Exam 1 Study Guide

Preparing for Exam 1

- Review the learning objectives
- Review the lecture slides
- Review zyBook, and the activities
- Review (and possibly rewrite) the lab exercises
- If you do not fully understand a topic, read the related textbook section
- Attend office hours to ask additional questions/clarifications
- Complete the practice problems

Exam 1 Learning Objectives

- Write-Compile-Execute
  - Explain the following steps when programming: write, compile, execute.
- Structure of Java
  - Describe the structure of a Java program and how the following entities relate to each other: classes, methods, statements.
  - Create a Java program for a given algorithm.
  - Explain the importance of documenting and commenting code.
  - Describe the importance of the `main` method.
- Program Errors
  - Identify program errors as syntax/compiler errors, logic errors, or runtime errors.
- Debugging
  - Debug a program to remove syntax errors, runtime errors, and/or logic errors.
- Scanner (console input)
  - Construct new Scanner object for console input.
  - Write code to read user input from the console.
  - Write and trace code that uses `nextInt`, `nextDouble`, `next`, and `nextLine` methods.

- Data Types (i.e., primitive data and object)
    - Explain what data types are.
    - Use various data types in Java.
- Expressions
    - Evaluate expressions containing ints, doubles, or mixed types of addition, subtraction, multiplication, division, and modulo operators.
    - Explain how promotion/coercion works in Java.
    - Explain how casting works in Java.
    - Write code containing primitive data types.
- Variables
    - Declare, initialize/assign value to, and use variables in code.
    - Design algorithms that use variables.
    - Trace values of variables through an algorithm or a segment of code.
    - Write Java programs that use variables.
- Formatting Text with printf
    - Write code that used the System.out.printf to output a formatted string.
- Constants and Class Constants
    - Describe when constants should be used in code.
    - Write code that properly uses constants.
- Math class
    - Define basic methods for Math class (e.g., abs, min, max, pow, sqrt, ceil, round).
    - Write and trace code that uses Math class methods.
    - Write and trace code that uses Math class constants.
- Strings
    - Construct new String objects.
    - Write and trace code that uses String methods.
- Static Methods: Parameters and Return Values
    - State the syntax of a static method.
    - Write a static method.
    - Explain what parameters are and why using them.

  - Define and write method header with parameters.
  - Trace a method with parameter(s) to determine the output.
  - Write a method with parameter(s) for a given programming problem.
  - Describe what return values are.
  - Define and write method header with return type.
  - Trace a method with a return value.
  - Write a method with a return value.
- Passing Parameters
  - Describe how parameters are passed to methods based on their types.
- Flow of Control
  - Describe the order in which statements in a Java program are executed.
  - Describe why flow of control is important.
  - Trace code with an understanding of flow of control.
- Scanner as Parameter
  - Write code that contains a `Scanner` as a parameter.
- Equality, Relational, and Logical Operators
  - Evaluate expressions containing equality, relational, and logical operators.
  - Write code containing equality, relational, and logical operators.
  - Negate boolean expressions using De Morgan's Law.
- boolean Data Type
  - Evaluate boolean expressions, including short-circuit evaluations.
- Conditionals
  - Trace conditional (e.g., if or if-else) statements and provide output.
  - Write conditional (e.g., if or if-else) statements to perform an operation or produce specified output.
  - Distinguish between when to use each conditional structure.