CS 106 - Introduction to Data Structures

Professor: Sorelle Friedler
sorelle@cs.haverford.edu

http://www.cs.haverford.edu/courses/cmsc106

An introduction to the fundamental data structures of computer science: strings, lists, stacks, queues, trees, BSTs, graphs, sets and their accompanying algorithms. Principles of algorithmic analysis and object reasoning and design will be introduced using mathematical techniques for the notions of both complexity and correctness. More practical issues, such as memory management and hashing, will also be covered. The programming language used to illustrate and implement these concepts will be able to support functional, imperative and object-oriented approaches.

Lab: Includes a weekly required programming lab section. Class sessions will also be taught in a computer lab.
Prerequisites: CS 104 (variants) or CS 105 or BMC 109 or 113 with a grade of 2.0 or better.
Students who have taken CS 107 or BMC CS 151 *shall not* receive credit for this course.
Enrollment Limit: 36. The course will be divided into three lab sessions of at most 12 each.
Lab monitors and TAs: Lab monitors are in KINSC H110 every Sunday - Thursday from 7-11pm to help you with your lab assignments. TAs will also hold office hours (TBD - see the class Google Calendar).
Office hours: see the class Google Calendar for times. Office hours are additionally available by appointment.
Textbook (required): *Data Structures and Algorithms in Java* by Goodrich and Tamassia. The textbook is available online via the Haverford library.

Schedule of Topics

This schedule is *tentative*. Labs are due **by 11:59pm on Tuesday** in the week listed. Students should expect ***at least* 10 hours of work each week.** For the most up-to-date dates and deadlines see the CS 106 Google Calendar.

Week 1. Introduction. Java basics. The importance of documentation and programming style.

- Reading: Chapter 1
- Programming topics: Java syntax, types and type conversion, variable scope, Javadoc comments, very basics of classes.
- In lab: how to check out assignments in GitHub Classroom, work in VS code, and getting started with Lab 0.

Week 2. Basics of object oriented programming (objects and classes and inheritance), variable scope.

- Lab 0 Due: Java basics
- Reading: Chapter 2
- Programming topics: Objects, Classes, inheritance, variable scope, Strings.

Week 3. Object oriented programming basics.

- Reading: Chapters 2, 3.1
- Programming topics: type generics, exceptions.
- Data structures: basics of arrays, ArrayLists, and main args input.
- Ethics: reproducibility, reading and writing documentation.
- Lab 1 Due: Designing data structures - ProPublica 1

Week 4. Linked lists.

- Reading: Chapter 3
- Lab 2 Due: ArrayLists - ProPublica 2

Week 5. Introduction to big-Oh notation

- Reading: Chapter 4

Week 6. Stacks and queues.

- Reading: Chapter 6

Week 7. Review week and midterm.

- Reading: review chapters 1-4 and 6
- Lab 3 Due: Linked lists.
- Wednesday: **Midterm 1 exam** during class time.

Week 8. **Break!**

Week 9. Lists, iterators, interfaces, and graph basics

- Reading: Chapter 7, 8.1, and 14.2
- Topics: graph basics (nodes, edges, weights, neighbors, degree, directed vs. undirected) and implementations (adjacency lists and adjacency matrices)
- Lab 4 Due: Stacks and queues

Week 10. Binary trees

- Reading: Chapter 8

Week 11. Priority queues

- Reading: Chapter 9
- Lab 5 Due: Binary trees

Week 12. Maps and hash tables

- Reading: Chapter 10

Week 13. Sorting and selection

- Reading: Chapter 12

Week 14. Search trees, union-find, and Huffman coding

- Reading: Chapter 11 and 14.7.3
- Lab 6 Due: Priority queues

Week 15. Review week and midterm

- Reading: review chapters 7-12, 14.2, and 14.7.3
- Wednesday: **Midterm 2 exam** during class time.

Final project due at the end of final exam period. Maps, hash tables, sorting and selection (data deduplication).

Labs and Exams

A general outline of the labs is given below. Lab starter code will be distributed to the class.

0. Java basics (1 week)
1. Data structure design (1 week)
2. Array lists (1 week)
3. Linked lists (with timing) (2 weeks)
4. Stacks and queues (1 week)
5. Binary trees (2 weeks)
6. Priority queues (2 weeks)
7. Final project: Maps / hash tables, sorting / selection (data deduplication)

There will be two midterms.

Attendance and Participation

Attendance at, and active participation in, all class and lab sessions is expected of all students. Attendance will serve as a multiplier on your final grade (see below). Students must attend a full class session in order to receive attendance credit for that day; a grace period of 10 minutes will be allowed. Students may miss at most 2 class sessions or lab sessions and still receive full credit for the attendance portion of the grade.

$$\text{Attendance grade} = \frac{\text{number of class or lab sessions attended} + 2}{\text{number of class or lab sessions}}$$

Note that the above grading scheme means that students who miss too many class sessions will fail the course, whether or not they do well on all assignments.

Grading

Grades will be awarded based on the number of points earned and according to the percentage breakdowns shown. Students will not be graded on a curve.

Labs	15%
Final Project	10%
Midterm 1	35%
Midterm 2	35%
Participation	5%

The above total will then be multiplied by the attendance grade to determine the total grade.

In order to receive a merit grade of 2.0 or above in the course students must additionally do *all* of the following:

- receive a merit grade of 2.0 or above on at least one exam,
- hand in non-trivial work on all lab assignments, and
- hand in the final project.

Late work policy

Students have two 24-hour tokens they may use to hand in work late. No other extensions will be provided, and late work will not be accepted without a late token (i.e., will receive zero credit).

Rules and Pet Peeves

- **Be on time.** This includes class, lab, office hours, and appointments.
- **Expect 24 hours before an email response** and read all emails within 24 hours.
- **Attend all classes and labs.**

Collaboration, Plagiarism, and AI use

The only acceptable sources you may use in this course are:
- the course textbook and your class notes,
- the TAs, lab monitors, and professor's help and office hours,
- appropriate discussion with your classmates (see below), and
- the official Java documentation available from Oracle online.

Note that, in addition to many other things not included in the above list, sources that may *not* be used include AI coding assistants or other AI tool as well as websites such as stackoverflow. In this class it is never appropriate to use AI to generate any part of your assignments, whether code, text, or other content. It is also not appropriate to use AI to generate ideas, outlines, study guides, or anything else; do not use it.

Appropriate work with your classmates is described in the departmental collaboration policy here. A summary is below, but students are encouraged to read the full linked document.

Work done in collaboration should never be copied from another student (e.g., from their computer or from joint work on the board). Work from previous semesters should never be shared with current students, or looked at by students in the current semester, though it is fine to share notes you make about lectures or the textbook. Code and other material should never be copied from another student or outside sources unless permission is explicitly given in advance by your professor and the code is cited.

If you are ever in doubt about if your collaboration or use of outside sources is appropriate, please talk to the professor or TAs for clarification.

Learning Accommodations

Haverford College is committed to creating a learning environment that meets the needs of its diverse student body and provides equitable access to students with disabilities. If you have (or think you may have) a disability related to mental health, chronic health, neurological state, and/or physical condition – please contact the Office of Access and Disability Services (ADS) at `hc-ads@haverford.edu`.

Students who have already been approved to receive academic ADA accommodations and want to use these in this course should share their accommodation letter and make arrangements to meet with me as soon as possible to discuss how their accommodations

will be implemented in this course. Please note that accommodations are not retroactive and require advance notice in order to successfully implement.

It is a state law in Pennsylvania that individuals must be given advance notice that they may be recorded. Therefore, any student who has a disability-related need to audio record this class must first be approved for this ADA accommodation by Access and Disability Services and then must communicate approval to me. I will then work with you to provide the accommodation while respecting all students' right to privacy.