Introduction and Overview

Introduction to Programming and Computational Problem Solving - 2

CSE 8B
CSE 8B: Introduction to Programming and Computational Problem Solving - 2

• Today
  – Course overview
  – Logistics
Introduction to programming courses

• A long time ago in a galaxy far, far away...
  – CSE 8A did not exist
  – CSE 8B did not exist
  – CSE 10: Introduction to Computer Science and Object-Oriented Programming using C++
  – CSE 11 did not exist
  – CSE 12: Basic Data Structures and Object-Oriented Design using C++
Introduction to programming courses

• Then, the **first** change: **Java**
  – CSE 8A did not exist
  – CSE 8B did not exist
  – CSE 10: Introduction to Computer Science and Object-Oriented Programming using **C++**
  – CSE 11: Introduction to Computer Science and Object-Oriented Programming using **Java**
  – CSE 12: Basic Data Structures and Object-Oriented Design using **C++** and/or **Java**
Introduction to programming courses

• Then, the second change: no more C++
  – CSE 8A did not exist
  – CSE 8B did not exist
  – CSE 10: Introduction to Computer Science and Object-Oriented Programming using C++
  – CSE 11: Introduction to Computer Science and Object-Oriented Programming using Java
  – CSE 12: Basic Data Structures and Object-Oriented Design using Java (and C++)
Introduction to programming courses

• Then, the **third** change: half-paced version of CSE 11 as two courses
  – CSE 8A: Introduction to Computer Science and Object-Oriented Programming using **Java I**
  – CSE 8B: Introduction to Computer Science and Object-Oriented Programming using **Java II**
  – CSE 11: Introduction to Computer Science and Object-Oriented Programming using **Java**
  – CSE 12: Basic Data Structures and Object-Oriented Design using **Java** (and **C++**)
Introduction to programming courses

• Starting this quarter, the **fourth** change: reboot CSE 8A, 8B, and 11
  – CSE 8A: Introduction to Programming and Computational Problem Solving - 1
  – CSE 8B: Introduction to Programming and Computational Problem Solving - 2 using **Java**
  – CSE 11: Introduction to Programming and Computational Problem Solving - Accelerated
  – CSE 12: Basic Data Structures and Object-Oriented Design using **Java** (and **C++**)
Introduction to programming courses

• CSE 8A: Introduction to Programming and Computational Problem Solving - 1
  – Introduction to (procedural) programming
  – No specific programming language

• CSE 8B: Introduction to Programming and Computational Problem Solving - 2 using Java
  – Introduction to object-oriented programming
  – Does not assume knowledge of Java

• CSE 11: Introduction to Programming and Computational Problem Solving – Accelerated
  – CSE 8A + CSE 8B in one quarter
CSE 8B topics

• Introduction to Java and review fundamentals of programming
• Objects and classes
• Object-oriented thinking
• Inheritance and polymorphism
• Exception handling and text input/output (I/O)
• Abstract classes and interfaces
• Recursion
• Event-driven programming
• Binary I/O
Introduction to Java and review fundamentals of programming

- Introduction to Java and programs
- Elementary programming
- Selections
- Mathematical functions, characters, and strings
- Loops
- Methods
- Single-dimension arrays and multidimensional arrays
Object-oriented programming

• Object-oriented programming (OOP) involves programming using objects

• This is the focus of CSE 8B
Objects and classes

• An object represents an entity in the real world that can be distinctly identified
  – For example, a student, a desk, a circle, a button, and even a loan can all be viewed as objects
  – An object has a unique identity, state, and behaviors

• Classes are constructs that define objects of the same type
Object-oriented thinking

- Classes provide more flexibility and modularity for building reusable software
- Class abstraction
  - Separate class implementation from the use of the class
  - The creator of the class provides a description of the class and let the user know how the class can be used
  - The user of the class does not need to know how the class is implemented
  - The detail of implementation is encapsulated and hidden from the user
Inheritance and polymorphism

• A subclass inherits from a superclass
  – For example, both a circle and a rectangle are geometric objects
    • GeometricObject is a superclass
    • Circle is a subclass of GeometricObject
      – Circle is-a GeometricObject
    • Rectangle is a subclass of GeometricObject
      – Rectangle is-a GeometricObject

• Polymorphism means that a variable of a supertype can refer to a subtype object
Exception handling

• Exceptions are errors caused by your program and external circumstances
  – These errors can be caught and handled by your program
Text I/O

• In order to perform I/O, you need to create objects using appropriate Java I/O classes
  – The objects contain the methods for reading/writing data from/to a file
Abstract classes and interfaces

• An abstract class cannot be used to create objects
• An abstract class can contain abstract methods that are implemented in concrete subclasses
• An interface is a classlike construct that contains only constants and abstract methods
  – In many ways, an interface is similar to an abstract class, but the intent of an interface is to specify common behavior for objects
Recursion

• Recursion is a technique that leads to elegant solutions to problems that are difficult to program using simple loops
• A recursive method is one that invokes itself directly or indirectly
Event-driven programming

• An event is an object created from an event source

• You can write code to process events such as a button click, mouse movement, and keystrokes
Binary I/O

- Binary I/O does not involve encoding or decoding and thus is more efficient than text I/O.
Syllabus

• Instructor: Ben Ochoa
• TAs: Ziang (Andy) Jing, Shangqing (Shawn) Gu, and Yundong Wang
• Tutor: Zhanchong (Bobby) Deng
• Course is on Canvas
• Course website
  – https://cseweb.ucsd.edu/classes/fa20/cse8B-a/index.html
• 18 lecture meetings (recorded)
  – 1 university holiday (Wednesday, November 11)
  – Thanksgiving Eve (Wednesday, November 25)
• Weekly discussion section (recorded)
• Class discussion
  – Piazza
Syllabus

• Grading
  – 8 homework assignments (50% of grade)
    • Programming using Java
    • Late policy: 15% grade reduction for each 12 hours late
      – Will not be accepted 72 hours after the due date
    • Lowest assignment grade is dropped
  – 8 quizzes and final exam (50% of grade)
    • Quiz two days after assignments are due
    • Final exam has 9 parts
      – First 8 parts correspond to quizzes and can increase your grade
Syllabus

• Grading
  – Extra credit
    • Education research study
      – You must be at least 18 years old to participate
      – Pre-study survey (this Friday)
      – Weekly reflections (same day as quizzes)
      – Post-study survey (after final exam)
      – 2% for participation, not your answers (so be honest)
    • Degree planning
      – Develop or refine your academic plan (mid-quarter)
      – 1%
    • Piazza
      – Ask (and answer) questions using Piazza, not email
      – Good participation could raise your grade (e.g., raise a B+ to an A-)
Textbook

• Introduction to Java Programming and Data Structures, 11th edition, comprehensive version
  – Y. Daniel Liang

• See book website
  – Errata
Collaboration Policy

It is expected that you complete your academic assignments on your own (or your group's, if explicitly allowed for an assignment) own words and code. The assignments have been developed by the instructor to facilitate your learning and to provide a method for fairly evaluating your knowledge and abilities (not the knowledge and abilities of others). So, to facilitate learning, you are authorized to discuss assignments with others; however, to ensure fair evaluations, you are not authorized to use the answers developed by another, copy the work completed by others in the past or present, or write your academic assignments in collaboration with another person (or group, if explicitly allowed for an assignment). On quizzes or exams, collaboration or copying of any kind is not allowed.
Academic Integrity Policy

Integrity of scholarship is essential for an academic community. The University expects that both faculty and students will honor this principle and in so doing protect the validity of University intellectual work. For students, this means that all academic work will be done by the individual (or group, if explicitly allowed for an assignment) to whom it is assigned, without unauthorized aid of any kind.
Academic Integrity Policy

You should not attempt to search for homework solutions online or in sources outside of the course text. If you accidentally stumble upon a homework solution in an outside source you must cite it in your homework solution. If your solution proves to be too similar to the cited one, you may lose credit on the problem; however, failure to cite the other solution will be treated as an academic integrity violation.
Academic Integrity Violation

If the work you submit is determined to be other than your own (or your group's, if explicitly allowed for an assignment), you will be reported to the Academic Integrity Office for violating UCSD's Policy on Integrity of Scholarship. In accordance with the CSE department academic integrity guidelines, students found committing an academic integrity violation on a homework assignment will receive a 0 on the assignment. Students found committing an academic integrity violation on a quiz or exam will receive an F in the course.
Wait list

• Number of enrolled students is limited by
  – Number of instructional assistants (TAs and tutors)

• General advice
  – Wait for as long as you can

• UCSD policy: concurrent enrollment (Extension) students have lowest priority
Java Software Development
Java Development Kit (JDK)

• Develop on your own computer or UCSD Linux Cloud
  – https://linuxcloud.ucsd.edu

• For CSE 8B, you must use the following JDK
  – Oracle Java SE Development Kit 8 Update 261 (i.e., 8u261)

• Oracle JDK 8u261 is already installed and configured on your UCSD Linux Cloud class account

• If using your own computer, then download it from https://www.oracle.com/java/technologies/javase/java-se-jdk8-downloads.html
Text Editor

• Use whatever text editor you want
  – Notepad, Notepad++, vi (or Vim), Emacs, etc.

• You can also use a more advanced source code editor (e.g., Visual Studio Code), but the instructional team will not assist you in configuring it

• Your source code must compile and run from the command-line
Next Lecture

• Introduction to Java and programs
• Elementary programming
• Reading
  – Chapters 1 and 2