CIS 246 – C++ Programming

Program:	**Week 06**
Points:	**20**
Chapter(s):	**6**
File(s) to Submit:	**RPSDriver.cpp, Player.h (as one zip file)**

Summary

Write a C++ application that uses overloaded functions and random number generation to simulate a game of rock, paper, scissors between a player and the computer.

Extra credit is available for this assignment.

Description

Write a driver program that creates two objects of the Player class. One object represents the player, and the other object represents the computer.

Prompt the player to type in their play. Generate a random number to determine the computer's play. Determine the winner and print the results to the screen.

Learning Objectives

In this assignment, you will practice:

- Writing overloaded functions
- Using overloaded functions
- Using random numbers

Requirements for the Player Class

Data Members

play (as an integer)

This doesn't need to be initialized. For the purposes of the game, a value of 1 means rock, 2 means paper, and 3 means scissors.

Member Functions

1. A function to return the value of the integer data member **play** as a **string return value.**

   Don't just return the integer value. You'll have to return the string "rock," "paper," or "scissors," depending on the value of the **play** data member. Use all lowercase.

2. An overloaded function called **setPlay** to set the value of **play** using a **string parameter**.

The argument will be one of the values “rock,” “paper,” or “scissors.” You can assume the word will be spelled correctly and will be in all lowercase.

3. An overloaded function called **setPlay** to set the value of **play** using an **int parameter**. This is just a regular setter function.

No constructor is needed for this class.

Requirements for the RPSDriver Program

1. **Create two objects of the Player class, for the player and the computer.**

2. **Prompt the player to type in their play as a STRING.**

   Plays are the values “rock”, “paper”, or “scissors”. Call the appropriate overloaded set function in the player object using whatever value is typed. (Remember, you don’t have to use selection to do this when using overloaded functions.) You do not have to check for spelling mistakes or values other than rock, paper, or scissors. Assume the words will be spelled correctly and in all lowercase.

3. **Generate a random number to determine the computer’s play.**

   Generate a random number from the set {1, 2, 3}. Call the appropriate overloaded set function in the computer object using the value generated.

4. **Determine the winner.**

   Use the get function from each object to retrieve the play for each object and use selection to determine the winner of the game, using standard rock, paper, scissors logic.

5. **Display the result to the screen.**

Extra Credit #1 – Maximum 10 Points

Give the user the option to play the game by typing in the words rock, paper, or scissors, or by typing numbers to represent the plays.

1. Add another function to the Player class to return the play as an **int** return value.

2. In **RPSDriver**, move the logic to determine the winner into **two overloaded functions**. One function must use two string parameters, and one must use two integer parameters. In both cases, the parameters are the player’s and computer’s plays. Remember to declare these functions using the **static** keyword.

3. Call the appropriate overloaded function from main using the appropriate functions of the Player class as arguments. **This function call must contain two other function calls as arguments.**

Extra Credit #2 – Maximum 10 Points

You must implement Extra Credit Option #1 first.

Replace the two overloaded functions in RPSDriver that determine the winner with **one template function** that determines the winner.

Prototypes are not used with template functions, so this function must be defined above main.

Requirements for Full Credit on This Project

SUBMIT YOUR OWN WORK – Plagiarism is not tolerated in this course. Please review the section on the Academic Honor Code in the syllabus. I will not hesitate to drop you from this class if you submit a program that is plagiarized.

COMPLETE AND ACCURATE – Your program must compile, execute, and give accurate output.

FOLLOW ALL REQUIREMENTS ACCORDING TO THE INSTRUCTIONS – Follow the instructions as written for completing this project, even if you [think you] know a "better" way to do something.

COMMENTS – Include comments in your code. There must be a comment at the top of each source code or header file that includes your name, the assignment number, and a description of the code in that file. There must be comments at each important step in your algorithm that describes that step.

BEST PRACTICES – Follow best practices in C++ programming, including, but not limited to, appropriate use of private/public, appropriate use of classes and/or header files, sets & gets, white space, alignment, meaningful variable names, naming conventions, using statements, etc. Points will be deducted for sloppy code that is hard to read, even if it works, so pay attention to these details.

SUBMIT ALL FILES BEFORE THE DUE DATE – See the due date for this assignment on the course calendar in Canvas and review the submission requirements below.

ALL SUBMISSIONS MUST BE YOUR OWN WORK – Review the syllabus regarding plagiarism and the Joliet Junior College Academic Honor Code. Review the Get Help page in every module if you need help with this assignment.

Submission Requirements

All source code file submissions must be compressed into a Compressed Folder, or zip file. The dropbox for this assignment will only allow compressed files (.zip extension) to be uploaded.

Instructions for creating a Compressed Folder (zip file) are available here:
https://support.microsoft.com/en-us/help/14200/windows-compress-uncompress-zip-files

Do not submit your code in any other file format but .cpp and .h (within a zip). I will not accept links to online storage. You must submit the actual file. Do not submit executable files. Do not submit project files from an IDE.

Sample Runs

Remember you are generating the computer's plays randomly, so it is unlikely that your output will duplicate this output exactly.

Sample Run – Required Program

```
Welcome to Rock Paper Scissors

Type rock, paper, or scissors: paper

You played: paper
Computer played: paper

It's a tie
```

Sample Run 1 – Extra Credit #1

```
Welcome to Rock Paper Scissors

1. Play by typing out the words
2. Play by typing numbers
3. Exit

What is your choice? 1

Type rock, paper, or scissors: paper

You played paper
Computer played rock

Player wins
```

Sample Run 2 – Extra Credit #1

```
Welcome to Rock Paper Scissors

1. Play by typing out the words
```

```
2. Play by typing numbers
3. Exit

What is your choice? 2

Type 1 for rock, 2 for paper, 3 for scissors: 1

You played rock
Computer played paper

Computer wins
```

Sample Run – Extra Credit #2

No difference in output from the version for Extra Credit #1