San José State University
Department of Computer Science

# CS/SE 153
# Concepts of Compiler Design

Section 1
Fall 2026

## Course and Contact Information

| | |
|---|---|
| Instructor: | Ron Mak |
| Office location: | Working from home |
| Email: | ron.mak@sjsu.edu |
| Website: | http://www.cs.sjsu.edu/~mak/ |
| Office hours: | One hour after Thursday classes |
| Class days/time: | TuTh: 1:30 – 2:45 PM |
| Classroom: | MH 422 |
| Prerequisites: | CS 47 or CMPE 102, CS 146, and CS 154 (with a grade of "C-" or better in each); Allowed Major: Computer Science, Applied & Computational Math, or Software Engineering; or instructor consent. |

## Course Catalog Description

"Theoretical aspects of compiler design, including parsing context free languages, lexical analysis, translation specification and machine-independent code generation. Programming projects to demonstrate design topics."

## Course Format

This class will meet in person in the classroom. Exams and quizzes will be given in the classroom.

## Faculty Web Page and Canvas

Course materials, syllabus, assignments, grading criteria, exams, and other information will be posted at my faculty website at http://www.cs.sjsu.edu/~mak and on the Canvas Learning Management System course login website at http://sjsu.instructure.com. You are responsible for regularly checking these websites to learn of any updates. You can find Canvas video tutorials and documentations at http://ges.sjsu.edu/canvas-students

## Course Goals

This course will concentrate on practical aspects of programming language translation and engineering a large, complex software application.

You will learn programming language translation in various forms (in this order):

- **Language conversion.** Convert a program written in one high-level language to an equivalent program written in another high-level language.
- **Interpreter with an interactive symbolic debugger.** Execute a program written in a procedural language and be able to set breakpoints, single-step, examine and set variable values, etc.
- **Compiler construction and language design.** Design and build a working compiler for a programming language that you invented. Write sample programs in your language and compile them into assembly language code that you can then assemble and run.

**Software engineering.** Employ the best practices of object-oriented design and team-based software engineering. A compiler is a large, complex program that requires much work and attention to details! Managing the development of such a program requires learning *critical job skills that are highly desired by employers*.

## Course Learning Outcomes (CLO)

Upon successful completion of this course, students will be able to:

CLO 1: Design the **grammar** for a programming language.
CLO 2: Develop a **scanner** and a **parser** for a programming language.
CLO 3: Use the **ANTLR 4** compiler development tools.
CLO 4: Perform **syntactic and semantic analyses** of source programs.
CLO 5: Generate **parse trees** and **symbol tables** for source programs.
CLO 6: Develop a **language converter** that converts a program from one language to another.
CLO 7: Develop an **interpreter** with an interactive **symbolic debugger** that executes a source program in a suitable runtime environment.
CLO 8: Develop a **compiler** that translates a source program into **executable machine code**.
CLO 9: Engineer a large, complex software application.

## Required Text

| | |
|---|---|
| Title: | **The Definitive ANTLR 4 Reference, 2nd edition** |
| Author: | Terence Parr |
| Publisher: | Pragmatic Bookshelf, 2013 |
| ISBN: | 978-1934356999 |
| | http://www.antlr.org |

## Online Pascal Tutorials

We will use Pascal as the example source language.

| |
|---|
| Pascal Tutorial looks very good. It even has an online compiler. |
| Learn Pascal also looks good, although it doesn't appear to cover set types. |

## Online Pascal Compilers

| |
|---|
| http://rextester.com/l/pascal_online_compiler |
| https://www.tutorialspoint.com/compile_pascal_online.php |
| https://www.jdoodle.com/execute-pascal-online |

## Useful References

| |
|---|
| Pascal User Manual and Report, 4th ed. (ISO Pascal Standard) |
| The Java Virtual Machine Instruction Set |

## Software to Install

You should install and use an interactive development environment (IDE) such as Eclipse or IntelliJ. To develop a compiler for your language, you will need to download and install the ANTLR 4 package and its Eclipse or IntelliJ plugin, and then modify them to generate the compiler components in Java.

## Project teams

You will form small project teams. *Team membership is mandatory for this class.* The teams will last throughout the semester. Once the teams are formed, you will not be allowed to move from one team to another, so form your teams wisely!

## Course Requirements and Assignments

You must have good Java programming skills and be familiar with development tools such as Eclipse or IntelliJ.

Team-based **lab assignments** will provide practice with compiler design techniques and give you experience adding new features to a large legacy code base. *Each student on a team will receive the same score for each team assignment.*

Each team will submit its assignments into Canvas, where the rubric for scoring each will be displayed. Late assignments will lose 20 points and an additional 20 points for each 24 hours after the due date.

***This is a challenging course that will demand much of your time and effort throughout the semester.***

The university's syllabus policies:

- University Syllabus Policy S16-9 at http://www.sjsu.edu/senate/docs/S16-9.pdf.
- Office of Graduate and Undergraduate Program's Syllabus Information web page at http://www.sjsu.edu/gup/syllabusinfo/

"Success in this course is based on the expectation that students will spend, for each unit of credit, a minimum of 45 hours over the length of the course (normally three hours per unit per

week) for instruction, preparation/studying, or course related activities, including but not limited to internships, labs, and clinical practica. Other course structures will have equivalent workload expectations as described in the syllabus."

## Exams

The quizzes and exams will test understanding (not memorization) of the material taught during the lectures and now well each of you participated in your team assignments and project. Quizzes and exams will use the lockdown browser.

There can be no make-up quizzes and midterm examination unless there is a documented medical emergency. Make-up final examinations are available only under conditions dictated by university regulations.

## Team Compiler Project

In addition to the team-based assignments, each project team will develop a compiler project during the semester. Each team will develop a working compiler for a newly invented language or for an existing language. Teams will be able to write, compile, and execute programs written in their invented or chosen languages. *Each student on a team will receive the same score for the team project.* Each project involves:

- Invent a new programming language or choose a subset of an existing language.
- Develop a grammar for the language.
- Generating a compiler for the language using the ANTLR 4 compiler-compiler tools. Other components may be borrowed from the compiler code given in the class.

A **minimally acceptable compiler** project has at least these features:

- Two data types with type checking.
- Basic arithmetic operations with operator precedence.
- Assignment statement.
- A conditional control statement (e.g., IF).
- A looping control statement.
- Procedures or functions with calls and returns.
- Parameters passed by value or by reference.
- Basic error recovery (skip to semicolon or end of line).
- Nontrivial sample programs written in the source language.
- Generate Jasmin assembly code that can be successfully assembled.
- Execute the resulting .class file.
- No crashes (e.g., null pointer exceptions).

**Note:** Taking an existing compiler and simply replacing the source language's keywords is not an acceptable project.

Each team will give an oral presentation at the end of the semester that includes a demo of its compiler. The rest of the class (along with the instructor) will score each presentation based on a given set of criteria. *Each student on a team will receive the same score for the project.*

Each team will write a report (5-10 pp.) that includes:

- The main features of your source language.
- The ANTLR 4 grammar for your source language with optional syntax diagrams.

- A high-level design description of the intermediate and backend tiers of the compiler.
- Code templates that show the Jasmin assembly code your compiler generates for some key constructs of the source language.

## Grading Information

Individual total scores will be computed with these weights:

| | |
|---|---|
| 30% | Assignments* |
| 30% | Compiler project* |
| 20% | Quizzes and midterm exam** |
| 20% | Final exam** |

* *team scores*
** *individual scores*

Course grades will be based on a curve. The median total score will earn a B–.

## Postmortem Report

At the end of the semester, each student must also turn in a short (1 or 2 pages) **individual postmortem report** that includes:

- A brief description of what you learned in the course.
- An assessment of your accomplishments for your project team on the assignments and the compiler project.
- An assessment of each of your other project team members.

Only the instructor will see these reports.

## Technology Requirements

Students are required to have an electronic device (laptop, desktop, or tablet) with a camera and microphone. SJSU has a free equipment loan program available for students: https://www.sjsu.edu/it/services/classroom-tech/equipment-loaning/

Students are responsible for ensuring that they have access to reliable Wi-Fi during tests. If students are unable to have reliable Wi-Fi, they must inform the instructor, as soon as possible or at the latest one week before the test date to determine an alternative. See WiFi Network Access website for current Wi-Fi options on campus. https://www.sjsu.edu/it/services/network/wifi-access.php

## University Policies

Per University Policy S16-9, university-wide policy information relevant to all courses, such as academic integrity, accommodations, etc. will be available on Office of Graduate and Undergraduate Program's Syllabus Information web page at http://www.sjsu.edu/gup/syllabusinfo/.

# CS/SE 153
# Concepts of Compiler Design

Section 1
Fall 2026
Instructor: Ron Mak

**Course Schedule** (subject to change with fair notice)

| Week | Dates | Topics |
|---|---|---|
| 1 | Aug 20 | Overview of the course<br>Types of programming languages<br>It's all about translation!<br>The Pascal programming language<br>Translation examples<br>*Submit proof of prerequisites*<br>*Form programming teams* |
| 2 | Aug 25<br>Aug 27 | Programming language grammar<br>Syntax diagrams<br>Syntax and semantics<br>ANTLR 4 grammar file<br>Tokens and production rules<br>Basic scanning algorithm |
| 3 | Sept 1<br>Sept 3 | A simple DFA scanner<br>A software architecture for translators<br>Multipass compilers<br>Translator data flow<br>Top-down recursive descent parsing<br>Parse assignment statements and expressions<br>Parse trees |
| 4 | Sept 8<br>Sept 10 | Handling syntax errors<br>Visit parse tree nodes<br>Symbol tables<br>A basic cross-reference listing<br>A simple interactive calculator |
| 5 | Sept 15<br>Sept 17 | **A language converter: Pascal to Java**<br>The ANTLR 4 compiler-compiler<br>Generate a scanner and a parser with ANTLR 4<br>ANTLR 4 parse tree visitor interfaces<br>Walk a parse tree using visitors<br>Convert declarations |
| 6 | Sept 22<br>Sept 24 | Strong typing and type checking<br>Convert statements and expressions<br>Scope and the symbol table stack<br>Parse array and record declarations<br>Convert procedures and functions |

| Week | Dates | Topics |
|---|---|---|
| 7 | Sept 29<br>Oct 1 | **A Pascal interpreter**<br>Runtime memory management for interpreting<br>The runtime stack and stack frames |
| 8 | Oct 6<br>Oct 8 | Execute statements and expressions<br>Execute procedure and function calls<br>A scripting command language<br>An interactive symbolic debugger<br><br>***Midterm exam: Thursday, October 8*** |
| 9 | Oct 13<br>Oct 15 | **A Pascal compiler**<br>The Java Virtual Machine (JVM) architecture<br>Jasmin assembly language<br>Code templates and code generation<br>Code for expressions and statements |
| 10 | Oct 20<br>Oct 22 | Code for procedure and function calls<br>Code to call `printf()`<br>Code for arrays and records |
| 11 | Oct 27<br>Oct 29 | Code to pass parameters by value and by reference<br>A pass-by-reference hack<br>Code optimization and register allocation<br>Compiling object-oriented languages |
| 12 | Nov 3<br>Nov 5 | Runtime libraries<br>Static vs. dynamic scoping<br>Runtime memory management, part 2<br>Garbage collection algorithms |
| 13 | Nov 10<br>Nov 12 | Compilation challenges<br>Stack machine vs. register machine<br>CISC vs. RISC<br>Context-free vs. context-sensitive grammars<br>Shift-reduce parsing<br>Bottom-up parsing with yacc and lex |
| 14 | Nov 17<br>Nov 19 | Syntax-directed translation<br>Attribute grammars<br>LL(1) and LR(0) parsers<br>An integrated development environment (IDE)<br>Case studies |
| 15 | Nov 24 | *Project preparation* |
| 16 | Dec 1<br>Dec 3 | *Project presentations* |
| | Dec 15 | ***Final exam: Tuesday, Dec. 15***<br>Time: 1:00–3:00 PM |