Java 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:

Date:

Final Replit Program Share Link:

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 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 problem I'm solving is a way to play a simple game of rock, paper, scissors.
The program will simulate a game of rock paper scissors.
Depending on what the user and the computer inputs, will determine the winner of the game.

The game is simple and fun. The user supplies the data simply by following the input guidelines. The user only has to enter one of the expected responses while they're playing the game, there is an option to play again after every match and the user has the option to play or end the program. The user will always get a random response from the "OtherPlayer" or the computer, which makes every game different.

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 Java) 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:

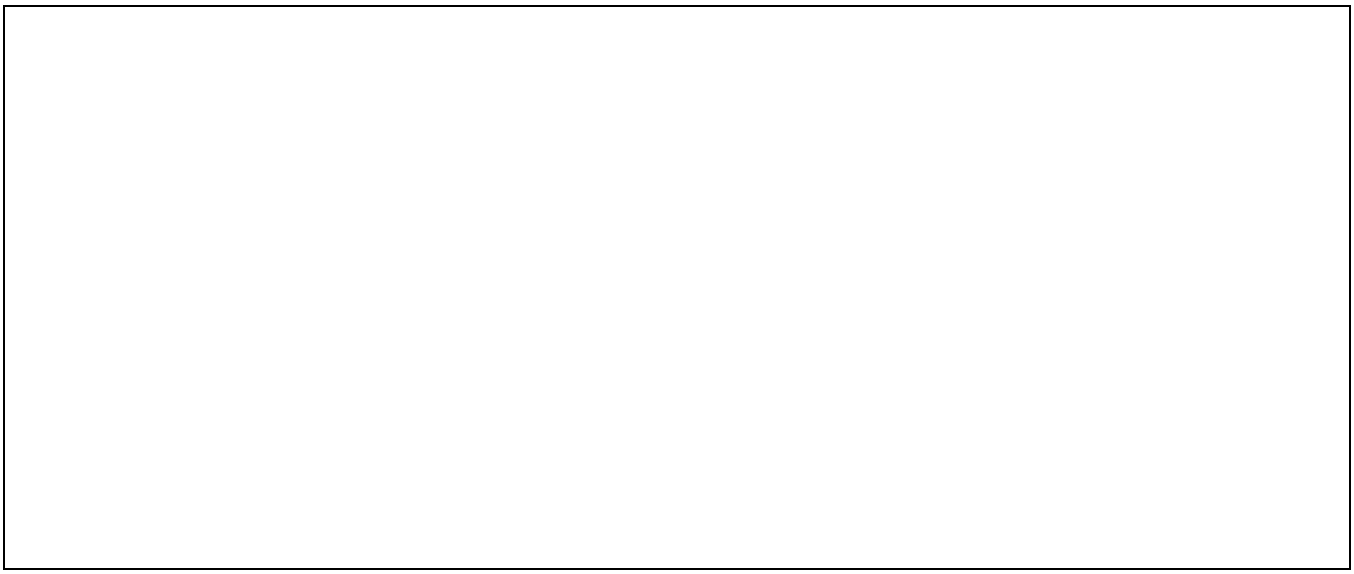
- Are there any steps that you don't fully understand? These are places to spend more time working out the details. Consider adding additional smaller steps in these spots.
- Remember that a computer program is very literal. Are there any steps that are unclear? Try giving the steps of your example/scenario 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: Player does not type in "rock"," paper", or "scissors"

1. The user is prompted to type in the correct words.

Scenario 2: The player inputs "rock", "paper" or "scissors"

1. The computer will randomly play "rock, paper or scissors", the player will win or lose



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 Java 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 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 Java? Aim for the clearest description of the steps, as this will make it easier to convert into program code later.

Rock Paper Scissors ()

The user is asked to select their choice of rock, paper or scissors.
If the user inputs the wrong text
else
the question is asked again, and the loop reiterates.
If the player types in the correct prompt (rock, paper or scissors)
Break;
The computer prints out their choice and lets the player know if they've won or lost and the player is asked if they want to play again.
If the user types in "yes"
The loop re-iterates, and the game starts over again
Else
The program stops.

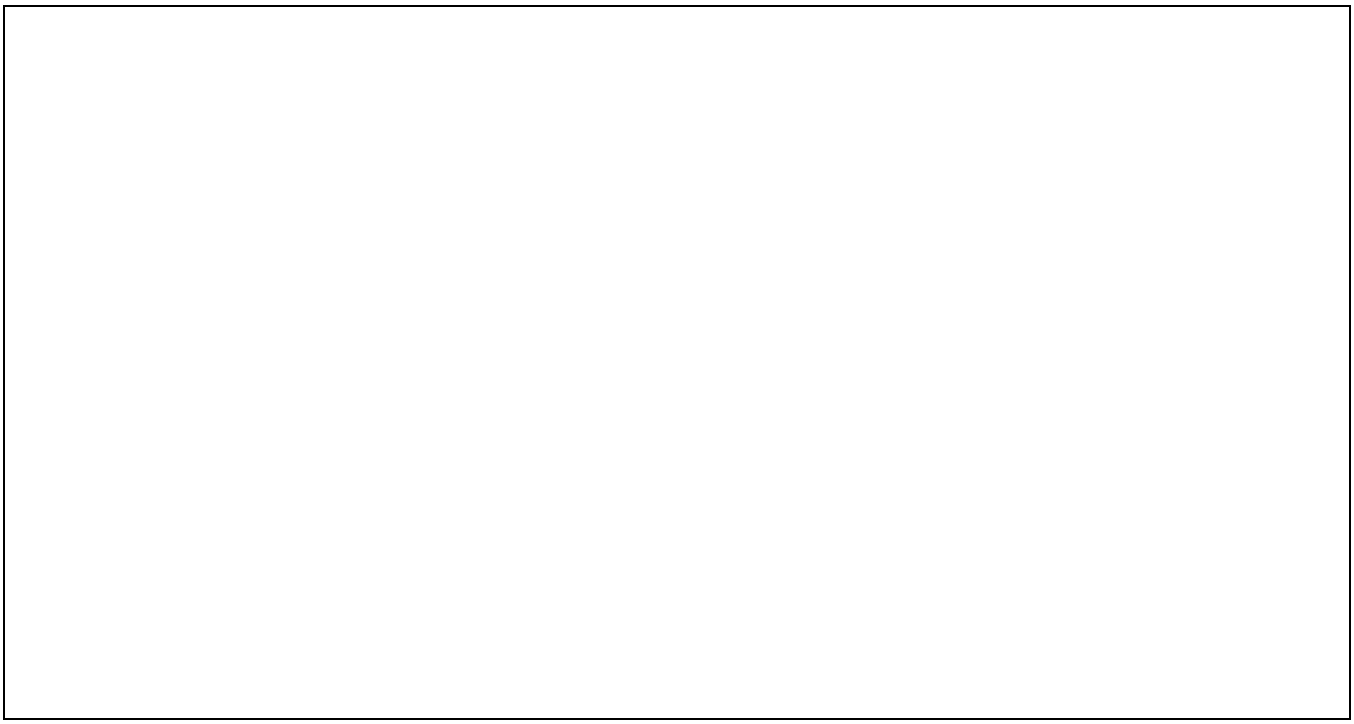
PART 4: Testing Your Program

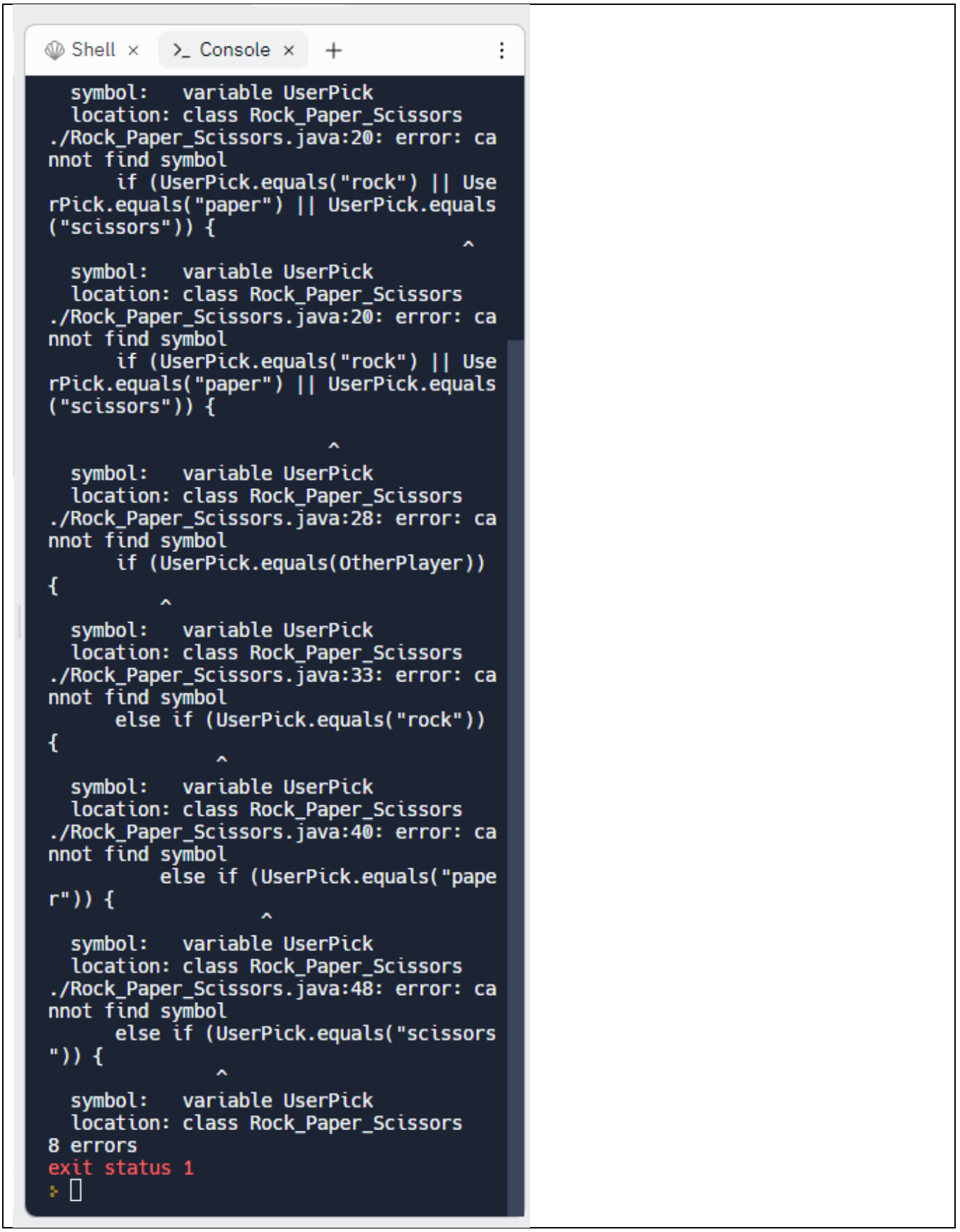
Task
While writing and testing your program code, describe your tests, record any errors, and state your approach to fixing the errors.

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 which 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?
  S




Shell
Console
symbol: variable UserPick
location: class Rock_Paper_Scissors
./Rock_Paper_Scissors.java:20: error: cannot find symbol
if (UserPick.equals("rock") || UserPick.equals("paper") || UserPick.equals("scissors")) {
^
symbol: variable UserPick
location: class Rock_Paper_Scissors
./Rock_Paper_Scissors.java:20: error: cannot find symbol
if (UserPick.equals("rock") || UserPick.equals("paper") || UserPick.equals("scissors")) {
^
symbol: variable UserPick
location: class Rock_Paper_Scissors
./Rock_Paper_Scissors.java:28: error: cannot find symbol
if (UserPick.equals(OtherPlayer)) {
^
symbol: variable UserPick
location: class Rock_Paper_Scissors
./Rock_Paper_Scissors.java:33: error: cannot find symbol
else if (UserPick.equals("rock")) {
^
symbol: variable UserPick
location: class Rock_Paper_Scissors
./Rock_Paper_Scissors.java:40: error: cannot find symbol
else if (UserPick.equals("paper")) {
^
symbol: variable UserPick
location: class Rock_Paper_Scissors
./Rock_Paper_Scissors.java:48: error: cannot find symbol
else if (UserPick.equals("scissors")) {
^
symbol: variable UserPick
location: class Rock_Paper_Scissors
8 errors
exit status 1

The String UserPick wasn't capitalized on Line 14.

Another issue that was presented was capitalizing all the prompts and then typing in the requested prompts. Since the cases didn't match the program kept outputting one response and wouldn't iterate through the loop. That problem took me the longest to correct unfortunately because the console didn't show that there were any errors, I had to meticulously look over the code for about an hour before I realized this issue. I tried to implement a to.UpperCase() to fix the issue but replit was starting to give me issues because I was maxing out the ram on the free version, so I chose to be more specific with my requirements for the users input.

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 Java 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.

```
import java.util.Random;
import java.util.Scanner;


public class Rock_Paper_Scissors {
  public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
    while(true) {
    String[] RockPaperScissors = {"rock", "paper","scissors"};
    String OtherPlayer = RockPaperScissors[new Random().nextInt(RockPaperScissors.length)];
  /* created 2 strings RockPapSciss takes 1 of the 3 inputs and randomizes the result. OtherPlayer
outputs the randomized input */


  String UserPick;
   /* The statement below "While (true)" verifies whether the user input one of the appropriate answers
to play the game, once that requirement is met, the loop is breaks and iterates through the next loop
*/

  while(true) {
      System.out.println("Rock Paper Scissors?");
      UserPick = scanner.nextLine();
      if (UserPick.equals("rock") || UserPick.equals("paper") || UserPick.equals("scissors")) {
       break;
      }
      System.out.println(" Please type Rock, Paper or Scissors");
    }
      /*prints out player selection */
     System.out.println("Other player threw out: " + OtherPlayer);

      if (UserPick.equals(OtherPlayer)) {
        System.out.println("Same move, Tie");
      }
        /* result of userPick and OtherPlayer outputting the same string.*/

      else if (UserPick.equals("rock")) {
      if (OtherPlayer.equals("paper")) {
        System.out.println("Paper beats Rock");
      } else if (OtherPlayer.equals("scissors")) {
        System.out.println("Rock beats Scissors, you win!");
        }
      } /* If userPick is Rock, it iterates through the possible ouputs of OtherPlayer and prints the
results of the match */
          else if (UserPick.equals("paper")) {
      if (OtherPlayer.equals("rock")) {
        System.out.println("Paper beats Rock, you win!");

      } else if (OtherPlayer.equals("scissors")) {
        System.out.println("Scissors beats Paper");
      }
```

```
    } /* If userPick is Paper, it iterates through the possible ouputs of OtherPlayer and prints the results
of the match */
      else if (UserPick.equals("scissors")) {
      if (OtherPlayer.equals("rock")) {
        System.out.println("Rock beats Scissors");
      } else if (OtherPlayer.equals("paper")) {
        System.out.println("Scissors beats Paper, you win!");
  }
      }
      System.out.println("Would you like to play again? yes or no?");
      String replay = scanner.nextLine();

      if (!replay.equals("yes")) {
        break;
      }
    }
  }
}
```

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/@sripton/RockPaperScissors#Rock_Paper_Scissors.java