Michael Camara

Honor code pledge: This work is mine unless otherwise cited

CMPSC 220

Due Date 11/12/15

Lab 8: Object Oriented Programming

Java and C++ share many similarities, but there are some important differences that a Java programmer needs to know when making a C++ program. These include the following:

1) *#include* and *using*: In order to access external libraries or classes, the *#include* preprocessor directive needs to be used, similar to the *import* statement in Java. Additionally, the *using namespace* statement can be used to allow shorthand for certain statements within a package. For instance, *#include <string>* will allow use of the string class in the std package via the syntax: *std:string*. Additionally, typing *using namespace std* will allow the user to simply type *string*, providing some degree of convenience.
2) Input and Output: In Java, most output is handled via the System.out class, and input is often handled with the Scanner class. In C++, the syntax for input and output is much different. *cin* and *cout* from the *iostream* class are used with the >> and << operators in the following style:
    a. cin >> name; // Retrieve string from user and store in the name variable
    b. cout << “Hello world!”; // Send the string, “Hello world!”, to the console

3) Header files: Separate “header” files are typically required for a class in C++. These files describe all method signatures and instance variables for a class, but they do not contain any method bodies or initialized variables. This provides a way to visualize a class’ structure easily, although it will require changing both this header (.h) file and the source (.cpp) file if new methods or instance variables need to be added, or if existing constructs need to be updated.

4) Accessing members of a class: Even within the same class, the “::” operator needs to be used to access members within that class. For instance, to access the instance variable *name* within the *Gator* class, one would need to type *Gator::name* in C++, whereas in Java you could simply type *name*. However, local variables declared within a method do not need this added notation and can simply be called by their declared name if they are referenced within the same method.

5) Instantiation: In Java, you often need to use the *new* operator to create an object of a class (e.g. Scanner scan = new Scanner(System.in)). In C++, this does not appear to be necessary. You simply specify the class name, followed by a variable name, and then arguments for a constructor (if needed). For example, to instantiate Gator objects you use the format: *Gator gator1(name1, “red”).*