CMPSC 111
Introduction to Computer Science I
Fall 2015

Exam 2 Study Guide
Delivered: Monday, December 7, 2015
Exam 2: Tuesday, December 15, 2015 at 9:00 am

Introduction

This course will have its final exam on Tuesday, December 15, 2015 from 9:00 to 12:00 noon. The exam will be "closed notes" and "closed book" and it will cover the following materials. Please review the "Course Schedule" on the Web site for the course to see the content and slides that we have covered this semester. Students may post questions about this material to our Slack team.

- Chapter One in Lewis & Loftus (e.g., "Introduction to Computation and Programming")
- Chapter Two in Lewis & Loftus, Sections 2.1–2.9 (e.g., "Data and Expressions")
- Chapter Three in Lewis & Loftus, Sections 3.1–3.7 (e.g., "Using Classes and Objects")
- Chapter Four in Lewis & Loftus, Sections 4.1–4.9 (e.g., "Writing Classes")
- Chapter Five in Lewis & Loftus, Sections 5.1–5.6 (e.g., "Conditionals and Loops")
- Chapter Eight in Lewis & Loftus, Sections 8.1–8.4 (e.g., "Using Arrays")
- Chapter Eleven in Lewis & Loftus, Sections 11.1–11.6 (e.g., "Exceptions")
- Using the many commands in the Linux operating system; editing in `gvim`, compiling and executing programs in Linux; knowledge of the basic commands for using `git` and Bitbucket.
- Your class notes and lecture slides, labs 1–10, practicals 1–9, and the handouts from lab.

The exam will be a mix of questions that have a form such as fill in the blank, short answer, true/false, and completion. The emphasis will be on the following topics:

- Fundamental concepts in computing and the Java language (e.g., definitions and background).
- Fundamental concepts in programming languages (e.g., conditional logic and iteration).
- Practical laboratory techniques (e.g., editing, compiling, and running programs; effectively using files and directories; correctly using Bitbucket through the command-line `git` program).
- Understanding Java programs (e.g., given a short, perhaps even one line, source code segment written in Java, understand what it does and be able to precisely describe its output).
- Composing Java statements and programs, given a description of what should be done. Students should be completely comfortable writing short source code statements that are in nearly-correct form as Java code. While your program may contain small syntactic errors, it is not acceptable to "make up" features of the Java programming language that do not exist in the language itself—so, please do not call a "`solveQuestionThree()`" method!

No partial credit will be given for questions that are true/false, completion, or fill in the blank. Minimal partial credit may be awarded for the questions that require a student to write a short answer. You are strongly encouraged to write short, precise, and correct responses to all of the questions. When you are taking the exam, you should do so as a "point maximizer" who first responds to the questions that you are most likely to answer correctly for full points. Please make sure that you review the past quiz so that you can comfortably answer its questions. Students should keep the time limitation in mind as you are absolutely required to submit the examination at the end of the class period unless you have written permission for extra time from a member of the Learning Commons. Students who do not submit their exam on time will have their overall point total reduced. Please see the course instructor if you have questions about these policies.

Reminder Concerning the Honor Code

Students are required to fully adhere to the Honor Code during the completion of this exam. More details about the Allegheny College Honor Code are provided on the syllabus. Students are strongly encouraged to carefully review the full statement of the Honor Code before taking this exam.

The following provides you with a review of Honor Code statement from the course syllabus:

The Academic Honor Program that governs the entire academic program at Allegheny College is described in the Allegheny Academic Bulletin. The Honor Program applies to all work that is submitted for academic credit or to meet non-credit requirements for graduation at Allegheny College. This includes all work assigned for this class (e.g., examinations, laboratory assignments, and the final project). All students who have enrolled in the College will work under the Honor Program. Each student who has matriculated at the College has acknowledged the following pledge:

> I hereby recognize and pledge to fulfill my responsibilities, as defined in the Honor Code, and to maintain the integrity of both myself and the College community as a whole.

Students who have questions about Allegheny College's Honor Code and how it applies to the completion of a quiz or an examination in Computer Science 111, should immediately schedule a meeting with the course instructor to openly discuss their concerns.

Detailed Review of Content

The listing of topics in the following subsections is not exhaustive; rather, it serves to illustrate the types of concepts that students should study as they prepare for the exam. Please see the course instructor during office hours if you have questions about any of the content listed in this section.

Chapter One

- Basic understanding of computer hardware and software
- Computer number systems (e.g., binary and decimal)
- Purpose for and steps of the fetch-decode-execute cycle in the CPU
- Layout of and access techniques for computer memory
- Knowledge of computer networking methods and programs

- Basic syntax and semantics of the Java programming language
- The correct way to write identifiers in the Java language
- Input(s) and output(s) of the Java compiler and virtual machine
- Basic principles of the object-oriented programming paradigm

Chapter Two

- Ways to perform input and output in a Java program
- Using escape sequences in the output of Java programs
- The variety of data types available to Java programmers
- The declaration of and assignment of values to variables
- Operators and operator precedence in Java expressions
- How to use the modular arithmetic operation in a Java program
- Techniques for converting variables from one data type to another
- Computer graphics and related topics such as pixels and screen resolution
- The use of the RGB system for specifying colors in Java programs
- Programming techniques for implementing applets in the Java language

Chapter Three

- The steps for creating a new instance of a Java class
- How to use technical diagrams to visualize an object in memory
- The difference between objects and primitives in Java programs
- The meaning of the term "alias" in a Java program
- The creation and use of Strings in the Java programming language
- The ways in which Java packages promote high-quality programming
- The variety of ways in which you can create and use random numbers
- How to call and use the methods provided by the `Math` class
- Ways in which programs create formatted output in a terminal window

Chapter Four

- Best practices for organizing and structuring a Java class
- How to declare an instance variable and a method in a Java class
- The ways in which encapsulation promotes good object-oriented design
- The visibility modifiers that support encapsulation in the Java language
- The meaning and purpose of accessor and mutator methods in Java
- All of the key parts of a Java method (e.g., parameters and `return` statements)
- The meaning and purpose of formal and actual parameters for methods
- The meaning of the "`void`" return type for a method in a Java program
- Appropriate strategies for implementing constructors in Java programs
- A basic understanding of the graphical objects seen in user interfaces

Chapter Five

- The meaning and purpose of boolean expressions in conditional logic
- How to write a truth table that explains the meaning of boolean expressions
- The different logical operators available for use in boolean expressions
- The overall structure and purpose of `if` statements in Java
- The overall structure and purpose of `if-else` statements in Java
- The overall structure and purpose of `if/else-if/else` statements in Java
- How to use a truth table to understand the meaning of `if` statements
- Knowledge of the best practices for comparing variables of different data types
- The meaning and purpose of looping constructs in the Java language
- The overall structure and purpose of `for` loops in Java
- The overall structure and purpose of `while` loops in Java
- The overall structure and purpose of `do-while` loops in Java
- The ways in which an `Iterator` can be used in a Java program
- How `break` and `continue` statements work in looping constructs
- The ways in which an `ArrayList` can be used in a Java program

Chapter Eight

- An example of a problem that is best solved through the use of an array
- The types of technical diagrams that are best suited to visualizing an array
- The benefits and drawbacks associated with using arrays in a Java program
- The means by which you define and use arrays in the Java programming language
- The key characteristics of the array data structure (e.g., stores a single type of data)
- The meaning of the word "index" and how it connects to the array data structure
- The meaning and purpose of arrays bounds checking the Java programming language
- How arrays are used to accept command-line arguments as input to a program
- The ways in which it is possible to pair together iteration constructs and arrays

Chapter Eleven

- Concrete examples of when and why exceptions might be thrown in a Java program
- Strategies for dealing with exceptions that arise during use of a Java program
- An explanation for why division by zero will raise an exception in a Java program
- The similarities and differences between caught and uncaught exceptions
- The structure and purpose of the `try-catch` construct in Java programs
- The structure and purpose of the `finally` clause in a Java program
- The ways in which exceptions are "propagated" when running a Java program
- How the Java programming language supports the handling of input/output exceptions
- Concrete examples of at least three different exception classes available in Java