Program #2

Due Oct 7 by 11:59pm **Points** 100

Working with Variables and Conditionals

Introduction

Design is very important when developing programs and there are a variety of ways to approach it. You may draw pictures, write it out in prose or structured text, use pseudo code, and more! The point of design is to give you a blueprint to follow while you are coding. This saves time debugging your program as you can catch mistakes early. It is better to spend one hour of designing than it is to spend five hours debugging.

For this assignment you must design a solution to the following problem statement and implement your solution.

Example Design Document: Polya_template.pdf (https://web.engr.oregonstate.edu/~goinsj/resources/CS161/Polya_template.pdf)

General Grading Breakdown:

- Understanding the problem. (Recognizing what is asked.) (5 pts)
- Devising a plan. (Responding to what is asked.) (10 pts)
- Carrying out the plan. (Developing the result of the response.) (75 pts)
- Test Plan/Looking back. (Checking. What does the result tell me?) (10 pts)

Problem Statement

The local middle school would like some text adventure games to keep their students occupied during down time. The school is leaving it up to your skill and good judgement to develop a game. It is up to you what the story and theme is but there are some requirements:

- There must be **at least five different "if" statements** involved in the adventure
- **At least one** of those "if" statements must contain a nested "if" statement
- There must be **an element of chance** that would change a user’s chosen path
- You must **handle invalid input** from the user.

An example run of the program could look as follows:

```
Hello and welcome to the Wilderness Adventure Game!
To go right enter 1, to go left enter 2: 1

You chose to go right. You have now entered the grasslands and are being followed by oxen. They look friendly.
```

```
Enter 1 to befriend the oxen, enter 2 to run from the oxen: 1

You attempted to befriend the oxen. You think you can ride one of them.
Enter 1 to attempt to ride the ox, enter 2 to walk away: 11

Oops! That wasn’t a valid response. Please try again:  1

Unfortunately fate was not on your side. [due to a random number within C++]
You were thrown from the ox’s back and are forced to walk away.
Enter 1 to walk right, enter 2 to walk left: 1

You chose to go right. You have now entered the swamplands and have spotted an approaching alligator. Enter 1 to h
ide, enter 2 to run home: 2

You chose to go home. The adventure has ended.
```

The rest of the implementation is up to you, but try to make your game as clean and attractive to play as possible.

(25 pts) Design Documentation

You should create a written document that addresses the points listed below. Please see the **example design template** (https://web.engr.oregonstate.edu/~goinsj/resources/CS161/Polya_template.pdf) .

Understanding the Problem (5 pts)

What is the problem asking you to do? What are the inputs for the problem? What questions do you have about the problem?

Devising a Plan (10 pts)

- What is the theme of the adventure going to be?
- What are the options?
- List the decisions you will give in your text adventure?
- Which decisions will have an element of chance?
- How are you going to create the element of chance?
- How are you going to handle invalid input entered by the user?

Test Plan (10 pts)

You need to consider good and bad values for your program. Remember to include tests for less obvious invalid inputs such as the word "one".

Input Value(s):	**Expected Outcome:**	**Actual Outcome:**
-100	Error message, prompt for a new input	Crashes
1	Game moves to next phase on right	Works

2	Game moves to next phase on left	Works
...	...	...

(65 pts) Implementation

In addition to the design process, you must implement your program as a C++ program. Here are implementation requirements:

- There must be an empty line separating each navigation option (e.g. your program should not simply generate a "wall-of-text").
- The choice system must be based on numbers. Non-numeric input is not allowed in this program.
- You must gracefully handle invalid input from the user that is not one of the allowed choices. **Clarification (10/6/18): The TAs will only test your code with numeric input (e.g. -125, 1, 2, 0, 1253, etc).**
- You must use **if/else** statements and/or **switch** statements.
- You need to use the **rand()** function to add an element of chance.
- You must have at least 5 different paths/solutions that can be followed to complete the game. **Clarification (10/6/18): As noted in class, for this assignment you simply need at least 5 "if" statements within the game portion of your code. In addition, at least one of those 5 "if" statements must have a nested "if". It is also acceptable to use "switch" statements in place of the "if" statements.**

Notice that these requirements leave lots of room for creativity! Try to create an interesting game that you think people would enjoy playing.

(10 pts) Extra Credit

Continue to ask the user if they want to play the adventure game again and repeat until the user decides not to play anymore.

(10 pts) Program Style/Comments

In your implementation, make sure that you include a program header in your program, in addition to proper indentation/spacing and other comments! Below is an example header to include. Make sure you review the **style guidelines for this class (https://web.engr.oregonstate.edu/~goinsj/resources/general/cpp_style_guideline.pdf)**, and begin trying to follow them, i.e. don't align everything on the left or put everything on one line!

```
/******************************************************
** Program: adventure.cpp
** Author: Your Name
** Date: 10/02/2018
** Description:
** Input:
```

```
** Output:
*******************************************************/
```

Reminder

Every assignment in this course is graded by demoing your work for 10 minutes with a TA. You are required to **meet with a TA within two weeks of the due date** to demo. You can schedule a demo with a TA from the TA Office Hours tab in Canvas. The available times can be viewed in the far right column of the table (labeled "Grading Hours"). Click on one of the links to access the poll and insert your name in a specific time slot.

- **Demo Outside 2 Weeks**: Assignments that are not demo'd within the acceptable time period will be subject to a 50 point deduction.
- **Demo Late Assignments**: Late assignments must still be demoed within the two week demo period beginning from the assignment's original due date.
- **Missing a Demo**: If you miss your demo with a TA, you will receive a 10 point (one letter grade) deduction to that assignment for each demo missed.

Each program needs to be written according to the style guidelines (https://web.engr.oregonstate.edu/~goinsj/resources/general/cpp_style_guideline.pdf) for this class. Remember that an important part of computer programming is making sure that your work is easily understandable by other programmers.

Electronically submit your C++ source code and your design documents by the assignment due date, using TEACH (https://engineering.oregonstate.edu/teach). You may use one of the document scanners (available in KEC1130 and some other College of Engineering labs) to scan your paper design into a PDF document that can be submitted to TEACH (https://engineering.oregonstate.edu/teach). Your submission will have two files: a .cpp file and a .pdf file containing your design paperwork.

Program #2 Rubric

Criteria	Ratings	Pts
Understand the Problem Stated questions and inputs/outputs, not just restate problem		5.0 pts
Program Design Some kind of paragraph, pseudocode, and/or flowchart describing the text adventure paths		10.0 pts
Testing Table Good values, 5 pts Bad values, 5 pts		10.0 pts
Verify the user input Get the user input (5 pts) Check if it is bad (not one of the valid choices) (10 pts) Note that the student only has to handle bad "numeric" input. e.g. -1, -1234, 3, 400, etc		15.0 pts
Have the student show you how they used at least 5 different if (or switch) conditions. (each is worth 5 pts)		25.0 pts
Have the student show you where and how they implemented an element of chance for one decision. (10 pts for rand, 5 pts for srand)		15.0 pts
Show that different user input yields different results The code uses at least one nested if statement (or nested switch)		10.0 pts
Program Style: Program Header/Comments, 5 pts Proper Indentation, 5 pts		10.0 pts
Extra Credit Did they use a loop to repeat the adventure (or any loop)? 10 pts		0.0 pts
		Total Points: 100.0