Bab 18

Lampiran dan Rubrik Penilaian

Sub-CPMK yang Dicakup dalam Bab Ini:

- **Sub-CPMK 1.1-6.2:** Menyediakan instrumen penilaian dan panduan praktis untuk seluruh capaian pembelajaran

18.1 Koleksi Quiz dan Uji Kompetensi

Uji pemahaman cepat untuk setiap topik utama:

18.1.1 Quiz 1: Arsitektur Kompilator

1. Apa perbedaan utama antara *front-end* dan *back-end* kompilator?
2. Mengapa kita memerlukan *Intermediate Representation*?

18.1.2 Quiz 2: Optimasi dan Code Gen

1. Apa resiko melakukan optimasi yang terlalu agresif?
2. Jelaskan istilah *Register Spilling*!

18.2 Template Dokumentasi Kode

18.2.1 Template Laporan Praktikum

```
NAMA MAHASISWA: [Nama Lengkap]
NIM: [Nomor Induk Mahasiswa]
KELAS: [Kelas]
MATA KULIAH: Teknik Kompilasi
TUGAS: [Judul Tugas]
TANGGAL: [Tanggal]

1. TUJUAN
   [Jelaskan tujuan praktikum]

2. TEORI DASAR
   [Jelaskan teori yang relevan]

3. IMPLEMENTASI
   [Jelaskan implementasi yang dilakukan]

4. HASIL DAN ANALISIS
   [Tampilkan hasil dan analisis]

5. KESULITAN DAN SOLUSI
   [Jelaskan kesulitan yang dihadapi dan solusinya]

6. KESIMPULAN
   [Buat kesimpulan dari praktikum]

7. LAMPIRAN
   [Lampirkan kode sumber lengkap]
```

18.2.2 Template Dokumentasi Kode

```c
/**
 * @file [nama_file].c
 * @brief [deskripsi singkat file]
 * @author [nama author]
 * @date [tanggal]
 *
 * [deskripsi detail file]
 */

/**
 * @brief [deskripsi fungsi]
 * @param [param1] [deskripsi parameter 1]
 * @param [param2] [deskripsi parameter 2]
 * @return [deskripsi return value]
 *
 * [detail implementasi fungsi]
 */
int function_name(int param1, char *param2) {
    // Implementation here
}
```

18.3 Checklist Kualitas Kode

18.3.1 Checklist untuk Lexer

- ☐ Regular expressions terdefinisi dengan benar
- ☐ Token recognition akurat
- ☐ Error handling untuk invalid input
- ☐ Memory management yang benar
- ☐ Kode style konsisten
- ☐ Dokumentasi lengkap
- ☐ Testing coverage memadai

18.3.2 Checklist untuk Parser

- ☐ Grammar definition benar
- ☐ Parsing algorithm efisien
- ☐ Syntax error reporting jelas
- ☐ AST construction benar
- ☐ Error recovery mechanism
- ☐ Testing dengan valid dan invalid input

18.3.3 Checklist untuk Code Generator

- ☐ Instruction selection optimal
- ☐ Register allocation efisien
- ☐ Target code generation benar
- ☐ Optimization terimplementasi
- ☐ Performance testing dilakukan
- ☐ Cross-platform compatibility

18.4 Format Submisi Tugas

18.4.1 Struktur Folder Proyek

```
[NIM]_[Nama]_[Tugas]/
|-- src/
|   |-- lexer.c
|   |-- parser.c
|   |-- ast.c
|   |-- symtab.c
|   |-- main.c
|   `-- util.h
|-- include/
|   |-- ast.h
|   `-- symtab.h
|-- tests/
|   |-- test1.c
|   `-- test2.c
|-- docs/
|   |-- README.md
|   `-- laporan.pdf
|-- Makefile
`-- README.md
```

18.4.2 Format Penamaan File

- Source code: `[nim]_[nama]_[komponen].c`
- Header files: `[nim]_[nama]_[komponen].h`
- Test files: `test_[komponen].c`
- Documentation: `laporan_[nim]_[nama].pdf`
- Executable: `[nim]_[nama]_[proyek]`

18.5 Best Practices Pengembangan Compiler

18.5.1 Coding Standards

- Gunakan consistent naming conventions
- Comment code yang kompleks
- Modular design dengan single responsibility
- Error handling yang robust
- Memory management yang safe

- Testing yang komprehensif

18.5.2 Version Control

```
# Git workflow untuk proyek compiler
git init
git add .
git commit -m "Initial commit: lexer implementation"
git branch feature-parser
git checkout feature-parser
# Implement parser
git add .
git commit -m "Add parser with LL(1) algorithm"
git checkout main
git merge feature-parser
git tag v1.0.0
```

18.5.3 Testing Strategies

- Unit testing untuk setiap komponen
- Integration testing untuk komponen interaksi
- End-to-end testing untuk complete compiler
- Performance testing untuk optimization validation
- Regression testing untuk maintenance

18.6 Resources Tambahan

18.6.1 Online Resources

- **Compiler Design**: https://www.cs.cornell.edu/courses/cs4120/2018fa/
- **LLVM Tutorial**: https://llvm.org/docs/tutorial/
- **Flex/Bison Manual**: https://www.gnu.org/software/flex/manual/
- **ANTLR**: https://www.antlr.org/

18.6.2 Recommended Books

- **Compilers: Principles, Techniques, and Tools** - Aho, Lam, Sethi, Ullman
- **Modern Compiler Implementation** - Andrew Appel

- **Engineering a Compiler** - Cooper and Torczon
- **Programming Language Pragmatics** - Michael Scott

18.6.3 Useful Tools

- **IDE**: VS Code, CLion, Eclipse CDT
- **Debugging**: GDB, Valgrind, AddressSanitizer
- **Profiling**: Perf, gprof, Intel VTune
- **Documentation**: Doxygen, Sphinx

Aktivitas Pembelajaran

1. **Rubrik Review**: Pelajari kriteria penilaian untuk setiap jenis tugas.
2. **Template Usage**: Gunakan template dokumentasi untuk laporan praktikum.
3. **Code Quality Check**: Lakukan self-audit menggunakan checklist kualitas kode.
4. **Workflow Implementation**: Terapkan best practices dalam pengembangan tugas.
5. **Resource Exploration**: Telusuri referensi tambahan untuk pendalaman materi.

Latihan dan Refleksi

1. Buat draf laporan praktikum menggunakan template yang disediakan!
2. Lakukan review kode menggunakan checklist kualitas untuk Lexer!
3. Simulasikan Git workflow untuk proyek kecil!
4. Susun struktur folder proyek sesuai standar submisi!
5. Analisis kriteria penilaian untuk mendapatkan nilai maksimal (Grade A)!
6. **Refleksi**: Bagaimana rubrik dan standar dokumentasi membantu efektivitas belajar Anda?

Asesmen (Evaluasi Kinerja)

Instrumen Evaluasi Kualitas Dokumentasi dan Praktik

A. Audit Dokumentasi

1. Kesesuaian dengan template laporan
2. Kejelasan penjelasan implementasi
3. Kualitas dokumentasi dalam kode (comments/doxygen)

B. Audit Standar Kode

1. Kepatuhan terhadap coding standards
2. Penggunaan version control (commit messages/history)
3. Struktur folder dan penamaan file

Rubrik Penilaian: Lihat Section 18.1

Checklist Pencapaian Kompetensi

Centang item berikut setelah Anda yakin telah menguasainya:

☐ Saya memahami kriteria penilaian untuk setiap komponen evaluasi

☐ Saya mahir menggunakan template dokumentasi yang disediakan

☐ Saya dapat menerapkan standar kualitas kode dalam pengembangan compiler

☐ Saya memahami format submisi tugas yang benar

☐ Saya dapat menerapkan best practices dalam version control dan testing

☐ Saya dapat memanfaatkan resources tambahan untuk belajar mandiri

Rangkuman

Bab ini menyediakan berbagai instrumen pendukung pembelajaran, mulai dari rubrik penilaian, template dokumentasi, checklist kualitas, hingga best practices dan referensi tambahan. Mahasiswa menggunakannya sebagai panduan standar dalam menyelesaikan tugas dan proyek.

Poin Kunci:

- Rubrik memberikan transparansi dalam penilaian kompetensi

- Template dan checklist memastikan konsistensi dan kualitas output
- Standar submisi mempermudah integrasi dan manajemen proyek
- Best practices meningkatkan profesionalisme dalam pengembangan software
- Resources tambahan membuka peluang eksplorasi lebih dalam

Kata Kunci: ***Rubrik Penilaian***, ***Template Dokumentasi***, ***Checklist Kualitas***, ***Best Practices***, ***Standard Submisi***, ***Learning Resources***