# SI413: Programming Languages and Implementation

Course Policy, Fall 2026 (AY27), v1.0

## Course Website

**External mirror**: https://roche.work/si413

**Intranet mirror**: https://courses.cs.usna.edu/si413

## Instructors


- Prof. Daniel S. Roche, 457 Hopper Hall, x36814, roche@usna.edu (Coordinator).


## Course Description

This course examines basic concepts underlying the design of modern programming languages: types, control structures, abstraction mechanisms, inheritance, concurrency and constructs for programming. The course includes programming assignments in several languages, and is structured around designing, writing specifications for, and building interpreters and compilers for new programming languages.

**Credits**: 2-2-3

**Pre-requisites**: SI342 and (IC312 or SY301)

## Learning Objectives

Upon completing this course, students will be able to:

1. Understand the functional programming paradigm and be able to solve problems in a functional language (supports outcome CS-6).
2. Develop a vocabulary for describing programming languages (supports outcome CS-6).
3. Understand lexical analysis, parsing, and basic interpretation (supports outcome CS-6).
4. Implement a simple interpreter (supports outcome CS-6).
5. Learn a new programming language independently and use it to solve basic tasks (supports outcome CS-6).

## Student Outcomes

Graduates of the program will have an ability to:

1. Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions.

2. Design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline.

3. Communicate effectively in a variety of professional contexts.

4. Recognize professional responsibilities and make informed judgments in computing practice based on legal and ethical principles.

5. Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline.

CS-6. Apply computer science theory and software development fundamentals to produce computing-based solutions.

## Textbook(s)

- Required: Course notes on website.
- Optional: Michael L. Scott. *Programming Language Pragmatics*, Morgan Kaufmann Publishers (any edition).
- Optional: Aho, Lam, Sethi, and Ullman. *Compilers: Principles, Techniques, and Tools*, Pearson (any edition).
- Optional: Abelson, Sussman, and Sussman. *Structure and Interpretation of Computer Programs*, MIT Press. Available online.

## Syllabus

The course is structured around a single task: designing and implementing a new programming language. Each unit is centered on a few additional programming language capabilities and a series of four tasks:

(1) Designing and specifying the syntax and semantics of a novel language to support the required capabilities;
(2) Writing example programs in the novel language (which will later be used for testing);
(3) Implementing a REPL-based interpreter for the language in Java;
(4) Implementing a compiler for the language using LLVM.

To support these concrete tasks within each unit, we will also introduce various new concepts and examples from existing programming languages.

Deadlines for each homework and lab will be posted on the course website above. The overall schedule for the semester is as follows.

- **Unit 1: Just Strings** (3 weeks)
  - Capabilities: literals, input/output, unary and binary operators.
  - Concepts: Specifications, undefined behavior, syntax and semantics, expressions and statements, interpreters and compilers
  - Languages: LLVM IR
- **Unit 2: Variables and Booleans** (3 weeks)
  - Capabilities: Boolean literals, string-boolean and logical operators, named variables
  - Concepts: Scanning and Parsing, symbol table, memory allocation, errors, type safety
  - Languages: Scheme
- **Unit 3: Taking Control** (2 weeks)
  - Capabilities: Conditionals, loops
  - Concepts: Abstract syntax trees, basic blocks, control flow graph, single static assignment, phi nodes
  - Languages: Rust
- **Midterm exam** (Thursday, October 15)
- **Unit 4: Functions** (3 weeks)

  - Capabilities: Function calls, lexical scope
  - Concepts: Heap allocation, anonymous functions, frames and closures
  - Languages: Scheme
- **Unit 5: Typing** (2 weeks)
  - Capabilities: Integer literals and operations, builtin functions, type checking
  - Concepts: Static and dynamic typing
  - Languages: Haskell
- **Unit 6: Optimization** (2 weeks)
  - Concepts: Optimization pipeline, phi nodes, loop unrolling, register allocation
  - Languages: Esoteric languages (e.g. Piet, Befunge, iogii)

## Extra Instruction

Extra instruction (EI) is strongly encouraged and should be scheduled by following the instructions on your instructor's homepage. EI is not a substitute lecture; students should come prepared with specific questions or problems.

## Collaboration

The guidance in the Honor Concept of the Brigade of Midshipmen and the Computer Science Department Honor Policy must be followed at all times. See https://www.usna.edu/CS/resources/honor.php.

All collaboration and outside sources should always be cited. The same rules apply for giving and receiving assistance. If you are unsure whether a certain kind of assistance or collaboration is permitted, you should assume it is not, work individually, and seek clarification from your instructor.

Specific instructions for this course:

- **Homeworks**: Collaboration with other students in the same class is permitted, but must be clearly documented.
- **Labs**: Each student is required to do their own work, and to completely understand the work they turn in; students may be asked to explain in detail how and why their code works after submission.

  Collaboration and assistance from other humans is permitted but should be limited to conceptual discussions and debugging help, never sharing or copying even small parts of working code. Any assistance received must be clearly and specifically documented.
- **Exams**: No collaboration is allowed. Each student may prepare and bring a single study sheet to the midterm and two to the final exam. These sheets must be hand-written (on tablet is OK), and individually prepared. Any group study guides should be shared with the instructor.

## Generative AI

PROVOSTNOTE 1531 defines an AI Usage Framework that courses use to prescribe what kinds of AI usage are allowed for given assignments and/or problems.

In this course, the only allowed AI tool is Google Gemini, logged in with your USNA account, and using the instructor-provided Gem for this course.

- **Homeworks**: AI-Prep and AI-Review permitted with the course Gem. AI use must be acknowledged on the homework being turned in.

- **Labs**: AI-Custom. Use of the course Gem is permitted, including to generate small snippets of code and to help with debugging. A complete transcript of all relevant AI conversations must be turned in along with the assignment. If you simply say "transcript", the Gem will give you a complete transcript in a single code block, ready to copy-paste and turn in.

  Do not ask Gemini to complete your entire assignment, but only for help with specific small parts or understanding of concepts. Generally only small pieces of code (like one or two lines) should be copied directly into your work, and as stated above you will be expected to explain (without using Gemini) how this code works after turning it in.

- **Exams**: AI-None for the actual exam.

  To generate the study sheet, the course Gem may be used as in AI-Prep, AI-Ideate, and AI-Review, but should not be used to generate the actual contents. AI use must be acknowledged on the sheet itself.

## Late Policy

PROVOSTINST 1531.91 defines the baseline late work policy for USNA courses. As authorized by the Department Chair, this course enforces a strict **zero-credit** policy for work submitted past established deadlines.

- **Routine Events:** Athletic travel, watch standing, and competing academic duties do not qualify for extensions; plan ahead and submit early.
- **Valid Exceptions:** Extensions are granted only for official On-Ramp Program (ORP) accommodations or documented acute emergencies (e.g., sudden illness).
- **Notification:** You must notify your instructor *before* the deadline passes, except in emergencies where prior contact is impossible.

## Classroom Conduct

Everyone in the classroom will show appropriate respect to each other at all times.

The section leader is responsible for recording attendance, bringing the class to attention, notifying the CS department office if the instructor is more than 5 minutes late, and directing the class in useful work in the instructor's absence.

Drinks are permitted, but they must be in closable containers. Food, alcohol, and tobacco (of all kinds) are prohibited. The use of laptops, tablets, or other devices during class is at the discretion of the instructor; such use must be related to the class and should never serve as a distraction to other students.

The class will follow the non-attribution of communication practice (Chatham House Rule).

## Grading

Your final grade will be computed as follows:

- 5%: Weekly homeworks (completion-based grading)
- 45%: Labs (language specification and implementation)
- 10%: Midterm exam
- 40%: Final exam

6 and 12 week grades will be computed based on this same breakdown, scaled according to the amount of work completed so far.

For labs we will use a form of *specifications grading*. Each lab will have specific standards which must all be met in order to earn a stated number of points. The grading will be "all or nothing": until the standard is met, there will be **no partial credit**.

However, the opportunity to **revise** (or re-revise) labs with a new deadline, generally within one week, will be given under the following conditions:

- The initial submission (and subsequent revisions) are turned in by the stated deadline.
- Each submission represents *significant progress* and effort beyond the previous version.
- All graded work must be turned in by the last day of classes.

Plus/minus grades will be assigned based on the following numerical cutoffs:

| | + | | - |
|---|---|---|---|
| **A** | | 93-100 | 90-92 |
| **B** | 87-89 | 83-86 | 80-82 |
| **C** | 77-79 | 73-76 | 70-72 |
| **D** | 67-69 | 60-66 | |
| **F** | | 0-59 | |

## Updates to the course policy

In case this course policy needs to be changed during the semester, students will be notified by email and verbally during class. The current version will always be posted on the course website.

## Submitted

Prof. Daniel S. Roche