The Hong Kong University of Science and Technology

UG Course Syllabus (Spring 2025-26)

[Course Title] Modern Compiler Construction

[Course Code] COMP4121

[No. of Credits] 3-credit

[Any pre-/co-requisites] Pre-requisite: COMP 3021 OR COMP 3031

Name: Dr. Lionel Parreaux

Email: parreaux@cse.ust.hk

Course Description

Compiler implementation techniques are relevant to many areas of software engineering. From parsing ad-hoc configuration file formats to validating complex specification languages to generating efficient code solving data-intensive problems at scale, many important problems of today and tomorrow require knowledge of basic compiler technology. This course exposes students to the essentials of modern compiler construction, including parsing, semantic analysis, program transformation, and code generation. Students will learn to design and implement their own programming language and extend it with a feature of their choice in a small team project. The course focuses on achieving these goals through effective high-level programming techniques, whose mastery will also make students better programmers in general.

Tentative course structure:

1. Overview, source languages and run-time models
2. Review of formal languages
3. Lexical analysis
4. Syntactic analysis (parsing)
5. Name analysis
6. Type checking
7. Type inference
8. Code generation 1
9. Code generation 2
10. Optimization

Intended Learning Outcomes (ILOs)

By the end of this course, students should be able to:

1. Design and implement computer languages, in particular simple programming languages.

2. Construct and extend compilers.

3. Implement complex language specifications.

4. Coordinate software development with a project partner.

5. Deliver working software artifacts written in a high-level programming language.

6. Master basic formal techniques for reasoning about program semantics.

Assessment and Grading

This course will be assessed using criterion-referencing and grades will not be assigned using a curve. Detailed rubrics for each assignment are provided below, outlining the criteria used for evaluation.

Assessment Task	Contribution to Overall Course grade (%)
Projects	60% (five minor projects worth 5% each, and one major project worth 35%)
Exam	40%
Total	100%

Assessments:

[List specific assessed tasks, exams, quizzes, their weightage, and due dates; perhaps, add a summary table as below, to precede the details for each assessment.]

Assessment Task	Contribution to Overall Course grade (%)	Due date
Projects	60%	Project 1: 11 Mar 2026 Project 2: 25 Mar 2026 Project 3: 15 Apr 2926 Project 4: 25 Apr 2026 Project 5: 16 May 2026 Project 6 (optional): 29 May 2026
Final exam	40%	23 May 2026

Assessment marks for individual assessed tasks will be released within two weeks of the due date.

Mapping of Course ILOs to Assessment Tasks

Assessed Task	Mapped ILOs	Explanation
Projects	ILO1, ILO3, ILO6	These are five mini-projects worth 12% each, in the form of regular programming assignments.
Final exam	ILO2, ILO4, ILO5	

Grading Rubrics

Detailed rubrics for each assignment will be provided. These rubrics clearly outline the criteria used for evaluation. Students can refer to these rubrics to understand how their work will be assessed.

Final Grade Descriptors:

Grades	Short Description	Elaboration on subject grading description
A	Excellent Performance	Demonstrates a comprehensive grasp of subject matter, expertise in problem-solving, and significant creativity in thinking. Exhibits a high capacity for scholarship and collaboration, going beyond core requirements to achieve learning goals.
B	Good Performance	Shows good knowledge and understanding of the main subject matter, competence in problem-solving, and the ability to analyze and evaluate issues. Displays high motivation to learn and the ability to work effectively with others.
C	Satisfactory Performance	Possesses adequate knowledge of core subject matter, competence in dealing with familiar problems, and some capacity for analysis and critical thinking. Shows persistence and effort to achieve broadly defined learning goals.
D	Marginal Pass	Has threshold knowledge of core subject matter, potential to achieve key professional skills, and the ability to make basic judgments. Benefits from the course and has the potential to develop in the discipline.
F	Fail	Demonstrates insufficient understanding of the subject matter and lacks the necessary problem-solving skills. Shows limited ability to think critically or analytically and exhibits minimal effort towards achieving learning goals. Does not meet the threshold requirements for professional practice or development in the discipline.

Course AI Policy

We generally do not allow the use of generative AI tools in the completion of the assignments in this course.

Communication and Feedback

Assessment marks for individual assessed tasks will be communicated via Canvas within two weeks of submission. Feedback on assignments will include the marks received for each item and the reasons for mark deduction. Students who have further questions about the feedback, including marks, should consult the instructor within five working days after the feedback is received.

Resubmission Policy

Except in special circumstances, resubmission for reassessment will not be allowed for any assessment.

Required Texts and Materials

Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman: Compilers: Principles, Techniques, and Tools (2nd Edition, 2006)
Torben Mogensen, Basics of Compiler Design, (2010 edition, http://hjemmesider.diku.dk/~torbenm/Basics/)

Academic Integrity

Students are expected to adhere to the university's academic integrity policy. Students are expected to uphold HKUST's Academic Honor Code and to maintain the highest standards of academic integrity. The University has zero tolerance of academic misconduct. Please refer to Academic Integrity | HKUST – Academic Registry for the University's definition of plagiarism and ways to avoid cheating and plagiarism.

Additional Resources

None