CS 3 Assignment 4 - Conditional Execution

**Programming Assignment 4:**Individual Assignment. **You must complete this assignment by yourself. You cannot work with anyone else in the class or with someone outside of the class. You may not copy solutions from the world wide web. You are encouraged to get help from the instructional staff.**

General Assignment Requirements

The purposes of this assignment are:

1. To practice creating a structured program to solve a problem.
2. To write methods that use parameters.
3. To write an interactive program.
4. To practice writing programs that use conditional execution.

For this assignment you are limited to the language features in chapters 1 through 4 of the textbook and the logical operators from section 5.3. (&& -logical AND, || - logical OR, and ! -  logical NOT)

**TIP: This program is in some ways more difficult that the first three. You must write methods that return values, get input from the user with Scanner, and use conditional logic (if - else).**

**Description:**Write a program to allow a human to play a game of Rock, Paper, Scissors versus the computer. The sample programs ([rps1.txt](https://www.cs.utexas.edu/~scottm/cs312/javaCode/A4_RockPaperScissors/rps1.txt), [rps2.txt](https://www.cs.utexas.edu/~scottm/cs312/javaCode/A4_RockPaperScissors/rps2.txt), and [rps3.txt](https://www.cs.utexas.edu/~scottm/cs312/javaCode/A4_RockPaperScissors/rps3.txt)) show 3 runs from my solution to this problem using the default computer player. You must use the given computer player class so the choices are predictable. This is necessary to that we can grade your program. If you were writing the program for yourself you would of course make the computer choices random. **But the program you turn in must use the predictable choices generated by the given computer player class.**

Given the same human name and computer choices, your program must match this output **exactly**.

The program:

* asks the user for their name
* asks the user how many rounds of Rock, Paper, Scissors they want to play
* plays that many rounds of the game
* for each round of the game
  + asks the user for their choice
  + have the computer make a random choice
  + prints out each player's choice
  + prints the results of the round
* after playing the specified number of rounds, displays the number or rounds the user won, the number of rounds the computer won, and the number of rounds that were a draw
* declares who the better player was based on the number of wins

This is not an easy program, mostly due to the size of the program. The individual steps are not too difficult, but their are many steps. The program description above gives you a rough idea of how to break the program up into parts.

Have a high level structure and then implement parts of that structure (the individual methods) one at a time, testing to make sure they work before going on. You may have to write some testing code that will not be part of the final program. **Do not write the whole program in main and then try and break it up into methods.**

Here are some tips on the various parts of the program.

**1. main method.**

The main method creates the an object of type RandomPlayer. If no values are sent to main a default RandomPlayer is created. If you send an argument to the main method it is assumed it is a single value that can be parsed to an int. (These pages describe how to send an argument to the main method in BlueJ or Eclipse). Pass the RandomPlayer object to the methods that need it. Feel free to share examples of your output for non default players on Piazza.

In the main program declare a Scanner variable that is hooked up to System.in. You must include the line of code

import java.util.Scanner;

at the top of your program.

Pass the Scanner object you create as a parameter to any methods that need it. The main method should not have a lot of statements, instead it shall call other methods. **Do not create multiple Scanners. Create one Scanner connected to System.in and pass that object to the methods that need it. If you create more than one Scanner in your program connected to System.in (Scanner sc = new Scanner(System.in);) you will fail, many many of our tests and lose a lot of correctness points.**

**2. ask the user for their name.**This is a good candidate for a separate method that returns a String.

**3. ask the user how many rounds of Rock, Paper, Scissors they want to play.**This is another good candidate for a separate method that returns an int. **You do not have to do any error checking on the user input. If they enter something that is not an int it is appropriate for the program to end due to a runtime error.**

**4. playing the rounds of the game** Given our current programming tools this will be the largest and most complex method. It is in turn broken down into several parts. You will need a number of local variables in this method.

**5. ask the user what their choice is.**The user will enter an integer as their choice. **You do not need to error check their input.**

**6. have the computer make a random choice.**To do this call the getComputerChoice method on the RandomPlayer object created in main and passed as a parameter.

**You must use the provided RandomPlayer.java file. Do not change it.. Do not use Random() (from java.util) directly and do not use Math.random().  Keep RandomPlayer.java in a separate file. Do not turn in RandomPlayer.java.**

**7. print out each player's choice**. You will find it useful to have a method that is passed an  int parameters and returns the correct String for that int. In this program 1 represents "Rock", 2 represents "Paper", and 3 represents "Scissors".

**8. print the results of that round.** This is the most algorithmically difficult part of the assignment because there are nine possible outcomes and using the programming tools of chapters 1 - 4 and section 5.3 it is difficult to remove redundancy.

The nine possible outcomes are

|  |  |  |
| --- | --- | --- |
| Computer Choice | Human Choice | Result |
| Rock | Rock | Draw |
| Rock | Paper | Human Wins |
| Rock | Scissors | Computer Wins |
| Paper | Rock | Computer Wins |
| Paper | Paper | Draw |
| Paper | Scissors | Human Wins |
| Scissors | Rock | Human Wins |
| Scissors | Paper | Computer Wins |
| Scissors | Scissors | Draw |

**You must follow the format as shown in the sample output.**

**9. after playing the specified number of rounds display how time the user won, how many times the computer won, and how many draws occurred.** The method that runs the rounds shall call a method to display this information.

**10. declare who the better player was based on the number of wins.** This can be part of the results method but will require some conditional execution with if statements. \

**Additional tips.**You may find it useful to have class constants for the integers representing rock, paper, and scissors. Recall class constants are declared at the top of the class outside any methods:

public static final int ROCK = 1;

**Style issues. We will grade program hygiene as well as correctness. Did you provide a good structure to the program using static methods? Did you minimize the scope of variables to the smallest necessary? Did you use meaningful identifiers? Did you provide consistent tabbing and spacing for code inside loops and if statements? Did you provide comments for you methods and your code?**

When finished turn in your RockPaperScissors.java file via Canvas.

|  |  |
| --- | --- |
| Provided File | Responsibility |
| [RandomPlayer.java (Provided file)](https://www.cs.utexas.edu/~scottm/cs312/javaCode/A4_RockPaperScissors/RandomPlayer.java) Used to provide computer pick. Can be set to predictable for testing purposes. | Provided by me. Do not turn in. Do not alter. |
| [RockPaperScissors.java (Provided shell)](https://www.cs.utexas.edu/~scottm/cs312/javaCode/A4_RockPaperScissors/RockPaperScissors.java) | You and me. (Okay, mostly you.) |

**Checklist:**Did you remember to:

* review the general assignment requirements?
* worked on the assignment by yourself?
* fill in the header in your file RockPaperScissors.java?
* complete the RockPaperScissors.java so it plays the game correctly?
* use good names for variables to make your program easy to understand?
* create a single Scanner object connected to System.in and pass as a parameter as necessary?
* ensure your program does not suffer a compile error or runtime error (unless the runtime error is caused by bad user input)?
* ensure your output matches the sample files exactly given the same input?
* turn in your Java source code in files named RockPaperScissors.java via canvas