

# **RENCANA PEMBELAJARAN SEMESTER (RPS)**

**Berbasis Outcome-Based Education (OBE)**

**PROGRAM STUDI TEKNIK INFORMATIKA**  
**FAKULTAS TEKNIK**  
**UNIVERSITAS LOREM IPSUM**

**MATA KULIAH TEKNIK KOMPILASI**

## **1. Identitas Mata Kuliah**

Nama Program Studi	:	Teknik Informatika
Nama Mata Kuliah	:	Teknik Kompilasi
Kode Mata Kuliah	:	TIF-401
Semester	:	6 (Enam)
SKS / Bobot Kredit	:	3 SKS (2 Teori, 1 Praktikum)
Dosen Pengampu	:	Tim Dosen Teknik Informatika
Tanggal Penyusunan	:	5 Februari 2026
Prasyarat	:	Struktur Data, Algoritma, Pemrograman

## **2. Capaian Pembelajaran Lulusan (CPL)**

CPL yang dibebankan pada mata kuliah ini mencakup kompetensi lulusan dalam aspek pengetahuan, keterampilan, dan sikap:

- **CPL-1 (Pengetahuan):** Menguasai konsep teoretis bidang teknik kompilasi secara mendalam, termasuk analisis leksikal, sintaksis, semantik, dan generasi kode, serta mampu memformulasikan penyelesaian masalah prosedural dalam pengembangan kompilator.
- **CPL-2 (Keterampilan Umum):** Mampu menerapkan pemikiran logis, kritis, sistematis, dan inovatif dalam konteks pengembangan atau implementasi sistem kompilator yang memperhatikan dan menerapkan nilai humaniora dan etika profesi.
- **CPL-3 (Keterampilan Khusus):** Mampu merancang, mengimplementasikan, dan mengevaluasi sistem kompilator lengkap menggunakan teknik-teknik modern dengan mempertimbangkan efisiensi, optimasi, dan kualitas kode yang dihasilkan sesuai standar industri.
- **CPL-4 (Sikap):** Menunjukkan sikap bertanggung jawab atas pekerjaan di bidang keahliannya secara mandiri dan mampu bekerja sama dalam tim dalam proyek pengembangan perangkat lunak kompilator.

### 3. Capaian Pembelajaran Mata Kuliah (CPMK)

Kemampuan atau kompetensi spesifik yang diharapkan mahasiswa kuasai setelah menyelesaikan mata kuliah:

- **CPMK-1:** Mahasiswa mampu menjelaskan arsitektur kompilator secara keseluruhan, termasuk fase-fase utama: analisis leksikal, parsing, analisis semantik, generasi kode intermediate, optimasi, dan generasi kode.
- **CPMK-2:** Mahasiswa mampu membangun lexer dan parser menggunakan grammar formal (misalnya regular expressions, context-free grammars), finite automata (NFA/DFA), dan teknik parsing top-down dan bottom-up.
- **CPMK-3:** Mahasiswa mampu menerapkan syntax-directed translation dan analisis semantik untuk menegakkan aturan bahasa, menyelesaikan tipe, mengelola scope, dan menangani error.
- **CPMK-4:** Mahasiswa mampu merancang dan menghasilkan representasi kode intermediate (misalnya three-address code, DAGs) dan melakukan optimasi yang tidak ber-gantung pada mesin (optimasi basic block, data-flow analysis).
- **CPMK-5:** Mahasiswa mampu mengimplementasikan generasi kode untuk arsitektur target, memetakan kode intermediate menjadi kode target yang efisien, mengelola struktur run-time (activation records, memory layout, symbol tables), dan menerapkan optimasi khusus mesin dasar.
- **CPMK-6:** Mahasiswa mampu mengevaluasi dan membandingkan alat kompilator (seperti parser generators) dan pendekatan optimasi, menganalisis trade-off antara waktu kompilasi, kualitas kode, dan efisiensi runtime.

### 4. Sub-CPMK / Indikator Pencapaian

Penjabaran CPMK menjadi indikator yang lebih terukur dan dapat diuji:

- **Sub-CPMK 1.1:** Menjelaskan perbedaan antara interpreter dan compiler
- **Sub-CPMK 1.2:** Mengidentifikasi fase-fase kompilator dalam arsitektur kompilator nyata
- **Sub-CPMK 1.3:** Menganalisis trade-off antara one-pass vs multi-pass compiler
- **Sub-CPMK 2.1:** Membuat regular expression untuk token spesifik bahasa
- **Sub-CPMK 2.2:** Mengimplementasikan NFA dan DFA untuk token recognition
- **Sub-CPMK 2.3:** Membangun recursive descent parser untuk grammar LL(1)
- **Sub-CPMK 2.4:** Menggunakan parser generator (Flex/Bison) untuk bahasa sederhana

- **Sub-CPMK 3.1:** Mengimplementasikan symbol table dengan nested scopes
- **Sub-CPMK 3.2:** Melakukan type checking untuk ekspresi kompleks
- **Sub-CPMK 3.3:** Menangani semantic error dengan pesan yang informatif
- **Sub-CPMK 4.1:** Merancang three-address code representation
- **Sub-CPMK 4.2:** Mengimplementasikan basic block identification
- **Sub-CPMK 4.3:** Melakukan optimasi local (constant folding, dead code elimination)
- **Sub-CPMK 5.1:** Memetakan intermediate code ke assembly target
- **Sub-CPMK 5.2:** Mengimplementasikan register allocation sederhana
- **Sub-CPMK 5.3:** Mengelola activation records dan calling conventions
- **Sub-CPMK 6.1:** Membandingkan performa hand-written vs generated compiler
- **Sub-CPMK 6.2:** Menganalisis trade-off compilation time vs runtime efficiency

## 5. Materi Pembelajaran (Bahan Kajian)

Daftar topik materi yang relevan dengan Sub-CPMK dan CPMK:

1. Pengenalan Kompilator dan Arsitektur Kompilator
2. Regular Expression dan Finite Automata untuk Lexical Analysis
3. Implementasi Lexer Hand-Written dan Generator-Based
4. Context-Free Grammar dan Notasi BNF/EBNF
5. Top-Down Parsing (Recursive Descent) dan Bottom-Up Parsing (LR)
6. Parser Generator (Flex/Bison) dan Semantic Actions
7. Abstract Syntax Tree (AST) dan Struktur Data
8. Symbol Table dan Scope Management
9. Type Checking dan Semantic Analysis
10. Intermediate Code Generation (Three-Address Code)
11. Runtime Environment dan Memory Management
12. Code Generation untuk Target Architecture
13. Optimasi Kompilator (Basic Block, Constant Folding, Dead Code Elimination)
14. Teknik Modern: LLVM IR, JIT Compilation, dan Optimasi Levels
15. Evaluasi dan Perbandingan Compiler Tools

## 6. Metode Pembelajaran

Strategi atau pendekatan pembelajaran yang dipilih sesuai OBE yang menekankan aktivitas mahasiswa:

- **Ceramah Interaktif:** Penjelasan konsep kompilator dengan diskusi tanya jawab
- **Problem-Based Learning (PBL):** Mahasiswa menyelesaikan permasalahan komplikasi nyata
- **Project-Based Learning:** Pengembangan compiler lengkap sebagai proyek bertahap
- **Praktikum Terbimbing:** Implementasi komponen kompilator dengan bimbingan
- **Peer Review:** Mahasiswa melakukan code review terhadap implementasi compiler
- **Flipped Classroom:** Mahasiswa mempelajari materi teori sebelum kelas, praktik di kelas
- **Studi Kasus:** Analisis compiler nyata (GCC, Clang, V8 JavaScript Engine)

## 7. Pengalaman Belajar Mahasiswa

Deskripsi tugas, aktivitas, atau pengalaman belajar yang mendukung pencapaian Sub-CPMK:

- Menganalisis arsitektur compiler sederhana (TinyCC, TCC)
- Mengimplementasikan lexer hand-written untuk subset bahasa C
- Membangun parser recursive descent untuk ekspresi aritmatika
- Menggunakan Flex dan Bison untuk parser generation
- Mengimplementasikan symbol table dengan nested scopes
- Melakukan type checking dan semantic analysis
- Menghasilkan three-address code dari AST
- Memetakan intermediate code ke assembly target
- Mengimplementasikan optimasi compiler dasar
- Mengeksplorasi LLVM infrastructure dan JIT compilation
- Berkolaborasi dalam tim untuk mengembangkan compiler lengkap
- Mempresentasikan hasil proyek compiler
- Melakukan benchmarking dan analisis performa

## 8. Kriteria, Indikator, dan Bobot Penilaian

Teknik/alat asesmen dipetakan ke Sub-CPMK/CPMK dengan bobot yang jelas:

Komponen	Teknik Asesmen	Indikator/CPMK	Bobot (%)
Kuis	Multiple Choice & Essay	Sub-CPMK 1.1, 1.2, 1.3	10
Tugas Praktikum 1	Lexer Implementation	Sub-CPMK 2.1, 2.2, 2.4	8
Tugas Praktikum 2	Parser Implementation	Sub-CPMK 2.3, 2.4	10
Tugas Praktikum 3	Symbol Table	Sub-CPMK 3.1, 3.2, 3.3	8
Tugas Praktikum 4	Semantic Analysis + TAC	Sub-CPMK 3.3, 4.1, 4.2	8
Tugas Praktikum 5	Code Generation	Sub-CPMK 5.1, 5.2, 5.3	6
UTS	Written Exam & Coding	CPMK-1, CPMK-2	20
Project Final	Complete Compiler	CPMK-2, CPMK-3, CPMK-4, CPMK-5	25
UAS	Comprehensive Exam	CPMK-1, CPMK-2, CPMK-3, CPMK-4, CPMK-5, CPMK-6	15
<b>Total</b>			<b>100</b>

### Kriteria Penilaian:

- A (85-100): Menguasai semua CPMK dengan sangat baik, mampu menerapkan dalam kasus kompleks
- B (70-84): Menguasai sebagian besar CPMK dengan baik
- C (60-69): Menguasai CPMK dasar dengan cukup
- D (50-59): Menguasai sebagian kecil CPMK
- E (<50): Belum menguasai CPMK yang ditetapkan

### Rubrik Penilaian Tugas Praktikum:

- **Correctness (40%)**: Program berjalan dengan benar, mengatasi edge cases
- **Code Quality (30%)**: Struktur kode, readability, dokumentasi
- **Completeness (20%)**: Semua requirement terpenuhi
- **Testing (10%)**: Test cases yang komprehensif

## 9. Evaluasi dan Refleksi Pembelajaran

Penilaian sumatif/formatif untuk memantau ketercapaian outcome secara menyeluruh:

- **Evaluasi Formatif:** Kuis mingguan, progress review proyek compiler, peer review untuk memberikan feedback berkelanjutan
- **Evaluasi Sumatif:** UTs dan UAs untuk mengukur pencapaian CPMK secara komprehensif
- **Refleksi Mahasiswa:** Jurnal belajar mingguan untuk refleksi diri terhadap pemahaman materi kompilator
- **Evaluasi Dosen:** Survey kepuasan mahasiswa di tengah dan akhir semester
- **Continuous Improvement:** Analisis hasil penilaian untuk perbaikan RPS di semester berikutnya

## 10. Daftar Referensi

Sumber belajar utama yang digunakan dalam penyusunan materi dan asesmen:

1. Aho, A. V., Lam, M. S., Sethi, R., & Ullman, J. D. (2006). *Compilers: Principles, Techniques, and Tools* (2nd ed.). Pearson Education. (Dragon Book)
2. Cooper, K. D., & Torczon, L. (2024). *Engineering a Compiler* (3rd ed.). Morgan Kaufmann. (TAA Textbook Excellence Award Winner 2024)
3. Grune, D., van Reeuwijk, K., Bal, H. E., Jacobs, C. J. H., & Langendoen, K. (2012). *Modern Compiler Design* (2nd ed.). Springer.
4. Levine, J. R. (2009). *flex & bison: Text Processing Tools*. O'Reilly Media.
5. Appel, A. W., & Palsberg, J. (2022). *Modern Compiler Implementation in Java* (2nd ed.). Cambridge University Press.
6. LLVM Project. (2024). *LLVM Language Reference Manual*. Retrieved from <https://llvm.org/docs/LangRef.html>
7. Thain, D. (2024). *Introduction to Compilers and Language Design*. Retrieved from <https://dthain.github.io/books/compiler/>
8. ANTLR Project. (2024). *ANTLR 4 Documentation*. Retrieved from <https://www.antlr.org/>
9. UC San Diego CSE 231: Compiler Construction. (2024). Retrieved from <https://ucsd-cse231.github.io/sp24/>
10. Stanford University CS143: Compilers. (2024). Retrieved from <https://su.stanford.edu/~cse143/>

Disusun oleh,

**Tim Dosen Teknik Informatika**  
Program Studi S1 Teknik Informatika  
Fakultas Teknik

5 Februari 2026