**CS 1400 Final Exam Study Guide**

This exam is a comprehensive exam and will cover topics discussed in class and worked via in-class and homework assignments. It will encompass topics from Exam #1, Exam #2, and the quizzes.

Following is a list of topics you should study/know

* How Java works (i.e. the Java Virtual Machine).
* The 8 primitive data types and what kinds of data they can store.
* The wrapper classes and how to use them.
* How to write data to the console.
* How to read in data from the user on the console.
* How to work with String objects, including comparison, manipulation, concatenation, and utilizing the String methods.
* How to work with if, else if, and else.
* Understand identifier scope.
* How to work with switch statements.
* How to work with for loops, while loops, and do while loops.
* How to work with nested loops.
* How to work with methods. This includes overloading methods.
* How to work with classes. This includes what public and private mean, what methods are, and what data members are.
* Be able to read code on paper and understand what the logical flow of the program will be if it were to be run on a computer.
* Be able to write code by hand on paper so that if it were placed on a computer, it would compile and run.
* How to work with arrays, how to define them, how to size them, how to pass them into methods, etc. Also know how to work with 2 dimensional arrays.
* What libraries are, and what the import statement means.
* How to work with the Math class and its methods we used in class and in the homework (min, max, random, etc.)
* Know what a constructor is. How to create one. What it means if a constructor has zero arguments. What overloading constructors mean and how to do it.
* How to override and produce your own .toString() method.
* How to work with the ArrayList class. The differences and similarities between arrays and ArrayLists
* Know the difference between local, instance, and class members (variables and methods).
* How to code a constructor, accessor and mutator methods
* What a compound operator is and how to use it.
* What a pre-fix/post-fix operator is and how to use it
* Logic operators
* Operators && and || and how they are used.
* What a logic operator is and how to use it
* Access modifiers – public, private, protected, final static, etc. – what they are and how they are used.
* How, when, and why to use helper methods
* What encapsulation is and how Java provides it.
* What inheritance is and why it is important. How Java implements it.
* What aggregation is and why it is important. How Java implements it.
* What composition is and why it is important. How Java implements it.
* What @Overrides does
* What polymorphism is and how Java implements it.
* What a “has-a” relationship is in Java.
* What a “is-a” relationship is in Java.
* How the instanceof operator works.
* What an abstract class is and how it is used.
* What an abstract method is and how it is used.
* What an exception is and how to handle it.
* How the try and catch blocks work and how they should be coded.
* What the exception class is.
* A comparison between a super class and sub class and the proper syntax to code them.
* A general idea of how recursion works and the conditions related to using a recursive method. Some examples where recursion is useful.
* What a break statement does in a loop
* How to write and use a switch statement
* What an enumerated type is and how to use it.
* What checked and unchecked exceptions are.
* The syntax of a “for-each” loop and how to use it in code.
* The differences between top-down and bottom-up design.
* What the “this” reference is and how it works.
* What a super class/sub class is and how they are related.
* What pass-by-value means

In addition to the above, you should know the following terms. Some definitions are provided here.

Primitive - 8 data types that are represented in raw bit form, they are NOT objects.

Object - An instantiated instance of a class. It can be manipulated (mutable). It

can have methods and data members.

Class - A blueprint for objects.

Method - A block of code that can be executed, that is part of a class. You call a normal method through an object. (Static methods are called through a class, not an object).

Data member - A variable stored in an object. A variable can be either a primitive, or another object, or an enum. Often programmers call data members a field.

Member - Anything belonging to an object. There are method members and data members. (Using static, it is also possible to have class method members and class data members.)

Constructor - Like a method, but not a method. You can't call it directly, when an object is created/instantiated, the constructor runs. The constructor is a block of code that is executed and helps initialize an object.

An overloaded constructor - A constructor that takes different arguments. To be "overloaded" there must be at least two constructors that exist.

Overloaded method - A method that takes different arguments. To be "overloaded" there must be at least two methods of the same name that exist. Overloaded methods cannot change the return type, the return type must be the same.

Default constuctor - A constructor that takes no arguments.

Argument - Related to parameter, argument is what you pass in.

Parameter - Related to argument, the definition in a method or constructor for what the arguments data type and name will be.

Type - Also commonly called data type. The nature of the variable/indentifier. It could be a primitive or an object or an enum. String is a type, int is a type, etc.

Return type - Related to methods. What a method promises to return.

Scope - The lifetime or the valid region of code in which a variable/identifier is known, or other classes are known. Use the curly brace rule, if something is defined within a set of curly braces, then outside those curly braces, it's out of scope. Scope also refers to public and private, which refers to when certain class

members can be accessed.

Identifier - The name of a variable or object.

Static - Belongs to the class, not to an object. If there are zero objects, there will be one of that static thing. If there are a thousand objects, there is still only one static thing.

Java Virtual Machine - The tools that help a Java program run on the machine. (JRE is the Java Virtual Machine. The JDK contain tools to help develop, specifically javac, the compiler.

IDE - Integrated Development Environment, an example is NetBeans. A program that makes coding much easier, it assists you with intellisense, debugging, compiling, etc.

Final - Putting final before a data type makes it a constant, e.g. final double pi = 3.1415;

Cast - To convert some data from one data type to another.

library/API - Tools associated with a product or language that assist you in developing by providing code that handles very common tasks.

Arrays - A contiguous block of data with a homogenous set of data. Array elements can be accessed with [].

ArrayList - An object which is a collection of other objects. It helps maintain a collection of items, it keeps track of order of the items, and allows you to easily add, resize, and remove items. ArrayList serves to do things that arrays are bad at.

Void - A datatype that means there isn't a data type. Void before a method means no data needs to be returned.

NULL - Specifically data that is all zero bits. Not necessarily the value 0.

public static void main() - The launching point for a program. A program should only have one of these.

Invocation - When you know a line of code will invoke or run another set of code.

Calling a method can be described as invoking a method. Creating an object will invoke the constructor.

New keyword - Allocates space for an object and provides a memory reference to the object.

Dot operator - Allows you to access members of a class or method.