**Python Journal Template**

**Directions: Follow the directions for each part of the journal template. Include in your response all the elements listed under the Requirements section. Prompts in the Inspiration section are not required; however, they may help you to fully think through your response.**

**Remember to review the Touchstone page for entry requirements, examples, and grading specifics.**

**Name: Rachel Cole**

**Date: 1/29/2022**

**Final Replit Program Share Link:** <https://replit.com/join/qwaheneush-rachelcole45>

**Complete the following template. Fill out all entries using complete sentences.**

**PART 1: Defining Your Problem**

**Task**

State the problem you are planning to solve.

**Requirements**

* Describe the problem you are trying to solve for.
* Describe any input data you expect to use.
* Describe what the program will do to solve the problem.
* Describe any outputs or results the program will provide.

**Inspiration**

When writing your entry below ask yourself the following questions:

* Why do you want to solve this particular problem?
* What source(s) of data do you believe you will need? Will the user need to supply that data, or will you get it from an external file or another source?
* Will you need to interact with the user throughout the program? Will users continually need to enter data in and see something to continue?
* What are your expected results or what will be the end product? What will you need to tell a user of your program when it is complete?

The game of Rock, Paper, Scissors is a popular game of luck. It is often used to objectively decide something, similar to flipping a coin. The problem to solve is to create a fair game of rock, paper, scissors that the person plays against a computer which will randomly choose one of the three options. I will create a simulation of the game. First, the game will ask for the player’s name. Next, the program will ask for an input of r for rock, p for paper, and s for scissors. The computer will then generate a random number 1-3 that represents rock, paper, and scissors. Using a series of if and else if statements, it will be determined whether the player beat the computer, tied, or lost. At the end, the program will ask if the player would like to play again.

**PART 2: Working Through Specific Examples**

**Task**

Write down clear and specific steps to solve a simple version of your problem you identified in Part 1.

**Requirements**

Complete the three steps below **for at least two distinct examples/scenarios**.

* State any necessary input data for your simplified problem.
* Write clear and specific steps in English (not Python) detailing what the program will do to solve the problem.
* Describe the specific result of your example/scenario.

**Inspiration**

When writing your entry below ask yourself the following questions:

* Are there any steps that you don’t fully understand? These are places to spend more time working out the details. Consider adding additional smaller steps in these spots.
* Remember that a computer program is very literal. Are there any steps that are unclear? Try giving the steps of your example/scenario to a friend or family member to read through and ask you questions about parts they don’t understand. Rewrite these parts as clearly as you can.
* Are there interesting edge cases for your program? Try to start one of your examples/scenarios with input that matches this edge case. How does it change how your program might work?

Scenario 1:

1. Player chooses r, p, or s
2. Computer chooses the same option as the player
3. Program displays that it is a tie
4. Program asks to play again

Scenario 2:

1. Player chooses p
2. Computer randomly chooses r
3. Program displays that player has won
4. Program asks to play again

Scenario 3:

1. Player chooses r
2. Computer randomly chooses p
3. Program displays that player has lost
4. Program asks to play again

Scenario 4:

1. Player inputs a key other than r, s, or p
2. Program loops back and asks player to input again
3. They enter the correct key
4. Displays whether they won, lost, or tied
5. Program asks to play again

**PART 3: Generalizing Into Pseudocode**

**Task**

Write out the general sequence your program will use, including all specific examples/scenarios you provided in Part 2.

**Requirements**

* Write pseudocode for the program in English but refer to Python program elements where they are appropriate. The pseudocode should represent the full functionality of the program, not just a simplified version. Pseudocode is broken down enough that the details of the program are no longer in any paragraph form. One statement per line is ideal.

**Help with writing pseudocode**

* Here are a few links that can help you write pseudocode with examples. Remember to check out part 3 of the Example Journal Template Submission if you have not already. Note: everyone will write pseudocode differently. There is no right or wrong way to write it other than to make sure you write it clearly and in as much detail as you can so that it should be easy to convert it to code later.
  + <https://www.geeksforgeeks.org/how-to-write-a-pseudo-code/>
  + <https://www.wikihow.com/Write-Pseudocode>

**Inspiration**

When writing your entry below ask yourself the following questions:

* Do you see common program elements and patterns in your specific examples/scenarios in Part 2, like variables, conditionals, functions, loops, and classes? These should be part of your pseudocode for the general sequence as well.
* Are there places where the steps for your examples/scenarios in Part 2 diverged? These may be places where errors may occur later in the project. Make note of them.
* When you are finished with your pseudocode, does it make sense, even to a person that does not know Python? Aim for the clearest description of the steps, as this will make it easier to convert into program code later.

Import necessary modules

set the game status to true for the while loop

While the status of the game is equal to True:

Print “Welcome to the game of Rock, Paper, Scissors…”

Wait one second

Name equals the input of “Please enter your name”

Wait one second

Print “Alrighty” + name of player + “let’s get started”

Wait one second

Set the statusInput equal to True

While statusInput is equal to true:

Set player input equal to input of “Enter r for rock, p for paper, or s for scissors”

If player equals r

playerChoice = 1

Statusinput equals false

Else If player equals p

playerChoice = 2

Status input equals false

Else If player equals s

playerChoice equals 3

Status input equals false

Else if it does not equal r, p, or s:

Statusinput equals true

Print “not a valid input”

Print “You chose” + input from what player selected

Wait one second

Computerchoice equals random integer between 1 and 3

If computerchoice is 1

Print “your opponent chose rock”

Else if computer choice is 2

Print “your opponent chose paper

Else if computerchoice is 3

Print “your opponent chose scissors”

If playerChoice = computerChoice :

Print “it’s a tie!”

Else If playerChoice = 1 AND computerChoice = 2 :

Print “oops, you lost this one!”

Else If playerChoice = 1 AND computerChoice = 3 :

Print “name wins!”

Else If playerChoice = 2 AND computerChoice = 1 :

Print “name wins!”

Else If playerChoice = 2 AND computerChoice = 3 :

Print “oops, you lost this one”

Else If playerChoice = 3 AND computerChoice = 1 :

Print “oops, you lost this one”

Else If playerChoice = 3 AND computerChoice = 2 :

Print “name wins!”

playAgain equals input name + “would you like to play again? Y for yes, n for no”

If playAgain equals y

statusGame equals True

Print “new game”

If playAgain equals n

statusGame equals false

Print “bye for now, game over”

Else

Print “invalid input game over”

**PART 4: Testing Your Program**

**Task**

While writing and testing your program code, describe your tests, record any errors, and state your approach to fixing the errors.

**Requirements**

* For at least one of your test cases, describe how your choices for the test helped you understand whether the program was running correctly or not.

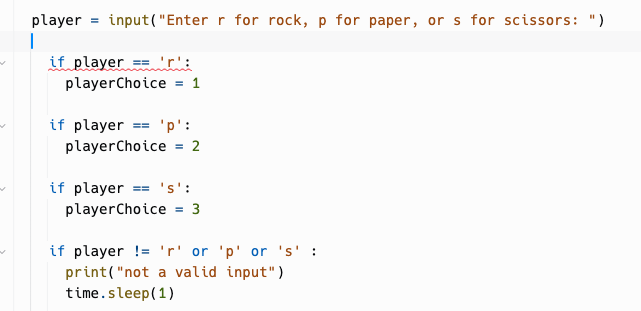
For each error that occurs while writing and testing your code:

* Record the details of the error from Replit. A screenshot or copy-and-paste of the text into the journal entry is acceptable.
* Describe what you attempted in order to fix the error. Clearly identify what approach was the one that worked.

**Inspiration**

When writing your entry below ask yourself the following questions:

* Have you tested edge cases and special cases for the inputs of your program code? Often these unexpected values can cause errors in the operation of your program.
* Have you tested opportunities for user error? If a user is asked to provide an input, what happens when they give the wrong type of input, like a letter instead of a number, or vice versa?
* Did the outcome look the way you expected? Was it formatted correctly?
* Does your output align with the solution to the problem you coded for?



First, I wanted this code to start over if the player entered an invalid input. I decided to put the code in a while loop. This meant I had to create a variable called statusInput and set that equal to true. Then while that was true, if they entered a valid input, it turned to false and would exit the loop, but if they entered an invalid input, it would stay true, say “not a valid input” and return to the top of the loop.

I also had an issue in that whichever key I input, it would say it was invalid. I fixed this by changing all the if’s after the first one to else if or elif statements. This fixed the problem and it started working again.



(This screenshot is from after the code was fixed/altered) I had a previous issue here that i wanted the random integer to represent a letter, because when i tried to do print(input(“your opponent chose” + random.randint(1,3)) it wouldn’t let me concatenate an integer. So I was going to convert it to a string with the str() function, but it was still just printing the number. I decided to just ditch that idea, and just print what the computer chose with a full string and no concatenation.



This screenshot is from the end of the entire while loop (the whole game is a while loop in order for it to restart at the top according to player’s input). Here I had to make sure I made a case for when the player inputs an invalid key. I could have created another while loop so it would ask again Y or N if they wanted to play again, but I decided to switch it up and use an else statement at the bottom. If the user inputs something invalid, it will just exit the game and be game over for them.

My main issues were with invalid inputs, while loops, if statements, and figuring out how to deal with user inputs. The while loops were difficult because I had to create a way to get out of them, which is why I created new variables and set them equal to True, and if there was a valid input, they were set equal to False so the loop didn’t start over and the code moved onto the next portion.

The if statements were confusing because they worked until I put them in while loops. When they were in the loops, no matter what I input, it would stay in the loop. I soon realized I needed to do else if statements.

Dealing with the user inputs was difficult as well. As I mentioned previously, I wanted the integers 1, 2, and 3 to represent r, p, and s respectively, but I would have had to make the integers a string and then convert them and it became very complicated very quickly. In the end I learned a lot working on my own project and this was definitely the best way for me to learn. I still have a long way to go, but this helped me tremendously and I will continue working on personal projects because I now see this is the best way for me to retain the information and figure out the code and language.

**PART 5: Commenting Your Program**

**Task**

Submit your full program code, including thorough comments describing what each portion of the program should do when working correctly.

**Requirements**

* The purpose of the program and each of its parts should be clear to a reader that does not know the Python programming language.

**Inspiration**

When writing your entry, you are encouraged to consider the following:

* Is each section or sub-section of your code commented to describe what the code is doing?
* Give your code with comments to a friend or family member to review. Add additional comments to spots that confuse them to make it clearer.

#This game is Rock Paper Scissors, a game typically used as a fair method of deciding between options

#Rock beats scissors, scissors beats paper, and paper beats rock

#import time module to be able to wait in between print statements

#import random module to generate a random integer for rock paper or scissors for computer opponent

import time

import random

#for the while loop, we need to set the game status to true so when a condition in the loop equals false, it goes out of the loop. But if it is true, it stays in the loop and asks for another input

statusGame = True

#start a while loop for the entire game and welcome player to game

while statusGame == True :

print("Welcome to the legendary game of Rock Paper Scissors")

#these time.sleep methods tell the code to wait 2 seconds (or whichever number is in the parentheses) until the next line of code. this is so the player can read what is printed and not feel pressed to hurry.

time.sleep(2)

#asking user to input their name and storing this in the variable called name

name = input("Please enter your name:")

time.sleep(.7)

#using the variable name, we tell user to get ready to play the game

print("Alrighty " + name + " buckle up and get ready for an intense game...")

time.sleep(1.5)

#need to set this variable equal to true, same reason as before... for the while loop to restart if it remains true.

statusInput = True

#beginning of a while loop

while statusInput == True:

#setting player variable equal to the input of rock, paper, or scissors

player = input("Enter r for rock, p for paper, or s for scissors: ")

#setting the user inputs equal to a number because later we will generate a random number 1-3 to represent computer's random choice of rock, paper, or scissors. Notice also the status input line in each if statment, if it is false, it will exit the code. If true, it will start the loop over and ask for another input.

#if the player enters r, then make the playerChoice variable equal to 1 and set the statusInput equal to False, thus exiting the code. For the next statements, this is repeated but with other inputs

if player == 'r':

playerChoice = 1

statusInput = False

elif player == 'p':

playerChoice = 2

statusInput = False

elif player == 's':

playerChoice = 3

statusInput = False

#if the user input is NOT r, p, or s, the statusInput is true which starts the while loop over. they will be told this was not a valid input and when the loop begins, it will again ask for r, p, or s

elif player != 'r' or 'p' or 's' :

statusInput = True

print("not a valid input")

time.sleep(1)

time.sleep(1)

#telling user what they chose

print("You chose: " + player)

time.sleep(1)

#this random.randit method will choose 1, 2, or 3 randomly. This is the opponent's (copmuter's) choice of rock, paper, or scissors.

computerChoice = random.randint(1,3)

#here we are telling the user what the computer chose. 1 is rock, 2 is paper, 3 is scissors.

if computerChoice == 1:

print("your opponent chose rock...")

elif computerChoice == 2:

print("your opponent chose paper...")

elif computerChoice == 3:

print("your opponent chose scissors...")

time.sleep(2)

#here are all of the possibilities of the outcomes.

#if equal, it will be a tie

if playerChoice == computerChoice :

print("it's a tie!")

#if player chooses rock and computer chooses paper, player loses

elif playerChoice == 1 and computerChoice == 2 :

print("oops, " + name + " you lost this one :/")

#if player chooses rock and computer chooses scissors, player wins

elif playerChoice == 1 and computerChoice == 3 :

print(name + " wins! congrats champ!")

#if player chooses paper, and computer chooses rock, player wins

elif playerChoice == 2 and computerChoice == 1 :

print(name + " wins! congrats champ!")

#if player chooses paper and computer chooses scissors, player loses

elif playerChoice == 2 and computerChoice == 3 :

print("oops, " + name + " you lost this one :/")

#if player chooses scissors and computer chooses rock, player loses

elif playerChoice == 3 and computerChoice == 1 :

print("oops, " + name + " you lost this one :/")

#if player chooses scissors and computer chooses paper, player wins

elif playerChoice == 3 and computerChoice == 2 :

print(name + " wins! congrats champ!")

time.sleep(1)

#here we are asking if the user would like to play again, y for yes, n for no. the statusGame variable is set equal to true at the very top of the code (see above). If the player would like to play again, this will remain true and the loop will start over. If they choose not to play again, the loop will discontinue and the code will stop, they will not play again. If there is an invalid input, this will also result in the end of the game.

playAgain = input(name + ", would you like to play again? y for yes, n for no: ")

if playAgain == 'y':

statusGame = True

print('\n')

print("NEW GAME")

elif playAgain == 'n':

statusGame = False

print("BYE BYE FOR NOW")

else :

print("Invalid input. Game over.")

statusGame = False

**PART 6: Your Completed Program**

**Task**

Provide the Replit link to your full program code.

**Requirements**

* The program must work correctly with all the comments included in the program.

**Inspiration**

* Check before submitting your touchstone that your final version of the program is running successfully.

<https://replit.com/join/qwaheneush-rachelcole45>