Bab 1

Pendahuluan dan Orientasi OBE

Sub-CPMK yang Dicakup dalam Bab Ini:

- Memahami keterkaitan antara kurikulum *Outcome-Based Education* (OBE) dengan desain sistem kompilator.
- Mengidentifikasi alur pembelajaran Teknik Kompilasi melalui peta konsep buku ajar.
- Merencanakan strategi belajar mandiri untuk menguasai setiap fase kompilasi secara sistematis.

1.1 Tujuan Buku Ajar

Buku ajar ini dirancang sebagai panduan komprehensif untuk menguasai ***Teknik Kompilasi*** secara sistematis dan terukur, selaras dengan standar *Outcome-Based Education* (OBE) [1]. Fokus utama buku ini adalah pada pembangunan fondasi teoretis dan keterampilan praktis dalam membangun arsitektur kompilator modern. Menurut [2], kompilasi adalah proses transformasi dari bahasa sumber ke bahasa sasaran.

1. Memberikan pemahaman mendalam tentang setiap fase kompilasi, mulai dari analisis leksikal hingga generasi kode target.
2. Mengembangkan kemampuan merancang dan mengimplementasikan komponen-komponen utama kompilator seperti *lexer*, *parser*, dan *semantic analyzer*.
3. Membangun keterampilan dalam optimasi kode dan manajemen memori pada *runtime*.
4. Memfasilitasi pencapaian *Capaian Pembelajaran Lulusan* (CPL) dan *Capaian Pembelajaran Mata Kuliah* (CPMK) yang telah ditetapkan dalam kurikulum.

Setelah mempelajari buku ini secara menyeluruh, mahasiswa diharapkan mampu:

- Menjelaskan arsitektur kompilator dan fungsi setiap fasenya.
- Membangun pemroses bahasa (*language processor*) menggunakan teknik manual maupun generator (*Flex/Bison*).
- Mengelola struktur data kompleks seperti *Symbol Table* dan *Abstract Syntax Tree* (AST).
- Menghasilkan kode target yang efisien untuk arsitektur mesin tertentu.
- Melakukan evaluasi performa dan optimasi pada tingkat *intermediate code* dan *target code*.

1.2 Keterkaitan Buku Ajar dengan RPS Berbasis OBE

Buku ajar ini dirancang selaras dengan Rencana Pembelajaran Semester (RPS) mata kuliah Teknik Kompilasi yang berbasis *Outcome-Based Education* (OBE). Keterkaitan ini memastikan bahwa setiap materi yang disajikan memiliki kontribusi langsung terhadap kompetensi lulusan, sebagaimana diterapkan dalam standar kurikulum internasional [3].

1.3 Arsitektur Kompilator Modern

Arsitektur kompilator modern umumnya terbagi menjadi front-end dan back-end [4].

1.3.1 Alignment dengan CPL dan CPMK

Struktur buku ini disusun untuk mendukung pencapaian indikator-indikator berikut:

- **CPL-1 (Pengetahuan)**: Menguasai konsep teoretis analisis leksikal, sintaksis, semantik, dan generasi kode secara mendalam.
- **CPL-3 (Keterampilan Khusus)**: Mampu merancang, mengimplementasikan, dan mengevaluasi sistem kompilator lengkap.
- **CPMK-1 s.d CPMK-6**: Meliputi seluruh spektrum pengembangan kompilator dari arsitektur awal hingga evaluasi performa akhir.

Setiap bab dalam buku ini memuat daftar **Sub-CPMK** di bagian awal untuk memberikan fokus yang jelas bagi mahasiswa mengenai kompetensi spesifik yang akan dikuasai.

1.3.2 Integrasi Metode Pembelajaran Aktif

Sesuai dengan RPS berbasis OBE, buku ini mendukung berbagai metode pembelajaran:

- **Problem-Based Learning**: Melalui studi kasus penanganan *semantic errors* dan optimasi lokal.
- **Project-Based Learning**: Pengembangan kompilator secara bertahap dalam setiap bab.
- **Praktikum Terbimbing**: Implementasi komponen menggunakan *tooling* industri seperti LLVM atau Clang.

1.3.3 Sistem Evaluasi Berbasis Kompetensi

Komponen asesmen yang disediakan di setiap akhir bab (latihan, asesmen, dan *checklist*) dirancang untuk mengukur pencapaian *Sub-CPMK* secara objektif, yang nantinya akan menjadi bobot penilaian utama dalam UTS dan UAS (total 35% sesuai RPS).

1.4 Petunjuk Penggunaan Buku Ajar

1.4.1 Untuk Mahasiswa

Mengingat kompleksitas pengembangan kompilator, mahasiswa disarankan untuk menggunakan buku ini dengan langkah-langkah berikut:

Tahap Persiapan:

1. Pahami target *Sub-CPMK* di awal bab agar fokus belajar tetap terjaga.
2. Tinjau kembali materi prasyarat (Struktur Data dan Algoritma) jika diperlukan.

Tahap Implementasi (Eksperimental):

1. Pelajari kode contoh yang disediakan dan jalankan menggunakan *compiler* atau *interpreter* yang sesuai.
2. Lakukan modifikasi pada parameter *lexer* atau aturan *grammar* untuk melihat dampaknya terhadap proses *parsing*.
3. Gunakan perangkat lunak pendukung seperti *Flex*, *Bison*, atau *Graphviz* untuk memvisualisasikan AST.

Tahap Evaluasi:

1. Kerjakan latihan refleksi untuk memperdalam pemahaman teoretis.

2. Lakukan penilaian mandiri menggunakan *checklist* kompetensi di akhir bab.

3. Gabungkan komponen yang telah dibuat di setiap bab menjadi satu proyek kompilator utuh.

1.4.2 Untuk Dosen

Dosen dapat memanfaatkan buku ini sebagai instrumen pembelajaran utama:

- **Modul Praktikum**: Gunakan aktivitas pembelajaran di setiap bab sebagai panduan tugas mingguan.
- **Bank Soal**: Manfaatkan bagian asesmen sebagai referensi dalam menyusun soal UTS (CPMK-1, 2) dan UAS (CPMK 1 s.d 6).
- **Alat Ukur Capaian**: Gunakan rubrik penilaian dan indikator dalam buku untuk mengukur ketercapaian outcomes mahasiswa.

1.5 Konteks Kurikulum OBE

1.5.1 Filosofi OBE dalam Teknik Kompilasi

Outcome-Based Education (OBE) adalah pendekatan yang menekankan pada apa yang bisa dilakukan oleh mahasiswa di akhir masa studi, bukan sekadar apa yang diajarkan. Dalam Teknik Kompilasi, hal ini berarti mahasiswa tidak hanya menghafal algoritma *parsing*, tetapi mampu membangun sebuah program yang secara nyata dapat menerjemahkan sebuah bahasa ke bahasa lain.

Empat Prinsip Utama OBE dalam Buku Ini:

1. **Clarity of Focus**: Fokus pada hasil akhir berupa kompilator yang berfungsi.
2. **Designing Down**: Materi disusun mundur dari kebutuhan akhir sebuah sistem *backend* kompilator.
3. **High Expectations**: Mahasiswa didorong untuk mengimplementasikan optimasi kode yang efisien.
4. **Expanded Opportunity**: Menyediakan berbagai aktivitas belajar mulai dari teori hingga proyek tim.

1.5.2 Implementasi Tahapan OBE

Buku ini membagi proses pencapaian kompetensi dalam empat pilar utama:

Komponen OBE	Implementasi dalam Teknik Kompilasi
Defined Outcomes	Sub-CPMK eksplisit untuk setiap fase (Leksikal, Sintaksis, dst).
Designing Down	Kurikulum dimulai dari pengenalan bahasa ke deteksi kesalahan hingga emisi kode.
Student Activity	Implementasi manual mesin *state* dan penggunaan alat otomatisasi generator.
Continuous Assessment	*Weekly reflection* dan audit kualitas kode secara berkala.

Tabel 1.1: Penerapan OBE dalam Pengembangan Kompilator

1.5.3 Hierarki Capaian Pembelajaran

Konsep Penting

Pemahaman mahasiswa terhadap Teknik Kompilasi divalidasi melalui hierarki capaian:

- **CPL**: Mahasiswa menguasai teori kompilasi secara utuh sebagai sarjana teknik.
- **CPMK**: Mahasiswa mampu mengintegrasikan fase-fase kompilasi menjadi sistem yang koheren.
- **Sub-CPMK**: Mahasiswa mahir dalam satu spesialisasi fase (misalnya: *Register Allocation*).

1.6 Peta Konsep Teknik Kompilasi

Buku ini disusun dalam 19 bab yang mencakup seluruh spektrum pengembangan kompilator, mulai dari landasan teori hingga referensi pengembangan lanjut:

1. **Bab I**: Pengenalan dan Konteks OBE
2. **Bab II**: Arsitektur Kompilator - Gambaran umum sistem
3. **Bab III-IV**: *Front-end* - Analisis leksikal dan representasi regular
4. **Bab V-VI**: *Syntax Analysis* - *Parsing* dan *grammar* formal
5. **Bab VII-X**: *Middle-end* - *Intermediate code*, tabel simbol, analisis semantik, dan penanganan kesalahan
6. **Bab XI-XIV**: *Back-end* - Tata letak memori, *code generation*, alokasi register, dan manajemen *stack*
7. **Bab XV-XVI**: *Analysis & Evaluation* - *Compiler tools* dan evaluasi performa

8. **Bab XVII-XIX**: *Assessment & Resources* - Evaluasi kompetensi, lampiran, dan daftar referensi

Alur Pembelajaran:

- **Fase Analisis (Bab II-VI)**: Memahami bagaimana bahasa manusia diterjemahkan menjadi token dan pohon hirarki.
- **Fase Transformasi (Bab VII-X)**: Memastikan kebenaran makna dan mengubahnya menjadi representasi antara.
- **Fase Sintesis (Bab XI-XIV)**: Membangun instruksi mesin yang optimal sesuai arsitektur target.
- **Fase Profesional (Bab XV-XIX)**: Menggunakan alat bantu modern dan melakukan standarisasi kualitas.

1.7 Proyek Buku: Compiler Subset C

Sepanjang Bab 2 hingga Bab 16, kita secara bertahap membangun **satu compiler untuk subset bahasa C**. Setiap bab menambah satu lapis ke proyek yang sama: spesifikasi token (Bab 3), lexer hand-written (Bab 3), lexer Flex (Bab 4), grammar (Bab 5), parser hand-written (Bab 6), teori bottom-up (Bab 7), parser Bison (Bab 8), AST (Bab 9), symbol table (Bab 10), type checking (Bab 11), IR (Bab 12), runtime (Bab 13), code generation (Bab 14), optimasi (Bab 15), dan integrasi (Bab 16). Spesifikasi berikut menjadi acuan tunggal agar semua contoh dan kode mengacu ke bahasa yang sama.

1.7.1 Spesifikasi Token Proyek Subset C

Token yang dikenali oleh compiler proyek (untuk Bab 3–4):

- **Identifier**: huruf atau underscore diikuti huruf, angka, atau underscore. Pola: `[a-zA-Z_][a-zA-Z0-9_]*`
- **Kata kunci**: `int`, `float`, `print`. (Nanti dapat diperluas: `if`, `else`, `while`.)
- **Literal**: integer `[0-9]+`, float `[0-9]+.[0-9]+`, string `"..."` dalam tanda kutip ganda.
- **Operator**: `+`, `-`, `*`, `/`, `=`, `==`, `!=`, `<`, `>`, `<=`, `>=`.
- **Punctuator**: `;`, `,`, `()`, kurung kurawal `{}`.
- **Komentar**: satu baris `//` dan banyak baris `/* */`; serta whitespace (spasi, tab, newline) diabaikan.

1.7.2 Spesifikasi Grammar Proyek Subset C

Grammar dalam BNF untuk Bab 5–8 (dan parser proyek):

- **Program**: barisan statement.
- **Statement**: deklarasi `;` | assignment `;` | print-statement `;`
- **Deklarasi**: `int` identifier | `float` identifier
- **Assignment**: identifier = ekspresi
- **Print-statement**: `print (` string-literal `)` | `print (` ekspresi `)`
- **Ekspresi**: term | ekspresi + term | ekspresi − term
- **Term**: factor | term $\star$ factor | term / factor
- **Factor**: literal | identifier | `(` ekspresi `)`

Precedence: $\star$ dan / lebih tinggi dari + dan −; associativity kiri untuk semuanya.

1.7.3 Peta Bab ke Lapis Proyek

Bab	Lapis proyek
3	Spesifikasi token + teori RE/FA
3	Lexer hand-written (mengikuti spec token)
4	Lexer proyek (Flex, file `simplec.l`)
5	Grammar proyek (BNF/EBNF di atas)
6	Parser hand-written (mengikuti grammar proyek)
7	Teori LR; grammar proyek termasuk kelas LR
8	Parser proyek (Bison, file `simplec.y`)
9	AST proyek (`ast.h`/`ast.c`)
10	Symbol table proyek (`symtab.h`/`symtab.c`)
11	Type checking proyek
12	IR proyek (TAC/quadruples dari AST)
13	Runtime; asumsi proyek untuk stack/activation record
14	Code generation proyek (IR $\rightarrow$ assembly)
15	Optimasi proyek (basic block, constant folding, dll.)
16	Integrasi dan presentasi compiler subset C lengkap

Semua bab dari Bab 3 sampai Bab 16 merujuk ke spesifikasi ini. Kode dan contoh dalam bab tersebut mengacu ke token set dan grammar di atas, serta ke file proyek (`simplec.l`, `simplec.y`, dan seterusnya) yang tumbuh di folder `proyek-compiler-subset-c/`.

Aktivitas Pembelajaran

1. **Analisis RPS**: Pelajari RPS Teknik Kompilasi dan identifikasi bagaimana CPL Pengetahuan (CPL-1) diukur melalui proyek pengembangan kompilator.

2. **Pemetaan Fase**: Buat diagram alir yang memetakan bab-bab dalam buku ini ke dalam tiga pilar utama: *Front-end*, *Middle-end*, dan *Back-end*.

3. **Tooling Audit**: Identifikasi *software* pendukung (GCC, Flex, Bison, Clang) yang akan digunakan di setiap bab berdasarkan peta konsep.

4. **Diskusi Kurikulum**: Diskusikan mengapa penguasaan Teknik Kompilasi sangat krusial dalam mencapai standar kompetensi lulusan Teknik Informatika di industri modern.

Latihan dan Refleksi

1. Jelaskan secara spesifik bagaimana pendekatan OBE membantu mahasiswa dalam menghadapi kompleksitas algoritma *parsing*!

2. Mengapa setiap aktivitas praktikum harus memiliki rubrik penilaian yang eksplisit dalam konteks OBE?

3. Bagaimana cara Anda memantau kemajuan pembangunan proyek kompilator Anda menggunakan *checklist* kompetensi?

4. Hubungkan antara *Target Architecture* (Bab XII) dengan tujuan akhir pencapaian kompetensi dalam RPS!

5. **Refleksi**: Sejauh mana Anda memahami bahwa membangun sebuah kompilator adalah bukti nyata pencapaian kompetensi teknik yang utuh?

Asesmen (Evaluasi Kinerja)

Instrumen Penilaian untuk Orientasi OBE Kompilator

A. Pilihan Ganda

1. Manakah yang merupakan contoh *Outcome* nyata dalam mata kuliah ini?

(a) Membaca buku Naga (Dragon Book)

(b) Lulus ujian teori

(c) Menghasilkan kode assembly yang dapat dijalankan

(d) Mengetahui sejarah FORTRAN

2. *Designing Down* dalam buku ini berarti:

(a) Mendesain dari tingkat mesin ke bahasa tingkat tinggi

(b) Menyusun materi berdasarkan urutan fase kompilasi untuk mencapai produk akhir

(c) Mengurangi beban materi yang sulit

(d) Hanya fokus pada latihan praktak

B. Essay

1. Jelaskan keterkaitan antara peta konsep (Bab I s.d XIX) dengan pencapaian kompetensi profesional seorang *System Programmer*!

2. Desainlah satu indikator pencapaian kompetensi untuk Sub-CPMK "Membangun Lexer Hand-written"!

Rubrik Penilaian: Lihat Lampiran A

Checklist Pencapaian Kompetensi

Centang item berikut setelah Anda yakin telah menguasainya:

☐ Saya memahami filosofi OBE dalam konteks pengembangan sistem kompilator.

☐ Saya dapat memetakan hierarki CPL, CPMK, dan Sub-CPMK Teknik Kompilasi ke dalam konten buku.

☐ Saya memahami alur kerja *Front-end*, *Middle-end*, dan *Back-end* melalui peta konsep.

☐ Saya telah menyiapkan *software stack* yang diperlukan untuk proyek semester ini.

☐ Saya berkomitmen untuk melakukan *self-assessment* secara berkala di setiap akhir bab.

Rangkuman

Bab ini memberikan fondasi bagi mahasiswa untuk memahami bagaimana buku ajar ini

disusun menggunakan standar OBE guna menjamin penguasaan Teknik Kompilasi yang utuh dan profesional.

Poin Kunci:

- Fokus utama adalah pencapaian *outcome* berupa sistem kompilator yang berfungsi.
- Kurikulum dirancang mundur (*designing down*) untuk memandu mahasiswa dari analisis ke sintesis kode.
- Peta konsep menunjukkan integrasi 19 bab sebagai satu kesatuan ekosistem pembelajaran.
- Peran aktif mahasiswa dalam evaluasi diri adalah kunci keberhasilan dalam sistem OBE.

Kata Kunci: ***OBE***, ***Teknik Kompilasi***, ***Peta Konsep***, ***CPL***, ***Outcome***, ***Compiler Design***

Daftar Pustaka

[1] StudyLib. *Outcomes-Based Education Curricula*. Materials on runtime environment and activation records. 2024. URL: https://studylib.net/doc/14111770/outcomes-based-education-curricula--academic-year-2015-.

[2] Alfred V. Aho **andothers**. *Compilers: Principles, Techniques, and Tools*. 2nd. Dragon Book. Pearson Education, 2006.

[3] Northeastern University. *CS 4410/6410: Compiler Design*. Course syllabus and materials, taught by Benjamin Lerner. 2024. URL: https://course.ccs.neu.edu/cs4410sp25/.

[4] Keith D. Cooper **and** Linda Torczon. *Engineering a Compiler*. 2nd. Morgan Kaufmann, 2011.