CSE 420 Syllabus, Schedule, and Protocols

(Fall 2025 Version)

Contents

Reference Textbook: Compilers Principles, Techniques, and Tools (Second Edition) by Aho, Lam, Sethi, and Ullman

Rules and Guidelines

1.a Use of the Reference Textbook

Note that the reference textbook by Aho and Ullman is considered the best book to teach compiler design all over the world and is currently in use in numerous universities. In prominent universities, sometimes there are more than one reference books, however, this book is always present and bulk of the materials are always picked from it. The reason is that the book is among the oldest and most detailed compiler related books one can find. Students will definitely find a good explanation and detail of any materials discussed in the class in this book. Consequently, students must avoid any assumption that they do not have adequate materials to study the concepts and approaches discussed in the classroom or to get clarification about implementation hints regarding lab assignments. **Everything is available in the book and studying the book is essential for doing well in the course**.

However, given the book is very detailed and rigorous, a mapping between the lecture content and sections/subsections of the book is provided for each lecture so that students do not have to search relevant items. In addition, some simplifying alternative reading materials are given for a few lectures for which the book content is scattered in too many places.

Students are recommended to come to the consultation hours of the theory/lab instructors if they find difficulty understanding any specific content of the reference textbook. However, they must never assume that some videos or slides can replace the need of the reference textbook.

1.b Self-study Policy

The compiler design course is designed to give a comprehensive experience of understanding the theory and practice of compiler constructions through the theory classes and lab assignments. To keep an alignment between the lab and theory in a short trimester system under such an objective, details of some concepts are skipped in theory classes. However, to have a holistic knowledge of a programming language compiler construction, students should know those details. The policy we adopt to instill the knowledge of detail is by assigning some **self-study** theory assignments. Understand that, these are self-study topics. That means, students must study some sections/subsections of the book **on their own** and solve the assignment problems. Students are allowed to do group studies, and are welcome to take help about theory assignments during the consultation hours. However, they must never plagiarize their assignment submissions. Any plagiarized submission will get a straight zero.

1.c Medical Emergency Reporting Guidelines

A student can miss a theory quiz or assignment deadline due to medical emergencies. Then the section instructor can decide to accept a late submission of assignment or arrange a makeup quiz for the student. However, this is solely under the discretion of the section instructor whether to provide a student with such options. Students must understand that sickness and travails are part of life and one needs to make progress in study regardless of their presence. In addition, note that medical reports for sickness will be accepted for the student him/herself only. Sickness of family members or relatives are not an acceptable excuse for missing any quiz or assignment deadline. Furthermore, a student must report about his/her sickness as soon as possible after recovering from the illness. A medical report shared much later will be ignored.

1.d Theory Assignment Late Submission Policy

Note that since lab assignments are done in pairs, by-weekly, and incremental; late submission of lab assignments are not accepted in any circumstances. Regarding the late submission of theory assignments, note that if the assignment is to be submitted using some online platform (Google forms, Google classroom, or Turnitin) students are responsible for submitting the assignments early enough so that any internet connectivity issue or server system glitch does not affect them. That is, they should not wait till the last 10 minutes of the deadline and run the risk of failing to submit their work even after completing it. Students must understand that instructors do not have any control over Google's or Turnitin's server. So that they cannot fix any system issue for them. Furthermore, each theory assignment is given two weeks to complete even though none requires more than 4 hours of work only to ensure that students have enough free time to work on them and to submit.

Theory Lecture Schedule

Lecture Schedules for the Midterm Exam

Lecture 1: Introduction to Compiler

The Structure and stages of the compiler
The analysis-synthesis model of compilation
Difference between a compiler and an interpreter

Reading References:

1. Compiler vs. Interpreter: Know The Difference And When To Use Each Of Them
2. Chapter 1: Sections 1.2 to 1.5 (Inclusive) of the reference textbook
3. Architecture of a Compiler and its Behavior by Dr. Muhammad Nur Yanhaona

Lecture 2: Introduction to Lexical Analysis

The role of a Lexical Analyzer
Tokens, Patterns, Token Attributes
Regular Definitions
The Structure of a Generated Lexical-Analyzer

Reading References:

1. Regular Expressions (Regex)
2. Regular Expressions
3. Chapter 3 Sections 3.1 of the reference textbook
4. Chapter 3 Sections 3.3 of the reference textbook
5. Introduction to Lexical Analysis by Dr. Muhammad Nur Yanhaona

Self-Study Assignment 1: students have to study the direct method of converting regular expression to DFA from Subsections 3.9.1 to 3.9.5 of Chapter 3 of the reference textbook and show the process of DFA construction for a reference regular expression. (Time 2 weeks)

Lecture 3: Introduction to Syntax Analysis

The role of a parser
Context Free Grammars
Parse Trees and Derivations
Grammar Ambiguity and Mitigation Techniques
Verifying the Language Generated by a grammar

Reading References:

1. Chapter 4 Section 4.1 of the reference textbook
2. Chapter 4 Subsection 4.2.1 to 4.2.5 of the reference textbook
3. Introduction to Syntax Analysis by Dr. Muhammad Nur Yanhaona

Lecture 4: Introduction to Bottom-up Parsing

Concept of Reductions
Stack simulation example of Bottom-up Parsing
Concept of Shift-reduce Parsing
Handles and Handle Pruning

Reading References:

1. Chapter 4 Section 4.5 of the reference textbook

Lecture 5: Introduction to Simple LR Parsing

LR(0) Items
Closure of a LR(0) Item Set
LR(0) Automaton Construction

Reading References:

1. Chapter 4 subsection 4.6.1 and 4.6.2 of the reference textbook
2. Introduction To SImple LR Parsing Example

Lecture 6: Construction of SLR Parsing Table

FIRST and FOLLOW computation
Generation of SLR Parsing Table from LR(0) automaton and FOLLOW set of non-terminals

Reading References:

1. Chapter 4 Subsection 4.4.2 of the reference textbook
2. Chapter 4 Subsection 4.6.4 of the reference textbook
3. Chapter 4 Subsection 4.6.3 (Only the structure of LR parsing table) of the reference textbook
4. First and Follow Computation Example

Lecture 7: SLR Parsing Algorithm

Behavior of a SLR Parser
Example of SLR Parsing
Shift-reduce/Reduce-reduce Conflicts and Other Errors in Bottom-up Parsing

Reading References:

1. Chapter 4 Subsection 4.6.3 of the reference textbook
2. Chapter 4 Subsection 4.8.3 of the reference textbook
3. Moves of an SLR(1) parser on id * id(Step by step)
4. Example of SLR(1) Parsing with Epsilon Rules

Self-Study Assignment 2: students have to study the canonical LR(1), that is, CLR(1), parser construction from Section 4.7.1 to 4.7.3 of the reference textbook and show the process of CLR(1) parser construction from a small reference grammar. (Time 2 weeks)

Lecture 8: Symbol Table

Importance of Symbol Tables
Scopes in Program and Nesting of Symbol Tables
Important attributes of variables and functions

Reading References:

1. Symbol table by Professor Rich Maclin of the University of Minnesota Duluth
2. Symbol Tables and Static Checks
3. Symbol Table Implementation example

Lecture 9: Introduction to Syntax-directed Translation

The logic of syntax directed translation
Inherited and Synthesized Attributes
Evaluation of syntax directed definitions

Reading References:

1. From the start of the Chapter to Example 5.2 in Page 307 of the reference textbook
2. Chapter 5 Section 5.4 of the reference textbook
3. Chapter 5 Example 5.11 in Page 318 of the reference textbook
4. Semantic Analysis and Syntax Directed Definitions by Dr. Muhammad Nur Yanhaona

Lecture 10: Construction of the Parse Tree using SDD Rules

Importance of a Syntax Tree
SDT/SDT for Syntax Trees
Construction Example of a Syntax Tree

Reading References:

1. Same as Lecture 9

One class equivalent time is reserved for two quizzes before the Midterm exam. If 12 classes can be taken before the midterm, then the other class should be for reviewing so far materials.

Lecture Schedule for the Final Exam

Lecture 11: Type Grammar and Attributes of a Type

Type Expression Grammar
Dependency Graphs
Attributes of an Array Type
S-attributed Definitions and L-attributed Definitions
Grammar for a sequence of variable declarations
Grammar and attributes of Record Types

Reading References:

1. Section 5.2 from Chapter 5 of the reference textbook
2. Section 5.3.2 of Chapter 5 of the reference textbook
3. Section 6.3 of Chapter 6 of the reference textbook

Lecture 12: Type Information Processing During Bottom-up Parsing

SDT for Array Type Calculation
SDT for variable widths and offsets calculation in a sequence of variable declarations

Reading References:

1. Same as Lecture 11

Lecture 13: Introduction to Intermediate Code Generation

Importance of three address codes as Intermediate Representation
Features of three address codes
Quadruples
Triples

Reading References:

1. Chapter 6 from Introduction up to Section 6.2 (Inclusive) of the reference textbook
2. Three-address code and its Quadruple,Triples and Indirect Triples Representation example

Lecture 14: Array Access Logic and SDT for Array Accesses

Addressing an array element
Translation of Array References

Reading References:

1. Chapter 6 Section 6.4 of the reference textbook
2. SDT for Array References

Lecture 15-16: Handling Flow-of-Control Statements

Grammar for branching and loops
SDT for translating if-else and while loops

Reading References:

1. Chapter 6 Section 6.6 of the reference textbook

Self-Study Assignment 3: students have to study the intermediate code generation for procedure from Section 6.9 of the reference textbook and generate the three-address code for a code block containing loops and nested procedure calls. (Time 2 weeks)

Lecture 17-18: **Handling Object Oriented Language Features during Compilation**

Handling class object's field access
Translation of class methods
The logic of dynamic dispatch

Reading References:

1. Handling Object Oriented Language Features during Compilation By Dr. Muhammad Nur Yanhaona

Lecture 19: Runtime Storage Organization: Stack

Different parts of a process memory
Memory Layout
The content of stack

Reading References:

1. From the start of the 7.2 to subsection 7.2.3 of the reference textbook

Lecture 20: Runtime Storage Organization: Heap

Locality in Programs and Optimization techniques using the Memory Hierarchy
Reducing Fragmentation with Best Fit and Next Fit Object Placement
Managing and combining (Coalescing) Free space

Reading References:

1. Chapter 7 Section 7.4.3 of the reference textbook
2. Chapter 7 Section 7.4.4 of the reference textbook

Two class equivalent time is reserved for the two quizzes before the Final exam and for reviewing materials discussed after the midterm.

Scoring Rubric

The following is the default scoring rubric for grading students in this course. If some of the assessments cannot be taken or need to be replaced by alternative assessments for some unforeseen circumstances in any semester then students will be notified about the updated scoring rubric. Hence, unless any specific alternative instructions are provided, students should assume that this rubric will be followed.

Assessment Type	Percentage Score	Comments
Quizzes	15%	n-1 quizzes will be counted out of n quizzes taken. Ideally, n is 4 or more.
Assignments	10%	All assignments count. Ideally there will be 3 assignments.
Lab Work	25%	Details of lab work assessments are given in the lab related document.
Midterm Exam	20%	
Final Exam	30%	

Midterm and Final Exam Syllabus

Note that the expectation is that we will cover 10 lectures before the midterm exam and 10 more after the midterm exam. However, our observation is that we miss classes for many external reasons every semester and could not reach our target. So, the general principle is that the midterm exam syllabus includes every lecture covered before the midterm exam and final exam syllabus covers every lecture covered from the beginning of the semester till the end of classes before the final exam. So, you understand that the final exam syllabus is comprehensive. However, the general policy we follow is that in the final exam, we only give short and mostly theoretical questions on any topic that is covered before the midterm exam. You should be able to answer them if you followed the classes diligently and solved the lab assignments on your own. Long problem-solving questions of the final exam questionnaire will come from topics covered after the midterm exam.

You are accustomed to sudden declarations from the registrar or chairperson's office that no new topics should be covered in the last week before midterm/final exam due to various reasons. Note that in the compiler design course, **we will ignore such instructions**. It is essential as we want to give you a complete picture of compiler constructions to the best of our ability in the short time-span we have every semester. Thus, we cannot afford to ignore any lecture topic. Furthermore, this course is for senior year students who should already have acquired the necessary skills for time management so that they can review everything new covered in the last weeks before midterm/final exams.

CO Description

Sl	CO Description	Weight: 95%	POs	Bloom's Taxonomy Domain/ Level	KPIs of relevant PO	Knowledge Profile (K1-K8)	Complex Engineering Problems (P1-P7)	Is this CO Assessed ?
C01	Review the fundamentals concepts of computer language processing system, different phases of a compiler, and analysis-synthesis model of compilation	5	a	Cognitive/ Understand		K3,K4	P1	Yes
C02	Design regular expressions for recognizing tokens, building symbol table, and transforming the regular expressions to finite state machines for developing lexical analyzer.	20	a	Cognitive/ Apply		K5,K6	P1,P3,P4	Yes
C03	Design context-free-grammars and parsers for syntax analysis to validate string of tokens of a programming language.	30	c	Cognitive/ Create		K6	P1, P3, P5, P6	Yes
C04	Construct syntax-directed definitions, translation schemes, annotated parse trees, dependency graph for semantic analysis and type checking.	25	e	Cognitive/ Apply		K5,K6,K8	P1, P2, P4, P5	Yes
C05	Illustrate intermediate codes, and code optimization techniques for efficient compiler design and how to generate code for programming languages considering different cost	20	j	Cognitive/ Apply		K5,K6	P1, P2, P3	Yes