseptor. Dalam docking, ligan diposisikan pada binding pocket target.

Lipinski Rule of Five
Aturan empiris untuk menilai drug-likeness senyawa berdasarkan berat molekul, logP, donor hidrogen, dan akseptor hidrogen.

Log file
File catatan hasil running program, termasuk parameter, pesan kesalahan, energi, iterasi, dan output penting.

LogP
Koefisien partisi oktanol-air yang menggambarkan hidrofobisitas molekul. Nilai logP berpengaruh terhadap kelarutan, permeabilitas, dan drug-likeness.

LUMO
Singkatan dari *Lowest Unoccupied Molecular Orbital*. LUMO adalah orbital kosong berenergi terendah yang dapat menerima elektron.

M

Machine learning chemistry
Penerapan pembelajaran mesin untuk prediksi sifat molekul, QSAR, klasifikasi senyawa, desain obat, dan penemuan material.

Mekanika molekul
Metode komputasi yang memodelkan molekul menggunakan hukum mekanika klasik dan force field.

Mekanika kuantum
Teori fisika yang menjelaskan perilaku partikel pada skala atom dan subatom. Menjadi dasar kimia kuantum dan DFT.

Metode komputasi
Pendekatan perhitungan yang digunakan dalam riset, misalnya HF, DFT, DFTB, docking, molecular dynamics, QSAR, atau machine learning.

Molecular docking
Prediksi posisi dan orientasi ligan dalam binding pocket protein serta estimasi kekuatan interaksinya.

Molecular dynamics
Simulasi perubahan posisi atom terhadap waktu berdasarkan gaya antaratom.

Molecular orbital
Orbital yang menggambarkan distribusi elektron dalam molekul sebagai hasil kombinasi orbital atom.

Momen dipol
Besaran yang menunjukkan pemisahan muatan dalam molekul. Momen dipol berkaitan dengan polaritas dan interaksi antar molekul.

N

Nanomaterial
Material berukuran nanometer yang memiliki sifat unik. Dalam kimia komputasi, nanomaterial dapat dikaji melalui DFT, molecular dynamics, atau simulasi atomistik.

Natural bond orbital
Analisis orbital yang digunakan untuk memahami ikatan, donor-akseptor, dan distribusi elektron dalam molekul.

NCI analysis
Singkatan dari *Non-Covalent Interaction analysis*. Digunakan untuk memvisualisasi dan menganalisis interaksi non-kovalen dalam molekul atau kompleks.

Node orbital
Daerah pada orbital tempat peluang keberadaan elektron bernilai nol. Jumlah node berkaitan dengan energi orbital.

Normal mode
Pola getaran molekul yang digunakan dalam analisis frekuensi vibrasi.

Nukleofil
Spesies kaya elektron yang cenderung menyerang pusat bermuatan positif atau miskin elektron.

Numerical convergence
Kondisi ketika perhitungan iteratif mencapai kestabilan numerik sesuai batas toleransi yang ditentukan.

O

Olaparib
Inhibitor PARP yang digunakan dalam terapi kanker tertentu, termasuk kanker payudara dengan mutasi BRCA. Dalam buku ini dapat menjadi model senyawa untuk desain hibrida.

Open Babel
Perangkat lunak dan library untuk konversi format molekul, manipulasi struktur, protonasi, dan preparasi file kimia.

OpenMM
Library untuk molecular dynamics, terutama simulasi biomolekul dengan dukungan GPU.

Optimasi geometri
Proses menemukan struktur molekul paling stabil dengan meminimalkan energi sistem.

Orbital molekul
Fungsi matematika yang menggambarkan distribusi elektron dalam molekul.

Output file
File hasil perhitungan yang berisi energi, geometri, orbital, muatan, frekuensi, atau data lain.

Overfitting
Kondisi dalam machine learning ketika model terlalu menyesuaikan data pelatihan sehingga buruk dalam memprediksi data baru.

P

PARP
Singkatan dari *Poly(ADP-ribose) polymerase*. Enzim yang terlibat dalam perbaikan DNA dan menjadi target penting dalam terapi kanker tertentu.

PDB
Singkatan dari *Protein Data Bank*. Basis data struktur tiga dimensi protein, DNA, RNA, dan kompleks biomolekul.

PDBFixer
Tool untuk memperbaiki file PDB, misalnya menambahkan atom hilang, residu hilang, hidrogen, pelarut, atau ion.

Pharmacophore
Susunan fitur molekul yang diperlukan untuk aktivitas biologis, seperti donor hidrogen, akseptor hidrogen, pusat hidrofobik, dan muatan.

Pipeline riset
Alur kerja terstruktur yang menghubungkan semua tahap penelitian, dari input data hingga kesimpulan.

Protein-ligan
Kompleks yang terbentuk antara protein target dan ligan. Analisis kompleks protein-ligan penting dalam docking dan desain obat.

PyMOL
Perangkat lunak visualisasi struktur molekul dan protein.

PySCF
Library Python untuk perhitungan kimia kuantum seperti HF, DFT, MP2, CASSCF, TDDFT, dan sistem molekul maupun periodik.

Python
Bahasa pemrograman yang banyak digunakan dalam sains komputasi, data science, machine learning, kimia komputasi, dan otomasi riset.

Q

QCElemental
Library Python yang menyediakan struktur data, konstanta fisika, satuan, tabel periodik, dan representasi molekul untuk kimia kuantum.

QCEngine
Library untuk menjalankan berbagai program kimia kuantum melalui standar input-output yang konsisten.

QCSchema
Standar format data untuk menyimpan informasi molekul, metode, basis set, dan hasil perhitungan kimia kuantum.

QSAR
Singkatan dari *Quantitative Structure–Activity Relationship*. Pendekatan yang menghubungkan struktur molekul dengan aktivitas biologis menggunakan descriptor dan model statistik atau machine learning.

QSPR
Singkatan dari *Quantitative Structure–Property Relationship*. Pendekatan yang menghubungkan struktur molekul dengan sifat fisikokimia tertentu.

Quantum chemistry
Kimia kuantum, yaitu penerapan mekanika kuantum untuk mempelajari struktur dan sifat molekul.

Quantum descriptor
Descriptor yang berasal dari perhitungan kimia kuantum, misalnya energi HOMO, energi LUMO, gap HOMO-LUMO, momen dipol, dan muatan atom.

R

Rancangan proposal
Dokumen akademik yang menjelaskan latar belakang, rumusan masalah, tujuan, manfaat, metode, data, jadwal, dan luaran penelitian.

RDKit
Library cheminformatics Python untuk membaca, membuat, memanipulasi, menganalisis, dan memvisualisasi molekul, termasuk SMILES, fingerprint, descriptor, dan konformer.

Receptor
Protein atau makromolekul target yang menerima ligan dalam docking.

Reproducibility
Kemampuan suatu penelitian untuk diulang dan menghasilkan hasil yang konsisten jika menggunakan input, metode, parameter, dan environment yang sama.

Residue
Unit asam amino dalam protein. Dalam docking, residu pada binding pocket dianalisis untuk memahami interaksi dengan ligan.

Riset S1
Penelitian tingkat sarjana yang harus realistis dari sisi waktu, perangkat, metode, kemampuan mahasiswa, dan ketersediaan data.

Root Mean Square Deviation
Ukuran deviasi posisi atom antara dua struktur. RMSD sering digunakan dalam validasi docking atau analisis dinamika molekul.

Running program
Proses menjalankan kode atau software komputasi untuk menghasilkan output riset.

S

Sampling konformer
Proses mencari berbagai kemungkinan bentuk tiga dimensi molekul untuk menemukan struktur yang paling stabil atau relevan.

SCF
Singkatan dari *Self-Consistent Field*. Prosedur iteratif dalam HF dan DFT untuk mencapai solusi elektronik yang stabil.

Scoring function
Fungsi matematis dalam docking untuk memperkirakan kualitas ikatan antara ligan dan reseptor.

Semiempiris
Metode komputasi yang menggabungkan teori dengan parameter empiris agar perhitungan lebih cepat.

Senyawa hibrida
Molekul rancangan yang menggabungkan dua fragmen aktif, misalnya fragmen letrozole-anastrozole atau olaparib-talazoparib.

SMILES
Singkatan dari *Simplified Molecular Input Line Entry System*. Representasi struktur molekul dalam bentuk string teks.

Spin
Sifat kuantum elektron yang berpengaruh pada keadaan elektronik molekul, terutama sistem radikal, ion, atau logam transisi.

Struktur molekul
Susunan atom dan ikatan dalam suatu molekul, baik dalam representasi 2D maupun 3D.

T

Talazoparib
Inhibitor PARP yang digunakan sebagai contoh senyawa dalam riset anti-kanker. Dapat dikaji bersama olaparib dalam desain senyawa hibrida.

Target molekuler
Protein, enzim, atau reseptor yang dipilih sebagai sasaran interaksi ligan dalam riset desain obat.

TDDFT
Singkatan dari *Time-Dependent Density Functional Theory*. Metode DFT untuk mempelajari keadaan tereksitasi dan spektrum elektronik.

Template kode
Struktur awal program yang dapat dimodifikasi untuk molekul atau kasus riset lain.

Terminal
Antarmuka berbasis teks untuk menjalankan perintah, script Python, instalasi library, dan program komputasi.

Toksisitas
Potensi suatu senyawa menimbulkan efek merugikan pada organisme atau sel. Prediksi toksisitas dalam ADMET bersifat awal dan perlu validasi.

Trajectory
Rekaman posisi atom terhadap waktu dalam simulasi molecular dynamics.

Transition state
Keadaan energi tinggi yang menghubungkan reaktan dan produk dalam jalur reaksi.

U

Umbrella sampling
Metode enhanced sampling dalam molecular dynamics untuk menghitung profil energi bebas sepanjang koordinat reaksi tertentu.

Unit cell
Sel satuan terkecil dalam struktur kristal yang dapat merepresentasikan keseluruhan kristal melalui translasi periodik.

Unoccupied orbital
Orbital yang belum terisi elektron. LUMO adalah contoh orbital kosong yang penting dalam analisis reaktivitas.

Update library
Proses memperbarui paket Python atau software. Dalam riset reproducible, update perlu dicatat karena dapat memengaruhi hasil.

Uji validasi
Prosedur untuk memastikan metode, data, atau hasil dapat dipercaya. Contohnya redocking dalam docking atau perbandingan hasil dengan literatur.

Uji virtual
Pengujian komputasi terhadap senyawa, misalnya docking, ADMET, QSAR, atau prediksi descriptor, sebelum eksperimen laboratorium.

V

Validasi docking
Proses memastikan protokol docking cukup layak, misalnya melalui redocking ligan asli dan perhitungan RMSD.

Validasi metode
Pengujian kelayakan metode komputasi melalui pembandingan dengan literatur, data eksperimen, atau metode lain.

van der Waals interaction
Interaksi non-kovalen lemah yang penting dalam stabilitas molekul dan kompleks protein-ligan.

Variabel penelitian
Aspek yang diamati atau dibandingkan dalam riset, misalnya jenis senyawa, metode DFT, basis set, docking score, descriptor, atau ADMET.

Virtual screening
Penyaringan komputasi terhadap banyak senyawa untuk menemukan kandidat yang paling menjanjikan.

Visualisasi molekul
Penyajian struktur molekul dalam bentuk gambar 2D atau 3D agar lebih mudah dipahami dan dianalisis.

VMD
Software visualisasi molekul yang banyak digunakan untuk protein, ligan, dan trajectory molecular dynamics.

W

Wavefunction
Fungsi gelombang yang menggambarkan keadaan kuantum sistem elektron. Digunakan dalam metode berbasis fungsi gelombang seperti Hartree–Fock.

Workflow
Alur kerja sistematis dalam riset komputasi, mulai dari input, proses, analisis, hingga output.

Workflow automation
Otomasi alur kerja komputasi menggunakan script agar proses perhitungan banyak molekul dapat dilakukan secara efisien.

Working directory
Folder kerja tempat file input, script, output, log, dan hasil analisis disimpan.

Write-up ilmiah
Penulisan hasil riset dalam format akademik, seperti proposal, laporan praktikum, skripsi, artikel, atau makalah seminar.

X

XYZ file
Format file sederhana untuk menyimpan koordinat atom tiga dimensi. Biasanya memuat jumlah atom, baris komentar, dan daftar atom beserta koordinat x, y, z.

xTB
Program semiempiris cepat berbasis metode GFN-xTB untuk optimasi geometri, energi relatif, frekuensi, dan screening molekul organik.

X-axis
Sumbu horizontal pada grafik. Dalam analisis data, X-axis dapat mewakili senyawa, waktu, descriptor, atau variabel bebas.

X-ray crystallography
Metode eksperimen untuk menentukan struktur kristal molekul atau protein. Struktur hasil kristalografi sering digunakan sebagai input docking.

XML force field
Format penyimpanan parameter force field yang dapat digunakan dalam simulasi molecular dynamics, misalnya pada OpenMM.

Y

Yield komputasi
Istilah tidak baku yang dapat digunakan secara terbatas untuk menggambarkan keluaran atau hasil yang diperoleh dari suatu alur komputasi.

Y-axis
Sumbu vertikal pada grafik. Dalam analisis docking, Y-axis dapat menampilkan binding energy, skor docking, atau nilai descriptor.

YAML
Format file teks terstruktur yang sering digunakan untuk konfigurasi environment, parameter workflow, atau metadata riset.

Ylida / ylide
Spesies kimia dengan muatan positif dan negatif pada atom berdekatan. Istilah ini dapat muncul dalam kajian reaktivitas organik, meskipun bukan fokus utama buku.

Z

Z-matrix
Format representasi struktur molekul berdasarkan jarak ikatan, sudut ikatan, dan sudut dihedral. Z-matrix sering digunakan dalam kimia kuantum.

Zero-point energy
Energi vibrasi minimum yang tetap dimiliki molekul pada suhu nol mutlak menurut mekanika kuantum.

Zinc database
Basis data senyawa yang sering digunakan dalam virtual screening dan desain obat komputasi.

Zona aktif
Bagian protein atau enzim yang berperan dalam pengikatan ligan atau reaksi katalitik. Dalam docking, zona ini sering menjadi pusat grid docking.

Zwitterion
Molekul yang memiliki muatan positif dan negatif sekaligus tetapi total muatannya netral. Bentuk zwitterion penting dalam studi asam amino, obat, dan pH biologis.

LAMPIRAN E

TEMPLATE LAPORAN HASIL KOMPUTASI

E.1 Identitas Perhitungan

Bagian ini memuat nama molekul, kode kandidat, rumus molekul, sumber struktur, SMILES, format file input, metode komputasi, basis set, muatan, spin, environment, versi Python, dan library utama. Identitas perhitungan harus ditulis lengkap karena menjadi dasar reproducibility.

Format isian:

.....
.....

Catatan pemeriksa:

.....
.....

E.2 Tujuan Perhitungan

Tuliskan tujuan secara spesifik. Contoh: menghitung energi elektronik dan parameter HOMO-LUMO kandidat hibrida LAH-01 menggunakan B3LYP-D3BJ/def2-SVP untuk mendukung seleksi awal kandidat anti-kanker payudara berbasis target aromatase.

Format isian:

.....
.....

Catatan pemeriksa:

.....
.....

E.3 Prosedur Singkat

Jelaskan urutan kerja mulai dari persiapan struktur, optimasi awal, perhitungan DFT, ekstraksi data, visualisasi, hingga penyimpanan output. Prosedur tidak perlu terlalu panjang, tetapi harus cukup jelas untuk diulang.

Format isian:

.....
.....

Catatan pemeriksa:

.....
.....

E.4 Tabel Hasil Utama

Tabel utama dapat berisi energi total, HOMO, LUMO, gap, momen dipol, status konvergensi, dan catatan struktur. Gunakan satuan yang jelas. Jika ada konversi satuan, sebutkan faktor konversi atau library yang digunakan.

Format isian:

.....
.....

Catatan pemeriksa:

.....

.....

E.5 Interpretasi

Interpretasi harus menghubungkan angka dengan makna kimia. Jangan hanya menulis “nilai gap sebesar sekian”. Jelaskan apakah gap tersebut lebih besar atau lebih kecil dari kandidat lain, apa kemungkinan maknanya, dan apa batasannya.

Format isian:

.....

.....

Catatan pemeriksa:

.....

.....

E.6 Kesimpulan Sementara

Kesimpulan sementara menjawab tujuan perhitungan, bukan tujuan seluruh penelitian. Tuliskan apakah kandidat layak dilanjutkan ke tahap docking, ADMET, atau optimasi lebih lanjut.

Format isian:

.....

.....

Catatan pemeriksa:

.....

.....

E.7 Lampiran File

Daftar file yang perlu dilampirkan meliputi file input, file output, struktur awal, struktur akhir, log running, tabel hasil, grafik, dan skrip analisis. Penamaan file harus konsisten dengan logbook.

Format isian:

.....

.....

Catatan pemeriksa:

.....

.....

LAMPIRAN F

PANDUAN INTERPRETASI PARAMETER KOMPUTASI

Energi total
Energi total hasil DFT menunjukkan energi sistem pada metode dan basis set tertentu. Nilai ini sangat berguna untuk membandingkan struktur atau konformer yang sejenis, tetapi tidak boleh digunakan secara sederhana untuk membandingkan molekul yang berbeda komposisi atomnya tanpa skema energi relatif yang benar.

Pertanyaan refleksi: apakah parameter ini konsisten dengan parameter lain, atau justru menunjukkan trade-off yang perlu dibahas?

HOMO
HOMO menggambarkan orbital molekul terisi tertinggi. Dalam interpretasi sederhana, energi HOMO dapat dikaitkan dengan kemampuan donor elektron. Namun, hubungan ini harus dipakai sebagai indikator awal, bukan kesimpulan tunggal.

Pertanyaan refleksi: apakah parameter ini konsisten dengan parameter lain, atau justru menunjukkan trade-off yang perlu dibahas?

LUMO
LUMO menggambarkan orbital kosong terendah. Energi LUMO sering dikaitkan dengan kemampuan akseptor elektron. Dalam desain molekul, perubahan substituen dapat memengaruhi tingkat energi ini.

Pertanyaan refleksi: apakah parameter ini konsisten dengan parameter lain, atau justru menunjukkan trade-off yang perlu dibahas?

HOMO-LUMO gap
Gap HOMO-LUMO sering dipakai sebagai indikator kestabilan elektronik dan reaktivitas. Gap kecil dapat mengindikasikan reaktivitas lebih tinggi, sedangkan gap besar dapat menunjukkan kestabilan relatif. Interpretasi harus dibandingkan dalam kelompok molekul yang relevan.

Pertanyaan refleksi: apakah parameter ini konsisten dengan parameter lain, atau justru menunjukkan trade-off yang perlu dibahas?

Momen dipol
Momen dipol menggambarkan distribusi muatan dan polaritas molekul. Nilai ini dapat memengaruhi interaksi dengan lingkungan polar, sisi aktif protein, atau pelarut. Namun, momen dipol gas phase dapat berbeda dari kondisi biologis.

Pertanyaan refleksi: apakah parameter ini konsisten dengan parameter lain, atau justru menunjukkan trade-off yang perlu dibahas?

Docking score

Docking score adalah nilai prediksi dari fungsi penilaian docking. Nilai lebih rendah sering dianggap lebih baik pada beberapa software, tetapi skor harus dibaca bersama pose, residu interaksi, validasi, dan batasan scoring function.

Pertanyaan refleksi: apakah parameter ini konsisten dengan parameter lain, atau justru menunjukkan trade-off yang perlu dibahas?

Ikatan hidrogen

Ikatan hidrogen dapat meningkatkan stabilitas pose ligan jika geometri donor-akseptor masuk akal. Jumlah ikatan hidrogen bukan satu-satunya ukuran; posisi dan residu yang terlibat lebih penting.

Pertanyaan refleksi: apakah parameter ini konsisten dengan parameter lain, atau justru menunjukkan trade-off yang perlu dibahas?

Interaksi hidrofobik

Interaksi hidrofobik sering dominan pada kantong protein nonpolar. Interaksi ini dapat mendukung afinitas, tetapi lipofilisitas berlebihan dapat mengganggu kelayakan ADMET.

Pertanyaan refleksi: apakah parameter ini konsisten dengan parameter lain, atau justru menunjukkan trade-off yang perlu dibahas?

LogP

LogP menggambarkan kecenderungan lipofilisitas. Nilai terlalu tinggi dapat mengindikasikan kelarutan rendah, sedangkan nilai terlalu rendah dapat mengganggu permeabilitas membran. Interpretasi harus disesuaikan dengan konteks kandidat.

Pertanyaan refleksi: apakah parameter ini konsisten dengan parameter lain, atau justru menunjukkan trade-off yang perlu dibahas?

TPSA

Topological polar surface area berkaitan dengan polaritas permukaan molekul. TPSA dapat membantu menilai permeabilitas dan potensi absorpsi, tetapi bukan satu-satunya parameter farmakokinetik.

Pertanyaan refleksi: apakah parameter ini konsisten dengan parameter lain, atau justru menunjukkan trade-off yang perlu dibahas?

HBD dan HBA

Hydrogen bond donor dan acceptor adalah parameter penting dalam aturan drug-likeness. Jumlah yang berlebihan dapat memengaruhi permeabilitas dan kelarutan.

Pertanyaan refleksi: apakah parameter ini konsisten dengan parameter lain, atau justru menunjukkan trade-off yang perlu dibahas?

Rotatable bond

Jumlah ikatan rotatable berkaitan dengan fleksibilitas molekul. Fleksibilitas tinggi dapat memengaruhi entropi ikatan dan pose docking. Molekul yang terlalu fleksibel sering memerlukan analisis konformer lebih cermat.

Pertanyaan refleksi: apakah parameter ini konsisten dengan parameter lain, atau justru menunjukkan trade-off yang perlu dibahas?

LAMPIRAN G
CONTOH MATRIKS KEPUTUSAN KANDIDAT

Kriteria 1: Penilaian kandidat berdasarkan indikator komputasi ke-1.

Deskripsi: isi dengan parameter yang relevan, misalnya energi relatif, HOMO-LUMO gap, docking score, residu kunci, pelanggaran Lipinski, prediksi toksisitas, atau kemudahan sintesis hipotetik.

Skor 1-5: 1 sangat lemah, 2 lemah, 3 cukup, 4 baik, 5 sangat baik. Gunakan catatan naratif agar penilaian tidak menjadi angka kosong.

Catatan keputusan: tuliskan alasan mengapa kandidat dipertahankan, dieliminasi, atau direkomendasikan untuk validasi lebih lanjut.

Kriteria 2: Penilaian kandidat berdasarkan indikator komputasi ke-2.

Deskripsi: isi dengan parameter yang relevan, misalnya energi relatif, HOMO-LUMO gap, docking score, residu kunci, pelanggaran Lipinski, prediksi toksisitas, atau kemudahan sintesis hipotetik.

Skor 1-5: 1 sangat lemah, 2 lemah, 3 cukup, 4 baik, 5 sangat baik. Gunakan catatan naratif agar penilaian tidak menjadi angka kosong.

Catatan keputusan: tuliskan alasan mengapa kandidat dipertahankan, dieliminasi, atau direkomendasikan untuk validasi lebih lanjut.

Kriteria 3: Penilaian kandidat berdasarkan indikator komputasi ke-3.

Deskripsi: isi dengan parameter yang relevan, misalnya energi relatif, HOMO-LUMO gap, docking score, residu kunci, pelanggaran Lipinski, prediksi toksisitas, atau kemudahan sintesis hipotetik.

Skor 1-5: 1 sangat lemah, 2 lemah, 3 cukup, 4 baik, 5 sangat baik. Gunakan catatan naratif agar penilaian tidak menjadi angka kosong.

Catatan keputusan: tuliskan alasan mengapa kandidat dipertahankan, dieliminasi, atau direkomendasikan untuk validasi lebih lanjut.

Kriteria 4: Penilaian kandidat berdasarkan indikator komputasi ke-4.

Deskripsi: isi dengan parameter yang relevan, misalnya energi relatif, HOMO-LUMO gap, docking score, residu kunci, pelanggaran Lipinski, prediksi toksisitas, atau kemudahan sintesis hipotetik.

Skor 1-5: 1 sangat lemah, 2 lemah, 3 cukup, 4 baik, 5 sangat baik. Gunakan catatan naratif agar penilaian tidak menjadi angka kosong.

Catatan keputusan: tuliskan alasan mengapa kandidat dipertahankan, dieliminasi, atau direkomendasikan untuk validasi lebih lanjut.

Kriteria 5: Penilaian kandidat berdasarkan indikator komputasi ke-5.

Deskripsi: isi dengan parameter yang relevan, misalnya energi relatif, HOMO-LUMO gap, docking score, residu kunci, pelanggaran Lipinski, prediksi toksisitas, atau kemudahan sintesis hipotetik.

Skor 1-5: 1 sangat lemah, 2 lemah, 3 cukup, 4 baik, 5 sangat baik. Gunakan catatan naratif agar penilaian tidak menjadi angka kosong.

Catatan keputusan: tuliskan alasan mengapa kandidat dipertahankan, dieliminasi, atau direkomendasikan untuk validasi lebih lanjut.

Kriteria 6: Penilaian kandidat berdasarkan indikator komputasi ke-6.

Deskripsi: isi dengan parameter yang relevan, misalnya energi relatif, HOMO-LUMO gap, docking score, residu kunci, pelanggaran Lipinski, prediksi toksisitas, atau kemudahan sintesis hipotetik.

Skor 1-5: 1 sangat lemah, 2 lemah, 3 cukup, 4 baik, 5 sangat baik. Gunakan catatan naratif agar penilaian tidak menjadi angka kosong.

Catatan keputusan: tuliskan alasan mengapa kandidat dipertahankan, dieliminasi, atau direkomendasikan untuk validasi lebih lanjut.

Kriteria 7: Penilaian kandidat berdasarkan indikator komputasi ke-7.

Deskripsi: isi dengan parameter yang relevan, misalnya energi relatif, HOMO-LUMO gap, docking score, residu kunci, pelanggaran Lipinski, prediksi toksisitas, atau kemudahan sintesis hipotetik.

Skor 1-5: 1 sangat lemah, 2 lemah, 3 cukup, 4 baik, 5 sangat baik. Gunakan catatan naratif agar penilaian tidak menjadi angka kosong.

Catatan keputusan: tuliskan alasan mengapa kandidat dipertahankan, dieliminasi, atau direkomendasikan untuk validasi lebih lanjut.

Kriteria 8: Penilaian kandidat berdasarkan indikator komputasi ke-8.

Deskripsi: isi dengan parameter yang relevan, misalnya energi relatif, HOMO-LUMO gap, docking score, residu kunci, pelanggaran Lipinski, prediksi toksisitas, atau kemudahan sintesis hipotetik.

Skor 1-5: 1 sangat lemah, 2 lemah, 3 cukup, 4 baik, 5 sangat baik. Gunakan catatan naratif agar penilaian tidak menjadi angka kosong.

Catatan keputusan: tuliskan alasan mengapa kandidat dipertahankan, dieliminasi, atau direkomendasikan untuk validasi lebih lanjut.

Kriteria 9: Penilaian kandidat berdasarkan indikator komputasi ke-9.

Deskripsi: isi dengan parameter yang relevan, misalnya energi relatif, HOMO-LUMO gap, docking score, residu kunci, pelanggaran Lipinski, prediksi toksisitas, atau kemudahan sintesis hipotetik.

Skor 1-5: 1 sangat lemah, 2 lemah, 3 cukup, 4 baik, 5 sangat baik. Gunakan catatan naratif agar penilaian tidak menjadi angka kosong.

Catatan keputusan: tuliskan alasan mengapa kandidat dipertahankan, dieliminasi, atau direkomendasikan untuk validasi lebih lanjut.

Kriteria 10: Penilaian kandidat berdasarkan indikator komputasi ke-10.

Deskripsi: isi dengan parameter yang relevan, misalnya energi relatif, HOMO-LUMO gap, docking score, residu kunci, pelanggaran Lipinski, prediksi toksisitas, atau kemudahan sintesis hipotetik.

Skor 1-5: 1 sangat lemah, 2 lemah, 3 cukup, 4 baik, 5 sangat baik. Gunakan catatan naratif agar penilaian tidak menjadi angka kosong.

Catatan keputusan: tuliskan alasan mengapa kandidat dipertahankan, dieliminasi, atau direkomendasikan untuk validasi lebih lanjut.

Kriteria 11: Penilaian kandidat berdasarkan indikator komputasi ke-11.

Deskripsi: isi dengan parameter yang relevan, misalnya energi relatif, HOMO-LUMO gap, docking score, residu kunci, pelanggaran Lipinski, prediksi toksisitas, atau kemudahan sintesis hipotetik.

Skor 1-5: 1 sangat lemah, 2 lemah, 3 cukup, 4 baik, 5 sangat baik. Gunakan catatan naratif agar penilaian tidak menjadi angka kosong.

Catatan keputusan: tuliskan alasan mengapa kandidat dipertahankan, dieliminasi, atau direkomendasikan untuk validasi lebih lanjut.

Kriteria 12: Penilaian kandidat berdasarkan indikator komputasi ke-12.

Deskripsi: isi dengan parameter yang relevan, misalnya energi relatif, HOMO-LUMO gap, docking score, residu kunci, pelanggaran Lipinski, prediksi toksisitas, atau kemudahan sintesis hipotetik.

Skor 1-5: 1 sangat lemah, 2 lemah, 3 cukup, 4 baik, 5 sangat baik. Gunakan catatan naratif agar penilaian tidak menjadi angka kosong.

Catatan keputusan: tuliskan alasan mengapa kandidat dipertahankan, dieliminasi, atau direkomendasikan untuk validasi lebih lanjut.

Kriteria 13: Penilaian kandidat berdasarkan indikator komputasi ke-13.

Deskripsi: isi dengan parameter yang relevan, misalnya energi relatif, HOMO-LUMO gap, docking score, residu kunci, pelanggaran Lipinski, prediksi toksisitas, atau kemudahan sintesis hipotetik.

Skor 1-5: 1 sangat lemah, 2 lemah, 3 cukup, 4 baik, 5 sangat baik. Gunakan catatan naratif agar penilaian tidak menjadi angka kosong.

Catatan keputusan: tuliskan alasan mengapa kandidat dipertahankan, dieliminasi, atau direkomendasikan untuk validasi lebih lanjut.

Kriteria 14: Penilaian kandidat berdasarkan indikator komputasi ke-14.

Deskripsi: isi dengan parameter yang relevan, misalnya energi relatif, HOMO-LUMO gap, docking score, residu kunci, pelanggaran Lipinski, prediksi toksisitas, atau kemudahan sintesis hipotetik.

Skor 1-5: 1 sangat lemah, 2 lemah, 3 cukup, 4 baik, 5 sangat baik. Gunakan catatan naratif agar penilaian tidak menjadi angka kosong.

Catatan keputusan: tuliskan alasan mengapa kandidat dipertahankan, dieliminasi, atau direkomendasikan untuk validasi lebih lanjut.

Kriteria 15: Penilaian kandidat berdasarkan indikator komputasi ke-15.

Deskripsi: isi dengan parameter yang relevan, misalnya energi relatif, HOMO-LUMO gap, docking score, residu kunci, pelanggaran Lipinski, prediksi toksisitas, atau kemudahan sintesis hipotetik.

Skor 1-5: 1 sangat lemah, 2 lemah, 3 cukup, 4 baik, 5 sangat baik. Gunakan catatan naratif agar penilaian tidak menjadi angka kosong.

Catatan keputusan: tuliskan alasan mengapa kandidat dipertahankan, dieliminasi, atau direkomendasikan untuk validasi lebih lanjut.

CATATAN PENUTUP AKHIR

Pembaca disarankan menggunakan buku ini secara aktif: menjalankan kode, mencatat hasil, membandingkan metode, dan menulis refleksi setelah setiap tahap riset.

Kimia komputasi adalah bidang yang menuntut ketelitian. Kesalahan kecil pada struktur input, muatan, basis set, atau parameter docking dapat mengubah hasil. Karena itu, disiplin dokumentasi, validasi, dan interpretasi menjadi bagian utama dari kompetensi peneliti.

Pengembangan selanjutnya dapat diarahkan pada integrasi workflow otomatis, penggunaan machine learning untuk QSAR, simulasi dinamika molekul yang lebih panjang, perhitungan energi bebas, dan validasi eksperimental terhadap kandidat yang paling menjanjikan.

DAFTAR PUSTAKA

A. Buku Dasar Kimia Komputasi, Kimia Kuantum, dan Pemodelan Molekul

Atkins, P. W., & Friedman, R. S. (2011). *Molecular quantum mechanics* (5th ed.). Oxford University Press.

Cramer, C. J. (2013). *Essentials of computational chemistry: Theories and models* (2nd ed.). John Wiley & Sons.

Foresman, J. B., & Frisch, Æ. (2015). *Exploring chemistry with electronic structure methods* (3rd ed.). Gaussian, Inc.

Jensen, F. (2017). *Introduction to computational chemistry* (3rd ed.). John Wiley & Sons.

Koch, W., & Holthausen, M. C. (2015). *A chemist's guide to density functional theory* (2nd ed.). Wiley-VCH.

Leach, A. R. (2001). *Molecular modelling: Principles and applications* (2nd ed.). Pearson Education.

Levine, I. N. (2014). *Quantum chemistry* (7th ed.). Pearson.

Lewars, E. G. (2016). *Computational chemistry: Introduction to the theory and applications of molecular and quantum mechanics* (3rd ed.). Springer.

McQuarrie, D. A. (2007). *Quantum chemistry* (2nd ed.). University Science Books.

Szabo, A., & Ostlund, N. S. (1996). *Modern quantum chemistry: Introduction to advanced electronic structure theory*. Dover Publications.

Young, D. C. (2001). *Computational chemistry: A practical guide for applying techniques to real-world problems*. John Wiley & Sons.

B. Teori DFT, Basis Set, dan Metode Struktur Elektronik

Becke, A. D. (1988). Density-functional exchange-energy approximation with correct asymptotic behavior. *Physical Review A, 38*(6), 3098–3100. doi: 10.1103/PhysRevA.38.3098.

Becke, A. D. (1993). Density-functional thermochemistry. III. The role of exact exchange. *The Journal of Chemical Physics, 98*(7), 5648–5652. doi: 10.1063/1.464913.

Ditchfield, R., Hehre, W. J., & Pople, J. A. (1971). Self-consistent molecular-orbital methods. IX. An extended Gaussian-type basis for molecular-orbital studies of organic molecules. *The Journal of Chemical Physics, 54*(2), 724–728. doi: 10.1063/1.1674902.

Grimme, S. (2006). Semiempirical GGA-type density functional constructed with a long-range dispersion correction. *Journal of Computational Chemistry, 27*(15), 1787–1799. doi: 10.1002/jcc.20495.

Hohenberg, P., & Kohn, W. (1964). Inhomogeneous electron gas. *Physical Review, 136*(3B), B864–B871. doi: 10.1103/PhysRev.136.B864.

Kohn, W., & Sham, L. J. (1965). Self-consistent equations including exchange and correlation effects. *Physical Review, 140*(4A), A1133–A1138. doi: 10.1103/PhysRev.140.A1133.

Lee, C., Yang, W., & Parr, R. G. (1988). Development of the Colle–Salvetti correlation-energy formula into a functional of the electron density. *Physical Review B, 37*(2), 785–789. doi: 10.1103/PhysRevB.37.785.

Perdew, J. P., Burke, K., & Ernzerhof, M. (1996). Generalized gradient approximation made simple. *Physical Review Letters, 77*(18), 3865–3868. doi: 10.1103/PhysRevLett.77.3865.

Weigend, F., & Ahlrichs, R. (2005). Balanced basis sets of split valence, triple zeta valence, and quadruple zeta valence quality for H to Rn. *Physical Chemistry Chemical Physics, 7*(18), 3297–3305. doi: 10.1039/B508541A.

C. Python, Scientific Computing, dan Library Dasar

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature, 585*, 357–362. doi: 10.1038/s41586-020-2649-2.

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering, 9*(3), 90–95. doi: 10.1109/MCSE.2007.55.

McKinney, W. (2010). Data structures for statistical computing in Python. In S. van der Walt & J. Millman (Eds.), *Proceedings of the 9th Python in Science Conference* (pp. 56–61).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12*, 2825–2830. Scikit-learn juga masih dipelihara sebagai library machine learning Python aktif. (Scikit-learn)

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods, 17*, 261–272. doi: 10.1038/s41592-019-0686-2.

D. Software Kimia Kuantum dan Simulasi Atomistik Berbasis Python

Enkovaara, J., Rostgaard, C., Mortensen, J. J., Chen, J., Dulak, M., Ferrighi, L., Gavnholt, J., Glinsvad, C., Haikola, V., Hansen, H. A., Kristoffersen, H. H., Kuisma, M., Larsen, A. H., Lehtovaara, L., Ljungberg, M., Lopez-Acevedo, O., Moses, P. G., Ojanen, J., Olsen, T., … Nieminen, R. M. (2010). Electronic structure calculations with GPAW: A real-space implementation of the projector augmented-wave method. *Journal of Physics: Condensed Matter, 22*(25), 253202. doi: 10.1088/0953-8984/22/25/253202.

GPAW Developers. (2026). *GPAW: DFT and beyond within the projector-augmented wave method*. GPAW Documentation. GPAW merupakan kode DFT Python berbasis PAW yang terintegrasi dengan ASE. (gpaw.readthedocs.io)

Larsen, A. H., Mortensen, J. J., Blomqvist, J., Castelli, I. E., Christensen, R., Dułak, M., Friis, J., Groves, M. N., Hammer, B., Hargus, C., Hermes, E. D., Jennings, P. C., Jensen, P. B., Kermode, J., Kitchin, J. R., Kolsbjerg, E. L., Kubal, J., Kaasbjerg, K., Lysgaard, S., … Jacobsen, K. W. (2017). The Atomic Simulation Environment—A Python library for working with atoms. *Journal of Physics: Condensed Matter, 29*(27), 273002. doi: 10.1088/1361-648X/aa680e.

Mortensen, J. J., Larsen, A. H., Kuisma, M., Ivanov, A. V., Taghizadeh, A., Peterson, A., Haldar, A., Dohn, A. O., Schäfer, C., Jónsson, E. Ö., … Thygesen, K. S. (2024). GPAW: An open Python package for electronic-structure calculations. *The Journal of Chemical Physics, 160*, 092503. doi: 10.1063/5.0182685.

Smith, D. G. A., Burns, L. A., Sirianni, D. A., Nascimento, D. R., Kumar, A., James, A. M., Schriber, J. B., Zhang, T., Zhang, B., Abbott, A. S., Berquist, E. J., Lechner, M. H., Cunha, L. A., Heide, A. G., Waldrop, J. M., Takeshita, T. Y., Alenaizan, A., Neuhauser, D., King, R. A., … Sherrill, C. D. (2020). Psi4 1.4: Open-source software for high-throughput quantum chemistry. *The Journal of Chemical Physics, 152*(18), 184108. doi: 10.1063/5.0006002. (AIP Publishing)

Sun, Q., Berkelbach, T. C., Blunt, N. S., Booth, G. H., Guo, S., Li, Z., Liu, J., McClain, J. D., Sayfutyarova, E. R., Sharma, S., Wouters, S., & Chan, G. K.-L. (2018). PySCF: The Python-based simulations of chemistry framework. *WIREs Computational Molecular Science, 8*(1), e1340. doi: 10.1002/wcms.1340. (Wires Online Library)

Sun, Q., Zhang, X., Banerjee, S., Bao, P., Barbry, M., Blunt, N. S., Bogdanov, N. A., Booth, G. H., Chen, J., Cui, Z.-H., Eriksen, J. J., Gao, Y., Guo, S., Hermann, J., Hermes, M. R., Koh, K., Koval, P., Lehtola, S., Li, Z., … Chan, G. K.-L. (2020). Recent developments in the PySCF program package. *The Journal of Chemical Physics, 153*(2), 024109. doi: 10.1063/5.0006074.

E. Cheminformatics, Representasi Molekul, dan Descriptor

Daylight Chemical Information Systems. (2026). *SMILES—A simplified chemical language*. Daylight Theory Manual.

Heller, S. R., McNaught, A., Pletnev, I., Stein, S., & Tchekhovskoi, D. (2015). InChI, the IUPAC International Chemical Identifier. *Journal of Cheminformatics, 7*, 23. doi: 10.1186/s13321-015-0068-4.

Krenn, M., Häse, F., Nigam, A. K., Friederich, P., & Aspuru-Guzik, A. (2020). Self-referencing embedded strings: A 100% robust molecular string representation. *Machine Learning: Science and Technology, 1*(4), 045024. doi: 10.1088/2632-2153/aba947. (arXiv)

Landrum, G. (2026). *RDKit: Open-source cheminformatics software*. RDKit Documentation. RDKit adalah toolkit cheminformatics dan machine-learning berbasis C++/Python yang aktif digunakan untuk struktur 2D/3D, descriptor, fingerprint, dan operasi molekul. (GitHub)

Moriwaki, H., Tian, Y.-S., Kawashita, N., & Takagi, T. (2018). Mordred: A molecular descriptor calculator. *Journal of Cheminformatics, 10*, 4. doi: 10.1186/s13321-018-0258-y. (Springer)

O'Boyle, N. M., Banck, M., James, C. A., Morley, C., Vandermeersch, T., & Hutchison, G. R. (2011). Open Babel: An open chemical toolbox. *Journal of Cheminformatics, 3*, 33. doi: 10.1186/1758-2946-3-33. (Springer)

Rogers, D., & Hahn, M. (2010). Extended-connectivity fingerprints. *Journal of Chemical Information and Modeling, 50*(5), 742–754. doi: 10.1021/ci100050t.

Weininger, D. (1988). SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences, 28*(1), 31–36. doi: 10.1021/ci00057a005.

F. Molecular Docking, Virtual Screening, dan Interaksi Protein–Ligan

Forli, S., Huey, R., Pique, M. E., Sanner, M. F., Goodsell, D. S., & Olson, A. J. (2016). Computational protein–ligand docking and virtual drug screening with the AutoDock suite. *Nature Protocols, 11*, 905–919. doi: 10.1038/nprot.2016.051.

Goodsell, D. S., Morris, G. M., & Olson, A. J. (1996). Automated docking of flexible ligands: Applications of AutoDock. *Journal of Molecular Recognition, 9*(1), 1–5. doi: 10.1002/(SICI)1099-1352(199601)9:1<1::AID-JMR241>3.0.CO;2-6.

Morris, G. M., Huey, R., Lindstrom, W., Sanner, M. F., Belew, R. K., Goodsell, D. S., & Olson, A. J. (2009). AutoDock4 and AutoDockTools4: Automated docking with selective receptor flexibility. *Journal of Computational Chemistry, 30*(16), 2785–2791. doi: 10.1002/jcc.21256. (PubMed)

Trott, O., & Olson, A. J. (2010). AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of Computational Chemistry, 31*(2), 455–461. doi: 10.1002/jcc.21334. (PMC)

Warren, G. L., Andrews, C. W., Capelli, A.-M., Clarke, B., LaLonde, J., Lambert, M. H., Lindvall, M., Nevins, N., Semus, S. F., Senger, S., Tedesco, G., Wall, I. D., Woolven, J. M., Peishoff, C. E., & Head, M. S. (2006). A critical assessment of docking programs and scoring functions. *Journal of Medicinal Chemistry, 49*(20), 5912–5931. doi: 10.1021/jm050362n.

G. ADMET, Drug-Likeness, dan Prediksi Farmakokinetik

Daina, A., Michielin, O., & Zoete, V. (2017). SwissADME: A free web tool to evaluate pharmacokinetics, drug-likeness and medicinal chemistry friendliness of small molecules. *Scientific Reports, 7*, 42717. doi: 10.1038/srep42717. SwissADME juga masih tersedia sebagai web tool aktif untuk fisikokimia, farmakokinetik, drug-likeness, dan medicinal chemistry friendliness. (SwissADME)

Lipinski, C. A., Lombardo, F., Dominy, B. W., & Feeney, P. J. (2001). Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Advanced Drug Delivery Reviews, 46*(1–3), 3–26. doi: 10.1016/S0169-409X(00)00129-0.

Pires, D. E. V., Blundell, T. L., & Ascher, D. B. (2015). pkCSM: Predicting small-molecule pharmacokinetic and toxicity properties using graph-based signatures. *Journal of Medicinal Chemistry, 58*(9), 4066–4072. doi: 10.1021/acs.jmedchem.5b00104. (PMC)

Veber, D. F., Johnson, S. R., Cheng, H.-Y., Smith, B. R., Ward, K. W., & Kopple, K. D. (2002). Molecular properties that influence the oral bioavailability of drug candidates. *Journal of Medicinal Chemistry, 45*(12), 2615–2623. doi: 10.1021/jm020017n.

Wishart, D. S., Feunang, Y. D., Guo, A. C., Lo, E. J., Marcu, A., Grant, J. R., Sajed, T., Johnson, D., Li, C., Sayeeda, Z., Assempour, N., Iynkkaran, I., Liu, Y., Maciejewski, A., Gale, N., Wilson, A., Chin, L., Cummings, R., Le, D., … Wilson, M. (2018). DrugBank 5.0: A major update to the DrugBank database. *Nucleic Acids Research, 46*(D1), D1074–D1082. doi: 10.1093/nar/gkx1037.

H. Molecular Dynamics dan Analisis Trajectory

Abraham, M. J., Murtola, T., Schulz, R., Páll, S., Smith, J. C., Hess, B., & Lindahl, E. (2015). GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX, 1–2*, 19–25. doi: 10.1016/j.softx.2015.06.001.

Case, D. A., Aktulga, H. M., Belfon, K., Ben-Shalom, I., Brozell, S. R., Cerutti, D. S., Cheatham, T. E., III, Cruzeiro, V. W. D., Darden, T. A., Duke, R. E., Giambasu, G., Gilson, M. K., Gohlke, H., Goetz, A. W., Harris, R., Izadi, S., Izmailov, S. A., Kasavajhala, K., Kovalenko, A., … Kollman, P. A. (2023). *Amber 2023*. University of California, San Francisco.

Eastman, P., Galvelis, R., Peláez, R. P., Abreu, C. R. A., Farr, S. E., Gallicchio, E., Gorenko, A., Henry, M. M., Hu, F., Huang, J., Krämer, A., Michel, J., Mitchell, J. A., Pande, V. S., Rodrigues, J. P. G. L. M., Rodriguez-Guerra, J., Simmonett, A. C., Singh, S., Swails, J., … Chodera, J. D. (2024). OpenMM 8: Molecular dynamics simulation with machine learning potentials. *The Journal of Physical Chemistry B, 128*(1), 109–116. doi: 10.1021/acs.jpcb.3c06662. (ACS Publications)

Gowers, R. J., Linke, M., Barnoud, J., Reddy, T. J. E., Melo, M. N., Seyler, S. L., Dotson, D. L., Domanski, J., Buchoux, S., Kenney, I. M., & Beckstein, O. (2016). MDAnalysis: A Python package for the rapid analysis of molecular dynamics simulations. In S. Benthall & S. Rostrup (Eds.), *Proceedings of the 15th Python in Science Conference* (pp. 98–105). SciPy. doi: 10.25080/Majora-629e541a-00e. (docs.mdanalysis.org)

McGibbon, R. T., Beauchamp, K. A., Harrigan, M. P., Klein, C., Swails, J. M., Hernández, C. X., Schwantes, C. R., Wang, L.-P., Lane, T. J., & Pande, V. S. (2015). MDTraj: A modern open library for the analysis of molecular dynamics trajectories. *Biophysical Journal, 109*(8), 1528–1532. doi: 10.1016/j.bpj.2015.08.015.

Phillips, J. C., Hardy, D. J., Maia, J. D. C., Stone, J. E., Ribeiro, J. V., Bernardi, R. C., Buch, R., Fiorin, G., Hénin, J., Jiang, W., McGreevy, R., Melo, M. C. R., Radak, B. K., Skeel, R. D., Singharoy, A., Wang, Y., Roux, B., Aksimentiev, A., Luthey-Schulten, Z., … Tajkhorshid, E. (2020). Scalable molecular dynamics on CPU and GPU architectures with NAMD. *The Journal of Chemical Physics, 153*(4), 044130. doi: 10.1063/5.0014475.

I. Machine Learning, QSAR, dan AI untuk Kimia

Cherkasov, A., Muratov, E. N., Fourches, D., Varnek, A., Baskin, I. I., Cronin, M., Dearden, J., Gramatica, P., Martin, Y. C., Todeschini, R., Consonni, V., Kuz'min, V. E., Cramer, R., Benigni, R., Yang, C., Rathman, J., Terfloth, L., Gasteiger, J., Richard, A., & Tropsha, A. (2014). QSAR modeling: Where have you been? Where are you going to? *Journal of Medicinal Chemistry, 57*(12), 4977–5010. doi: 10.1021/jm4004285.

Cova, T. F. G. G., & Pais, A. A. C. C. (2019). Deep learning for deep chemistry: Optimizing the prediction of chemical patterns. *Frontiers in Chemistry, 7*, 809. doi: 10.3389/fchem.2019.00809. (Frontiers)

Korshunova, M., Ginsburg, B., Tropsha, A., & Isayev, O. (2021). OpenChem: A deep learning toolkit for computational chemistry and drug design. *Journal of Chemical Information and Modeling, 61*(1), 7–13. doi: 10.1021/acs.jcim.0c00971.

Lim, S., Lu, Y., Cho, C. Y., Sung, I., Kim, J., Kim, Y., Park, S., & Kim, S. (2022). On modeling and utilizing chemical compound information with deep learning technologies: A task-oriented approach. *Computational and Structural Biotechnology Journal, 20*, 5460–5472. doi: 10.1016/j.csbj.2022.09.017. (PMC)

Wu, Z., Ramsundar, B., Feinberg, E. N., Gomes, J., Geniesse, C., Pappu, A. S., Leswing, K., & Pande, V. (2018). MoleculeNet: A benchmark for molecular machine learning. *Chemical Science, 9*(2), 513–530. doi: 10.1039/C7SC02664A.

J. Database Struktur, Protein, dan Senyawa

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., & Bourne, P. E. (2000). The Protein Data Bank. *Nucleic Acids Research, 28*(1), 235–242. doi: 10.1093/nar/28.1.235.

Kim, S., Chen, J., Cheng, T., Gindulyte, A., He, J., He, S., Li, Q., Shoemaker, B. A., Thiessen, P. A., Yu, B., Zaslavsky, L., Zhang, J., & Bolton, E. E. (2023). PubChem 2023 update. *Nucleic Acids Research, 51*(D1), D1373–D1380. doi: 10.1093/nar/gkac956.

Sterling, T., & Irwin, J. J. (2015). ZINC 15—Ligand discovery for everyone. *Journal of Chemical Information and Modeling, 55*(11), 2324–2337. doi: 10.1021/acs.jcim.5b00559.

UniProt Consortium. (2023). UniProt: The Universal Protein Knowledgebase in 2023. *Nucleic Acids Research, 51*(D1), D523–D531. doi: 10.1093/nar/gkac1052.

K. Riset Anti-Kanker Payudara, Aromatase, PARP, dan Senyawa Terkait

Anastrozole. (2026). *DrugBank Online*. DrugBank.

Aromatase. (2026). *UniProt Knowledgebase*. UniProt Consortium.

Bardia, A., Hurvitz, S. A., Tolaney, S. M., Loirat, D., Punie, K., Oliveira, M., Brufsky, A., Sardesai, S. D., Kalinsky, K., Zelnak, A. B., Weaver, R., Traina, T., Dalenc, F., Aftimos, P., Lynce, F., Diéras, V., Cortés, J., Schmid, P., Carey, L. A., … Rugo, H. S. (2021). Sacituzumab govitecan in metastatic triple-negative breast cancer. *The New England Journal of Medicine, 384*(16), 1529–1541. doi: 10.1056/NEJMoa2028485.

Burstein, H. J., Curigliano, G., Thürlimann, B., Weber, W. P., Poortmans, P., Regan, M. M., Senn, H. J., Winer, E. P., & Gnant, M. (2021). Customizing local and systemic therapies for women with early breast cancer: The St. Gallen International Consensus Guidelines for treatment of early breast cancer 2021. *Annals of Oncology, 32*(10), 1216–1235. doi: 10.1016/j.annonc.2021.06.023.

Early Breast Cancer Trialists' Collaborative Group. (2015). Aromatase inhibitors versus tamoxifen in early breast cancer: Patient-level meta-analysis of the randomised trials. *The Lancet, 386*(10001), 1341–1352. doi: 10.1016/S0140-6736(15)61074-1.

Letrozole. (2026). *DrugBank Online*. DrugBank.

Lord, C. J., & Ashworth, A. (2017). PARP inhibitors: Synthetic lethality in the clinic. *Science, 355*(6330), 1152–1158. doi: 10.1126/science.aam7344.

Olaparib. (2026). *DrugBank Online*. DrugBank.

Robson, M., Im, S.-A., Senkus, E., Xu, B., Domchek, S. M., Masuda, N., Delaloge, S., Li, W., Tung, N., Armstrong, A., Wu, W., Goessl, C., Runswick, S., & Conte, P. (2017). Olaparib for metastatic breast cancer in patients with a germline BRCA mutation. *The New England Journal of Medicine, 377*(6), 523–533. doi: 10.1056/NEJMoa1706450.

Talazoparib. (2026). *DrugBank Online*. DrugBank.

L. Reproducibility, Pelaporan, dan Etika Riset Komputasi

Allison, J. R. (2017). Computational methods for exploring protein conformations. *Biochemical Society Transactions, 45*(5), 1135–1144. doi: 10.1042/BST20170094.

Barba, L. A. (2018). Terminologies for reproducible research. *arXiv*. arXiv:1802.03311.

Donoho, D. L. (2010). An invitation to reproducible computational research. *Biostatistics, 11*(3), 385–388. doi: 10.1093/biostatistics/kxq028.

Ioannidis, J. P. A. (2005). Why most published research findings are false. *PLoS Medicine, 2*(8), e124. doi: 10.1371/journal.pmed.0020124.

Sandve, G. K., Nekrutenko, A., Taylor, J., & Hovig, E. (2013). Ten simple rules for reproducible computational research. *PLoS Computational Biology, 9*(10), e1003285. doi: 10.1371/journal.pcbi.1003285.

Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., … Mons, B. (2016). The FAIR guiding principles for scientific data management and stewardship. *Scientific Data, 3*, 160018. doi: 10.1038/sdata.2016.18.

M. Dokumentasi Resmi dan Sumber Web Aktif untuk Praktikum

ASE Developers. (2026). *Atomic Simulation Environment documentation*. ASE Community. ASE menyediakan platform Python untuk membuat, memanipulasi, menjalankan, memvisualisasi, dan menganalisis simulasi atomistik. (ASE community)

GPAW Developers. (2026). *GPAW documentation*. GPAW Project. (gpaw.readthedocs.io)

MDAnalysis Developers. (2026). *MDAnalysis documentation*. MDAnalysis Project. Dokumentasi MDAnalysis juga memuat rujukan resmi untuk sitasi software. (docs.mdanalysis.org)

OpenMM Developers. (2026). *OpenMM user guide*. OpenMM Project. (docs.openmm.org)

RDKit Developers. (2026). *RDKit documentation and GitHub repository*. RDKit Project. (GitHub)

Swiss Institute of Bioinformatics. (2026). *SwissADME*. SIB Swiss Institute of Bioinformatics. (SwissADME)

The University of Queensland. (2026). *pkCSM: Predicting small-molecule pharmacokinetic properties*. Biosig Lab. (Biosig)

Penulis,

KASMUI

RISET KIMIA KOMPUTASI MENGGUNAKAN PYTHON

+ RISET SENYAWA ANTI-KANKER PAYUDARA

SINOPSIS ISI BUKU

Buku ini membimbing pembaca memahami riset kimia komputasi berbasis Python secara runtut, mulai dari ekosistem library utama seperti PySCF, RDKit, Open Babel, ASE, dan alat analisis terkait, hingga penerapannya untuk studi senyawa anti-kanker payudara. Pembahasan mencakup representasi molekul, desain topik riset, penyusunan proposal skripsi, optimasi geometri, perhitungan DFT, docking, prediksi ADMET, reproducibility, serta penulisan laporan ilmiah.

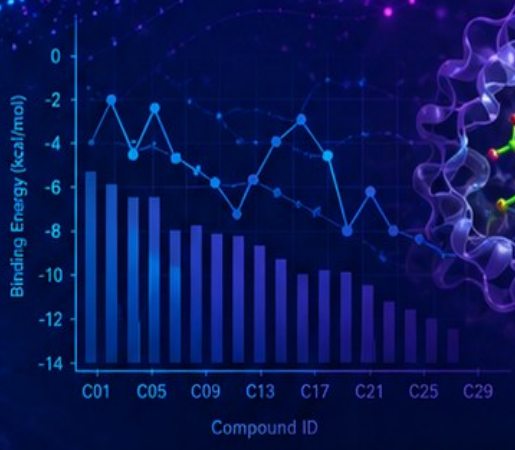
Disusun sebagai buku ajar, panduan praktikum, dan referensi riset S1, buku ini juga dilengkapi contoh alur kerja, implementasi kode Python/PySCF, panduan pelaksanaan riset enam bulan, modul praktikum terstruktur, serta bank tugas dan latihan analisis. Dengan pendekatan yang sistematis, buku ini membantu mahasiswa, dosen, dan peneliti pemula menjembatani konsep kimia, pemrograman ilmiah, dan pengembangan kandidat senyawa secara in silico.

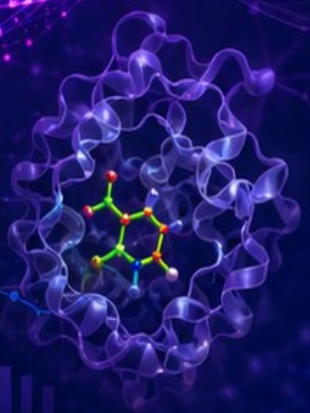
```python
import numpy as np
from pyscf import gto, scf, dft
from rdkit import Chem
from rdkit.Chem import AllChem
import matplotlib.pyplot as plt

mol = Chem.MolFromSmiles('CC(=O)NC1....')

mf = dft.RKS(mol)
mf.xc = 'B3LYP'
mf.kernel()

energy = mf.e_tot
print(f"DFT Energy: {energy:.6f} Eh")
```





KASMUI