CSCE 2004 – Programming Project 3
Midpoint Due Date – Mar 17, 2017 at 11:59pm
Final Due Date – Mar 22, 2017 at 11:59pm

1. Problem Statement:

The primary goal of this programming assignment is to give students experience defining and calling functions. You will start with a partial implementation of a game modeled after the 1970's computer game called "Adventure". This game was written before computer graphics was invented, so the game was all "text based" (it was still a huge hit back in the day).

You will be staring this programming project with some sample code in "project4.cpp". This program consists of a collection of functions that are intended to simplify your implementation. You may call these functions any way you want in your project, but you must not change the parameters definitions. Several of the functions describe what is found in different rooms of the cave. The other functions are used to add to the total treasure, fight battle with creatures, and get directions from the user playing the game. To finish the game you must complete the following tasks:

- **Design a maze of rooms:** On a piece of paper, draw a maze of 8 rooms, with Room1 as the start of the maze, and Exit as the exit of the maze. Draw a collection of N,S,E,W lines connecting the rooms to each other. Try to make it a little tricky to make it from Room1 to Exit. When you reach Exit, the program will print a message and exit.

- **Create new room functions:** Using the existing room functions as examples, you will have to write 4 new room functions with short descriptions of what is found in each room. Your descriptions don't need to be long, but make sure they are "politically correct" so you can demo your game to your friends and family.

- **Add room navigation:** The current room functions print out room descriptions and add to the total treasure, but they are not connected to each other in any way. You need to add some code somewhere in the program to ask the user for directions. Using this information and your map from above, you need to call the corresponding room functions. For example, if Room4 is N of Room7, then you should call the Room4 function when the user types an N when they are in Room7. If you want your game to be "geographically realistic" you should implement navigation such that when the user types S in Room4, they will return back to Room7.

- **Play the game:** Add some code to the main function to initialize variables and start the game in Room1 of the maze. Then type in a sequence of N,S,E,W commands to move through the maze and collect gold on the way to the Exit exit. Save a copy of your program output in a text file to be included in your final report.

Students are allowed to use any combination of C++ features we have discussed in class or lab. You are not required or allowed to look ahead to more advanced C++ features like arrays, files or classes to perform this task.

2. Design:

For this assignment, you have two big design decisions. First, you must create your map of 8 rooms in the maze. Connecting your rooms with N,S,E,W is the fun part. One tricky point is deciding what to do if a user types in E when they are in a room that does not have another room to the E side. The normal solution is to stay in the room and ask the user for directions again. You can think of this as a line that loops back to the same room.

Second, you need to figure out how to navigate from room to room. To do this, you must look at the direction the user entered, and make a recursive call to the corresponding function for that room.

3. Implementation:

You are starting this programming project with our sample program, so you already have something that compiles and runs. Your task is to add several features to this program. It is very important to make these changes incrementally one feature at a time, writing comments, adding code, compiling, and debugging. This way, you always have a program that "does something" even if it is not complete.

4. Testing:

Test your program to check that it operates correctly for all of the requirements listed above. Also check for the error handling capabilities of the code. Try your program with several input values, and save your testing output in text files for inclusion in your project report.

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.