COMSC 165

Summer 2020

Programming Assignment 2

Worth 20 points (2% of your grade)

DUE: Wednesday, 6/24/20 by 11:59 P.M. on Canvas

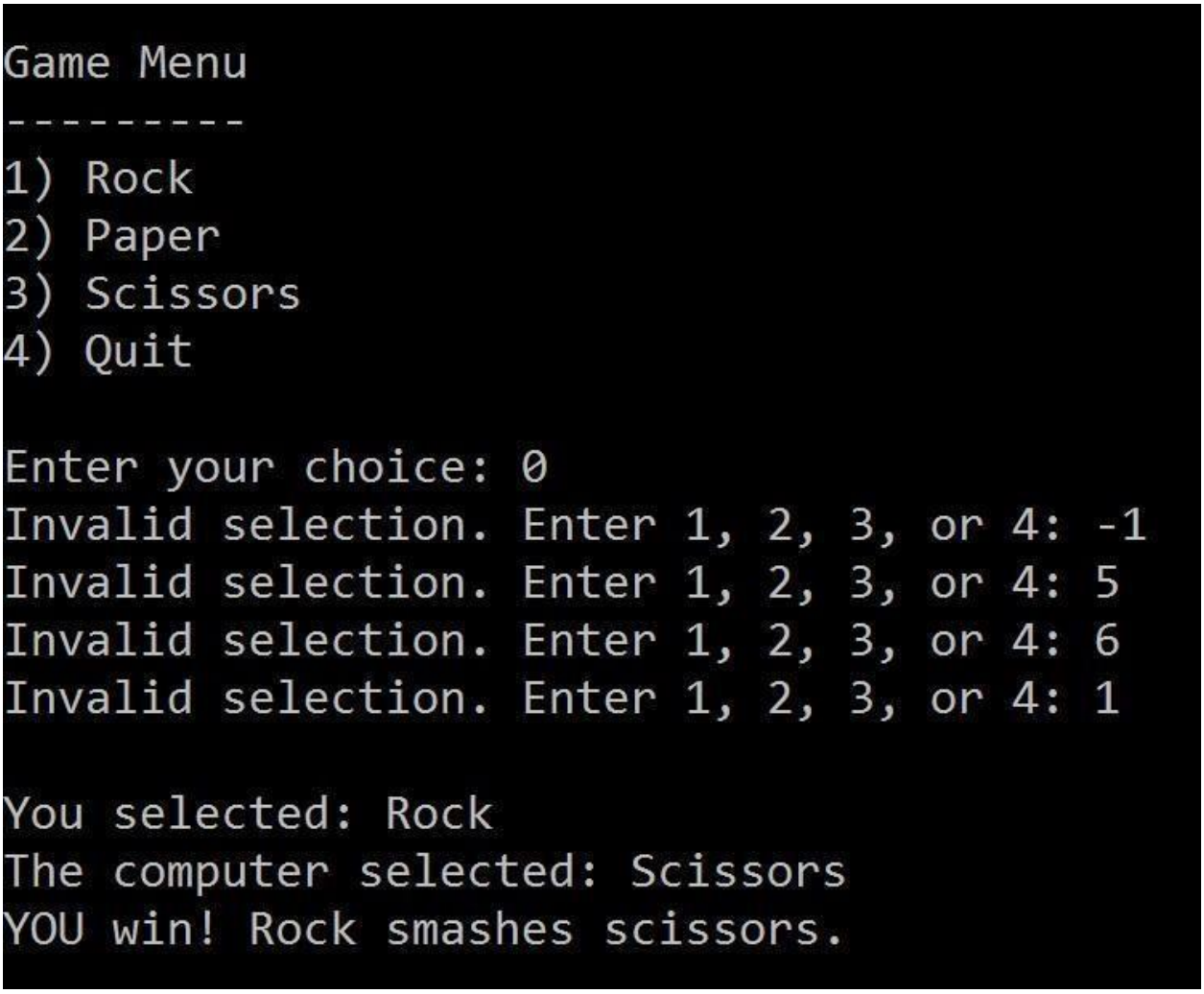
Start by downloading the **165_assign2.cpp** file from the Programming Assignment 2 folder on Canvas

NOTE: Your submission for this assignment should be a single **.cpp** file and a single **.pdf** file. The following naming convention should be used for naming your files: **firstname_lastname_165_assign2.cpp and firstname_lastname_165_assign2.pdf**. The pdf file that you submit should contain the screenshots of your sample runs of the program (see below). For example, if your first name is "James" and your last name is "Smith", then your files should be named James_Smith_165_assign2.cpp James_Smith_165_assign2.pdf.

COMMENTS: Your program should have at least **ten (10)** different detailed comments explaining the different parts of your program. Each individual comment should be, at a minimum, a sentence explaining a particular part of your code. You should make each comment as detailed as necessary to fully explain your code. You should also number each of your comments (i.e., comment 1, comment 2, etc.).

SAMPLE RUNS: You should submit screenshots of at least **five (5)** different sample runs of your program. Each sample run needs to use different inputs, and your sample runs should **NOT** be the same as the sample runs that are used in this write-up for the assignment. You should also number each of your sample runs (i.e., sample run 1, sample run 2, etc.).

Since the computer's choice is randomly determined, you don't have complete control over the output that you get. So in each of your sample run screenshots, you should **first** purposefully enter invalid inputs, so that each of your five sample runs are different from each other (**AND** different from all of my sample runs in this write-up). For each sample run, put in some invalid inputs (which will be handled by your input validation loop) **before** entering a valid input. **All** of your sample runs should follow this format, where invalid inputs are entered **before** a valid input is entered:



Where in each sample run, you need to put in **different** invalid inputs before a valid input (1-4) is entered. And between all five of your sample runs, the user should choose each option from the menu at least one time. The quit option should be chosen once, and one of rock, paper, or scissors should be chosen twice.

For your second programming assignment you will be writing a C++ program that implements the game of Rock, Paper, Scissors using **functions**. The game has one human player (you!) against a computer.

The program should follow these steps:

1. When the program begins, a **random number** in the range of 1 through 3 is generated. If the number is 1, then the computer has chosen rock. If the number is 2, then the computer has chosen paper. If the number is 3, then the computer has chosen scissors. (**Don't display the computer's choice before the user has played!!!)**

2. The following menu is displayed to the user:

   Game Menu

   1) Rock
   2) Paper
   3) Scissors
   4) Quit

3. The user is prompted for their choice. If they enter 1, their choice is rock. If they enter 2, their choice is paper. If they enter 3, their choice is scissors. If they enter 4, then the game ends.

Input validation loop: If the user enters anything other than 1-4, the menu should come back up again until the user enters something valid.

4. The user's choice is displayed.

5. The computer's choice is displayed.

6. The outcome of the game is determined according to the following rules:

A. If one player chooses rock and the other player chooses scissors, then rock wins. (The rock smashes the scissors.)
B. If one player chooses scissors and the other player chooses paper, then scissors wins. (Scissors cuts paper.)
C. If one player chooses paper and the other player chooses rock, then paper wins. (Paper wraps rock.)
D. If both players make the same choice, then a tie has occurred.

Also make sure to **seed** the random number generator, so the program doesn't generate the same sequence of random numbers each time it is run.

The main function has already been completed for you. In addition, the function prototypes and function headers have also been completed for you. You will be implementing the body of the following functions:

int getComputerChoice()
INPUT(S): No input parameters
RETURNS: The computers choice (1, 2, or 3)

PURPOSE:
 	
// **********
// The getComputerChoice function returns the computer's *
// game choice. It returns 1 for rock (via the ROCK *
// constant), or 2 for paper (via the PAPER constant), *
// or 3 for scissors (via the SCISSORS constant). *
// **********

int getUserChoice()

INPUT(S): No input parameters

RETURNS: The user's choice (1, 2, 3, or 4)

PURPOSE:

```
// ******************************************************
// The getUserChoice function displays a menu allowing    *
// the user to select rock, paper, or scissors. The       *
// function then returns 1 for rock (via the ROCK         *
// constant), or 2 for paper (via the PAPER constant),    *
// or 3 for scissors (via the SCISSORS constant),         *
// or 4 for quit (via the QUIT constant).                 *
// ******************************************************
```

void displayChoice(int choice)
INPUT(S): The user's or computers choice (1, 2, or 3)
RETURNS: Nothing is returned

PURPOSE:
 	
// **
// The displayChoice function accepts an integer argument *
// and displays rock, paper, or scissors. *
// **

void determineOutcome(int user, int computer)
INPUT(S): The user's choice (1, 2, or 3) and the computers choice (1, 2, or 3)
RETURNS: Nothing is returned

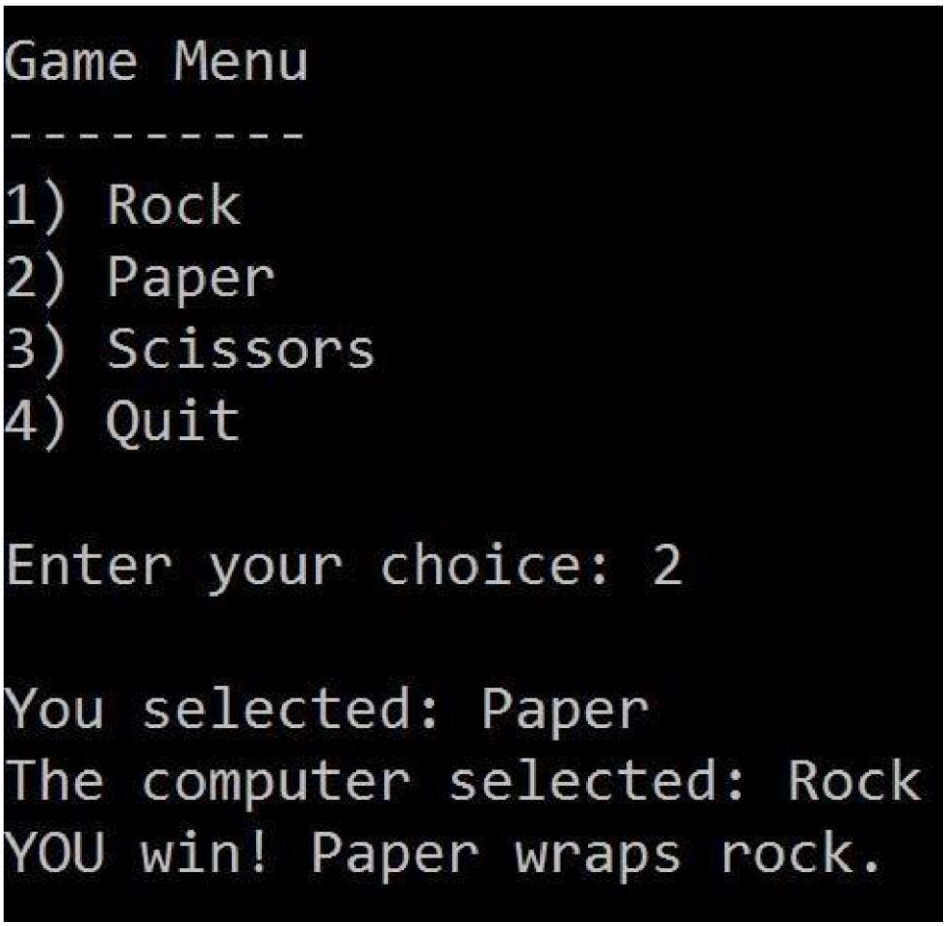
PURPOSE:
 	
// **
// The determineOutcome function accepts the user's game * // choice and the computer's game choice as arguments and * // displays a message indicating the winner *
// or that a tie occurred. *
// **

Sample Runs:

```
Game Menu
---------
1) Rock
2) Paper
3) Scissors
4) Quit

Enter your choice: 1

You selected: Rock
The computer selected: Scissors
YOU win! Rock smashes scissors.

Game Menu
---------
1) Rock
2) Paper
3) Scissors
4) Quit

Enter your choice: 2
```

```
You selected: Paper
The computer selected: Scissors
Computer wins! Scissors cuts paper.
```

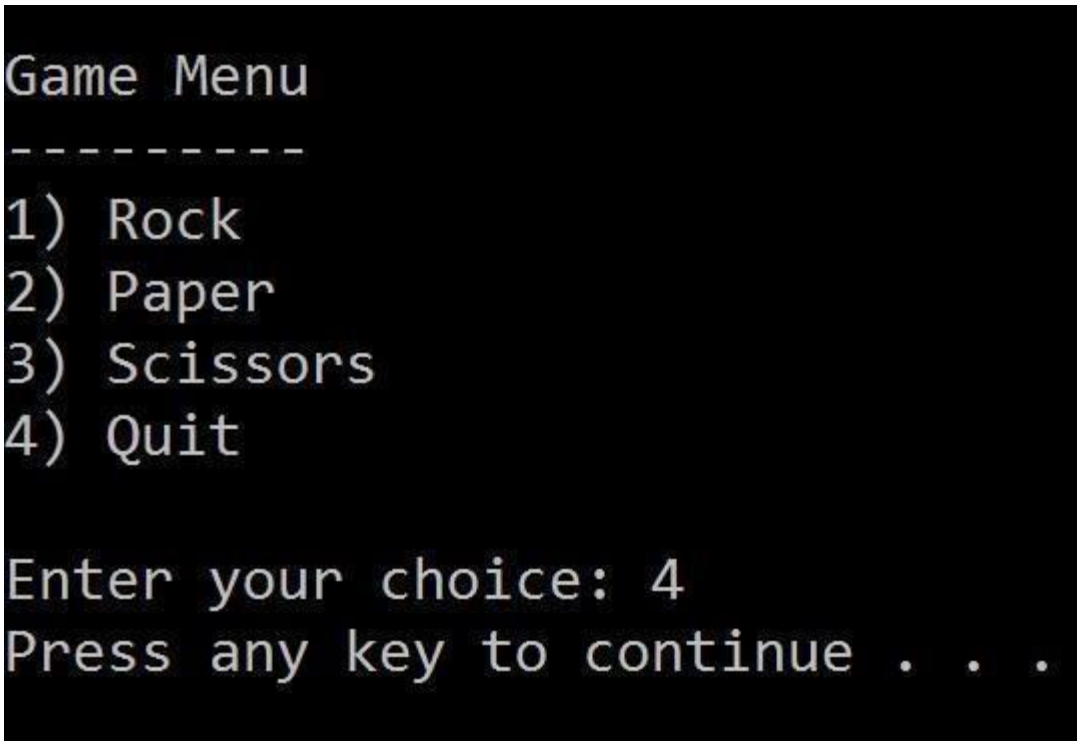
```
Game Menu
---------
1) Rock
2) Paper
3) Scissors
4) Quit

Enter your choice: 3

You selected: Scissors
The computer selected: Scissors
Tie. No winner.
```


Game Menu
---
1) Rock
2) Paper
3) Scissors
4) Quit
Enter your choice: 2
You selected: Paper
The computer selected: Rock
YOU win! Paper wraps rock.

```
Game Menu
---------
1) Rock
2) Paper
3) Scissors
4) Quit

Enter your choice: 3

You selected: Scissors
The computer selected: Rock
Computer wins! Rock smashes scissors.
```

```
Game Menu
---------
1) Rock
2) Paper
3) Scissors
4) Quit

Enter your choice: 1

You selected: Rock
The computer selected: Rock
Tie. No winner.
```

```
Game Menu
---------
1) Rock
2) Paper
3) Scissors
4) Quit

Enter your choice: 0
Invalid selection. Enter 1, 2, 3, or 4: -1
Invalid selection. Enter 1, 2, 3, or 4: 5
Invalid selection. Enter 1, 2, 3, or 4: 6
Invalid selection. Enter 1, 2, 3, or 4: 1

You selected: Rock
The computer selected: Scissors
YOU win! Rock smashes scissors.
```


Game Menu
----
1) Rock
2) Paper
3) Scissors
4) Quit
Enter your choice: 4
Press any key to continue . . .