CSCE 2004 – Programming Project 1
Due Date – Feb 1, 2017 at 11:59pm

1. Problem Statement:

The goal of this programming assignment is to give students experience with input/output commands and basic numerical calculations in C++. Specifically, your program should do the following:

- Print a message asking the user to enter the A,B,C coefficients of a quadratic equation: y(x) = A*x^2 + B*x + C. Save these values in three variables.
- Calculate and print the two roots of the quadratic equation using the classic quadratic formula (see https://en.wikipedia.org/wiki/Quadratic_formula). For this program, you can assume the user will always enter A,B,C values that result in a positive discriminant value, so no error checking is needed.
- Calculate and print the X value that is halfway between the two roots of the quadratic equation. If we are lucky, this should be the X location where the quadratic curve has a local maximum or local minimum.
- Prompt the user to enter an X value. Read this value and calculate and print the corresponding value of y(x) using the quadratic formula above.
- Prompt the user to enter a second X value. Read this value and calculate and print y'(x) using the first derivative of quadratic formula: y'(x) = 2*A*x + B.

2. Design:

Your first design task is to decide how to calculate the two roots of the quadratic equation. For this, you will need to look up the classic quadratic formula and think about the sequence of operations you need to perform to calculate these values. To implement the square root operation, you will need to call the sqrt() function that is part of the <cmath> library.

Your next task is to decide what sequence of operations are needed to evaluate the quadratic equation at a given X value. Your final design task is to decide what sequence of operations are needed to evaluate the derivative of the quadratic equation at a given X value. Make sure you explain your design decisions in your code using comments and again later in your project report.

3. Implementation:

Since you are starting with a "blank piece of paper" to implement this project, it is very important to develop your code incrementally writing comments, adding code, compiling, debugging, a little bit at a time. This way, you always have a program that "does something" even if it is not complete.

As a first step, it is always a good idea to start with an empty main function, and add the code to read input data from the user, and then simply print these values back out again. Once this part is working, you can start performing calculations with the input data.

4. Testing:

Test your program to check that it operates correctly for all of the requirements listed above. To verify that you are calculating roots properly, you may want to use A,B,C values where you know the location of the roots in advance. For example, quadratic equations of the form: y(x) = (x-a) * (x-b) = x^2 - (a+b)*x + a*b will have two roots at x=a and x=b.

You are NOT required to add error checking in this program, but it is always good to test a program to see what happens if the user inputs unexpected data (like “hello mom”) or if the user enters A,B,C values that result in a negative discriminant. You should cut/paste these results into your project report to document what your program does in these cases.

5. Documentation:

When you have completed your C++ program, write a short report using the “Programming Project Report Template” describing what the objectives were, what you did, and the status of the program. Does it work properly for all test cases? Are there any known problems? Save this project report in a separate document to be submitted electronically.

6. Project Submission:

In this class, we will be using electronic project submission to make sure that all students hand their programming projects and labs on time, and to perform automatic plagiarism analysis of all programs that are submitted.

When you have completed the tasks above go to Blackboard to upload your documentation (a single docx or pdf file), and your C++ program (a single cpp or txt file). Do NOT upload an executable version of your program.

The dates on your electronic submission will be used to verify that you met the due date above. All late projects will receive reduced credit:

> 10% off if less than 1 day late,
> 20% off if less than 2 days late,
> 30% off if less than 3 days late,
> no credit if more than 3 days late.

You will receive partial credit for all programs that compile even if they do not meet all program requirements, so handing projects in on time is highly recommended.

7. Academic Honesty Statement:

Students are expected to submit their own work on all programming projects, unless group projects have been explicitly assigned. Students are NOT allowed to distribute code to each other, or copy code from another individual or website. Students ARE allowed to use any materials on the class website, or in the textbook, or ask the instructor and/or GTAs for assistance.

This course will be using highly effective program comparison software to calculate the similarity of all programs to each other, and to homework assignments from previous semesters. Please do not be tempted to plagiarize from another student.

Violations of the policies above will be reported to the Provost's office and may result in a ZERO on the programming project, an F in the class, or suspension from the university, depending on the severity of the violation and any history of prior violations.