Master Syllabus for TCSS 143
Version: September 2009

Prerequisites

- TCSS 142 and its prerequisites, or equivalent

Preconditions

- expressions, variables, assignment
- usage of Java types `int`, `double`, `boolean`, `char`, `String`, `Math`
- output with `System.out.println`
- input from the console and files using, for example, `java.util.Scanner`
- selection with `if/else` and `switch` statements
- repetition with `while`, `for`, and `do/while` statements
- creation and manipulation of one-dimensional arrays; exposure to multi-dimensional arrays
- procedural decomposition with static methods
- data / control abstraction by writing simple classes with fields, constructors, and methods
- ability to read a problem / assignment description and implement a solution usinga classbasic programming style such as whitespace and commenting

Topics Covered for Comprehension and Application

- basic programming style and object-oriented design
  - more formal documentation using Javadoc comments
  - problem decomposition into classes and methods
  - class design: state and behavior (data and control abstraction), encapsulation, usage of access modifiers, static vs. non-static
- Java language / API
  - access modifiers: `public`, `private`, `protected`
  - creating an inheritance hierarchy in Java; creating and implementing an interface
  - exceptions: `try/catch` statement, throwing existing exception types to propagate errors
  - the "list family" subset of Java's collections framework: `List`, `LinkedList`, `ArrayList`, `Stack`
- data structures
  - usage of provided lists (linked list, array list)
  - usage of stacks and queues
  - implementation of linked data structures such as linked list

Postcondition Skills

- implement a basic input / process / output loop
- facility using single and multidimensional arrays to solve problems
- use Java's collection framework to solve interesting problems
- use a provided class, given only its API (e.g. `Scanner` for file I/O)
- class definition: define a class, including implementing interfaces, extending abase class, overriding methods from the Object or other base class, and using that class in an application
- implementing code to match a problem statement, including programs with more than one class using recursive functions to solve programming problems