CIS 255 (Java Programming)
Exam 2 Study Guide
Jefferson State Community College • Instructor: T. Battles

Format

- Twelve multiple-choice questions, including terminology, concepts, and syntax
- A hand-tracing problem: given a complete program, indicate the output the program generates
- Coding: write the code to solve a specific problem

Source Material: Chapters 4 and 5 in the textbook, with emphasis on concepts in the lectures (these chapters depend on material from the earlier chapters)

Exam aides: One double-sided or two single-sided pieces of paper up to letter-sized (8 ½" x 11") with any material (notes, sample code) that would be helpful for any section of the exam. You may use a basic or scientific calculator (standalone, not as an app), but otherwise **no interaction with computers, cell phones, or other electronic devices is allowed.**

Key Elements to Study

Chapter 4

- Increment and decrement
  - Operator syntax
  - Placement (effect of prefix vs. postfix)
- Loops
  - Basic terminology: loop body, iteration, pretest vs. posttest, used for input validation
  - `while` loop
    - Loop header: key word, Boolean expression in parentheses (with **no** semicolon afterwards)
    - Curly brackets around loop body
    - Inclusion of statement in loop body to update variable in Boolean expression
    - Possibility of zero iterations with a Boolean expression that is initially `false`
  - `do-while` loop
    - Loop header and footer key words
    - Placement of Boolean expression after loop body (**with** semicolon afterwards)
    - Minimum of one iteration regardless of Boolean expression's initial state
  - `for` loop
    - Loop header syntax: three sections in the loop header separated by semicolons (with **no** semicolon after the closing parenthesis)
    - Inclusion of multiple initialization or update statements in the header
    - Omission of initialization or update statements in the header
    - Flow of execution
  - Loops applied
    - Input validation: forcing the user to enter replacement values until input is valid
    - Accumulation (initial value, process of adding the values)
    - Terminating a loop with a sentinel value
    - Nesting one loop inside another (how this execution proceeds)
  - `break` and `continue`: how they affect a program's execution, why they should be avoided

- File Input and Output
  - Wildcard `import` statement and `throws` clause required
  - `PrintWriter`
    - Declaring with file name
    - Writing to the file
    - Closing the file (why this is important)
    - The use of `FileWriter` with `PrintWriter` to indicate appending rather than overwriting
  - `File` and `Scanner`
    - Declarations
    - Using `Scanner` methods
    - Closing the file
    - Using a loop to read an entire file
    - Checking for a file's existence
- Random numbers
  - Declaring a `Random` object
  - Invoking methods for various types of random numbers
    - When an argument is required
    - The range of the random numbers returned by each method call

Chapter 5

- Method definitions
  - Location within class
  - Method Header
    - Key words and their meanings
    - Naming a method
    - Including a parameter list to receive values (data types and names)
  - Body
    - Using local variables
    - When a `return` statement is required and what it must include
    - The difference in `return` for `void` methods versus for value-returning methods
    - Documentation: comment structure with tags for parameters, return value
- Method calls
  - Calling a `void` method
  - Calling a value-returning method (handling the value returned)
  - Calling a method within a structure (decision, loop)
  - Ensuring that the argument list matches the parameter list
  - Difference between primitive arguments and object arguments; effect of immutability
  - Flow of control with methods other than `main` calling other methods
- Reading from / writing to files in multiple methods
  - Sending the objects as arguments to parameters for various methods
  - Where a `throws` clause is required
- Overloading
  - What a method signature is
  - How to make each method's signature unique
  - How Java matches a call to a definition