CptS 122 – Data Structures

Lab 11: Inheritance in C++

Assigned: Monday, July 17, 2017
Due: At the end of the lab session

I. Learner Objectives:

At the conclusion of this programming assignment, participants should be able to:

- Design, implement and test classes in C++ which apply inheritance
- Compare and contrast inheritance ("is-a") relationships versus composition ("has-a") relationships
- Apply and implement *overloaded* functions and operators

II. Prerequisites:

Before starting this programming assignment, participants should be able to:

- Analyze a basic set of requirements for a problem
- Create test cases for a program
- Design, implement and test classes in C++
- Declare and define *constructors*
- Declare and define *destructors*
- Compare and contrast *public, protected,* and *private* access specifiers in C++
- Describe what is an *attribute* or data member of a class
- Describe what is a *method* of a class
- Apply and implement *overloaded* functions
- Distinguish between pass-by-*value* and pass-by-*reference*
- Discuss *classes* versus *objects*

III. Overview & Requirements:

This lab, along with your TA, will help you navigate through designing, implementing, and testing inheritance with classes in C++. It will also help you with understanding how to apply inheritance to an application. You should NOT apply polymorphism in this lab!

Labs are held in a "closed" environment such that you may ask your TA questions. Please use your TAs knowledge to your advantage. You are required to move at the pace set forth by your TA. Please help other students in need when you are finished with a task. You may work in pairs if you wish. However, I encourage you to compose your own solution to each problem. Have a great time! Labs are a vital part to your education in CptS 122 so work diligently.

I highly encourage you to work in teams for these tasks!

Tasks:

Task 1: Create a base *class* called *Person*, which has *private* data members for a name, age, gender, and height. Implement a constructor, copy constructor, destructor, overloaded assignment operator, overloaded stream insertion and extraction operators, and getters and setters. See Task 2 for testing this class.

Task 2: Create a derived *class* called *TestPerson*, which *publically* inherits from *class Person*. Create test methods for each method in *class Person*. Remember these functions should not accept any parameters or return any values. However, they should print messages for "test case passed" or "test case failed".

Task 3: Modify your *class Person* so that the data members are *protected* instead of *private*. How does this affect the tests cases that you created in Task 2?

Task 4: Create a derived *class* called *Student*, which *publically* inherits from *class* Person. Add three *private* data members to *class Student*. These include an array of *struct Course*, the number of courses taken, and the total number of credits. The *struct Course* should contain a string for course name, credits assigned to course, and current grade in course. Implement appropriate constructors, overloaded operators, and setters and getters for this class. Also, implement methods for computing total credits taken and current GPA. How would your implementation change if you decided to define a *class Course* instead of *struct Course*?

Task 5: Create a derived *class* called *Teacher*, which *publically* inherits from *class Person*. Add three *private* data members to *class Teacher*. These include an array of *struct Course*, the number of courses taken, and the total number of credits. The *struct Course* should contain a string for course name, credits assigned to course, and average grade of students' in course. Implement appropriate constructors, overloaded operators, and setters and getters for this class. Also, implement methods for computing total credits taught and average grades of students across the courses taught. How would your implementation change if you decided to derive *class Teacher* from *class Student* instead of *class Person*?

Task 6: Create an application, which allows students to register for classes taught by a particular teacher.

IV. Submitting Labs:

- You are not required to submit your lab solutions. However, you should keep them in a folder that you may continue to access throughout the semester. You should not store your solutions to the local C: drive on the EME 120/128 machines. These files are erased on a daily basis.

V. Grading Guidelines:

- This lab is worth 10 points. Your lab grade is assigned based on completeness and effort. To receive full credit for the lab you must show up on time and continue to work on the problems until the TA has dismissed you.