CSC 207
Algorithms and Object-Oriented Design

Object Orientation with Java
Introduction

Shervin Hajiamini

Before: PhD at Washington State University

Email: hajiamini@grinnell.edu

Course site: http://hajiamini.cs.grinnell.edu/courses/CSC207/

- Syllabus
- Schedule of topics
- Labs and assignments
Notes from Syllabus

• CSC 207 learning outcomes:
  ▪ Understand primitive and non-primitive data types
  ▪ Find details about the cause and place of program errors
  ▪ Create objects that contain data and operations
  ▪ Calculate program runtime using mathematical functions
  ▪ Test program correctness unit testing
  ▪ Apply various data structures that each stores or retrieves data in a specific order

➢ Use Java programming language for solving the object-oriented problems
Notes from Syllabus (cont.)

• Grading breakdown
  ▪ Reading assignments: 5%
  ▪ Labs: 10%
  ▪ Quizzes: 10%
  ▪ Programming assignments: 45%
  ▪ Midterm exam: 15%
  ▪ Final exam: 15%
Reading Assignments

• For each topic, do the assigned reading
• Watch the pre-recorded lectures
• Submit your answers by the class time
Labs

- Labs are done during the class time.
- Class attendance is mandatory.
- You are paired with another student for doing labs.
- Submit your works by Saturday.
Quizzes

• Quizzes are taken weekly.
• Questions come from readings in the preceding week.
Programming Assignments

- Assignments are individual works.
- Deadlines are in Schedule of Topics document.
- After a deadline, 10% is deducted for each late day.
Exams

• Midterm and final exams are take-home.
• The exams cover selected topics from the readings.
Resources

• Office Hours: Monday to Friday, 2:30pm - 3:00pm

• Mentor: Tino Tazvitya
  ▪ Email: tazvitya@grinnell.edu
  ▪ Tino will hold mentor session weekly.
  ➢ Fill out doodle polls
Pair Programming Guidelines

• Mutual respect: be respectful and patient.

• Communication
  ▪ Listen carefully and respond respectfully.
  ▪ Speak up for asking questions and clarifying ideas.
  ▪ Respond promptly to outside-class communications (e.g., emails).
  ▪ Be open-minded to accept correction.

• Outside class meet-up
  ▪ Make time to meet your partners. Show up on time.
  ▪ If you cannot make it, let your partner know ahead of time.

• Give your best effort on problems
Lab Partners

• You and your lab partner may not have the same coding skills.
• This happens in most real-world working situations.

• As an experienced partner
  ▪ You benefit from finding errors by answering your partner’s questions.
  ▪ Strengthens your communication skill.

• As a less-experienced partner
  ▪ Ask for clarification. Communicate with your partner as much as possible.
  ▪ Studies show that paired work is better than solo work done by the experience partner.
Object Orientation
Algorithms and Data Structures

• An algorithm is a set of *instructions* to solve a problem.
• Data structure is a mechanism for *organizing* (structuring) data in memory.

• For example, consider a book shelf.
  ▪ Organization: arrange books alphabetically by author name
  ▪ Algorithm: find a book given an author name

• The choice of data structure affects the *performance* of algorithms.
  ▪ If the books are not ordered, it takes a long time to find a book.
Programming Paradigm

• Paradigm is a way for expressing an algorithm.

• Early algorithms were written based on examples.

• Modern paradigms automate the steps of algorithms.

• With the imperative paradigm, an algorithm is a sequence of individual steps.
  • Imperative algorithms look like recipes in a cookbook.

• With the object-oriented paradigm, an algorithm deals with objects and their interactions.
Object-oriented programming

• Object
  ▪ Something that collects data and capabilities.
  ▪ Consider a book as an object
    Data: author name, publisher
    Capability: display the text of a page

• Atomicity
  – Encapsulation
  – Information hiding

• Code reuse
  – Implement an object once and use it repeatedly
“Hello, World!” program