**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: Whitney Perry**

**Date: 10/29/2024**

**Final IDE Program Join Link:** [**https://replit.com/join/sjpdbpkior-whitneyperrypro**](https://replit.com/join/sjpdbpkior-whitneyperrypro)

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. * 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:   * Is your problem clearly defined? * 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? |
| Rock, paper, scissors is a game of chance. It often is played by children to decide an outcome of something trivial such as “who gets the front seat?”, “who goes first?”, “who gets the first pick?” When deciding to play, the first point is to decide how many times to play and whoever gets the most amount of wins, wins the overall game. I will create a similar game except the player is playing against the software instead of another player. During gameplay, the player will be asked if they choose rock, paper, or scissor. Rock beats scissor, scissor beats paper, paper beats rock. The software will randomly generate one of these three options and then see who wins. This will be repeated for however many turns are chosen. If an even number was chosen and it’s a tie, another round will play to break the tie. At the end, the final winner will be declared. |

## 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 beats software in a round:   1. Player chooses paper and software chooses rock. 2. The values are compared 3. The player wins the round.   Scenario 2: Player loses to software in a round:   1. Player chooses paper and software chooses scissors. 2. The values are compared 3. The player loses the round.   Scenario 3: Player beats software in the game:   1. Player starts game choosing to play 3 rounds. 2. Player chooses paper and software chooses rock. 3. The values are compared 4. The player wins round. 5. Player chooses paper and software chooses scissors. 6. The values are compared 7. The player loses the round. 8. Player starts game choosing to play 3 rounds. 9. Player chooses paper and software chooses rock. 10. The values are compared 11. The player wins round. 12. All 3 rounds are complete with the player having 2 wins and software having 1 win. 13. The player wins the game. |

## 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. |
| Load game rules  Initialize rock as r, paper as p, and scissors as s.  If player value = r and software value = r  Add 1 to round and play again  Else if player value = r and software value = p  Add 1 to lose  Else if player value = r and software value = s  Add 1 to win  Else if player value = p and software value = r  Add 1 to win  Else if player value = p and software value = p  Add 1 to round and play again  Else if player value = p and software value = s  Add 1 to lose  Else if player value = s and software value = r  Add 1 to lose  Else if player value = s and software value = p  Add 1 to win  Else if player value = s and software value = s  Add 1 to round and play again  Else  Inform player that the available options are rock, paper, or scissor and to try again.  Start game  Set Win = 0  Set Lose = 0  Set Round = 0  Set Player input = blank  Set Software input = blank  Ask player how many rounds to play with an integer greater than or equal to 1 and no higher than 10.  If  number outside range, ask again.  Else  Set round = player input  Play Game  Minus 1 to round  Set player input = blank ad software input = blank  Request for r (rock), p (paper), or s (scissor) from play  If  Player input does not equal r, p, or s, ask again  Else  Generate a random value of r, p, or s and set as software input  Compare values with the game rules  If  Round = 0 and win and lose are not equal, display results and end the game  Else if  Round = 0 and win and lose are equal, add 1 to round and play again  Else  play again |

## 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 your IDE. 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? |
| 1. I didn’t get a screenshot of this but I originally struggled with understanding how to get the scope of variables to work. I went back to the text to research passing variables as a parameter and I was eventually able to figure it out. 2. I had issues with ensuring valid inputs for integers. After some online research, I was able to be resolved with a do while loop:   do {  System.out.println("\n\nPick Rock, Paper, or Scissors?\n");  while (!input.hasNextLine()) {  System.out.println("Enter a valid choice: rock, paper, or scissors.\n");  input.next(); // Clear the invalid input  }   1. I had logic errors where player was deemed the looser of the game when it was a tie and the rounds were 0. I originally had my tiebreaker code as part of the update score section that checked all three tie conditions and then also checked if the rounds were 0 and then add one but I don’t think the logic was right because it ultimately got ignored. I decided it would be cleaner to add it to checkWinner() and to be more explicit for the definition of what losing the game means.   if (round != 0) {  playGame(round,"",win,lose); //Keep playing if the rounds aren't finished  }  else if (round == 0 && win == lose) {  System.out.println("Tie breaker!"); //Play another round if it's a tie  round += 1;  playGame(round,"",win,lose);  }  else if (round == 0 && win > lose) {  System.out.println("\n" + win + " vs. " + lose);  System.out.println("You won the game!!!"); //Declare the player as the winner  }  else if (round == 0 && lose > win)  {  System.out.println("\n" + win + " vs. " + lose);  System.out.println("You lost the game!!! :("); //Declare player as the loser  }  else {  System.out.println("Error!");  } |

## 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.Scanner;  import java.util.Random;  public class Main {  public static void main(String[] args) {  //Declare variables  Scanner input = new Scanner(System.in);  String cmd = "";  Integer win = 0;  Integer lose = 0;  Integer round = 0;  String P1 = "";  //Menu Screen Selections  while(!cmd.toLowerCase().equals("exit")) {  displayMenu();  cmd = input.nextLine(); // Get user input  if(cmd.toLowerCase().equals("rules")) {  displayRules(); }  else if(cmd.toLowerCase().equals("play")) {  gameRounds(0);  }  }  }  //Menu Display  public static void displayMenu() {  System.out.println("\n\nRock, Paper, Scissors");  System.out.println("---------------------");  System.out.println("\nRules - Display Game Rules");  System.out.println("Play - Start Game");  System.out.println("Exit - End Game");  System.out.print("Enter a Command: ");  }  //Rules  public static void displayRules() {  System.out.println("\nRules:");  System.out.println("You are playing Rock, Paper, Scissors against the computer.");  System.out.println("The game is played by choosing one of the three options:");  System.out.println("\tRock, Paper, or Scissors.");  System.out.println("The game will also choose one of the three options.");  System.out.println("The winner is determined based on the following:");  System.out.println("\tRock beats Scissors");  System.out.println("\tScissors beats Paper");  System.out.println("\tPaper beats Rock\n");  System.out.println("If both players choose the same option, the game is tied and you play again.");  System.out.println("Choose to play from 1 round up to 10 rounds.");  System.out.println("Good Luck!\n\n");  }  //Decide how many rounds to play  public static void gameRounds(int round) {  Scanner input = new Scanner(System.in);  do {  System.out.println("\nHow many rounds would you like to play?");  System.out.println("Pick a number between 1 & 10");  while (!input.hasNextInt()) {  System.out.println("Enter a valid number.");  input.next(); // Clear the invalid input  }  round = input.nextInt();  if (round < 1 || round > 10) {  System.out.println("Enter the number of rounds to play between 1 & 10."); //Enforce a range of numbers between 1 and 10  }  } while (round < 1 || round > 10);  System.out.println("Let's play " + round + " rounds.\n");  playGame(round,"",0,0);  }  //Play Game  public static void playGame(int round, String P1, int win, int lose) {  Scanner input = new Scanner(System.in);  Random random = new Random();  do {  System.out.println("\n\nPick Rock, Paper, or Scissors?\n");  while (!input.hasNextLine()) {  System.out.println("Enter a valid choice: rock, paper, or scissors.\n");  input.next(); // Clear the invalid input  }  P1 = input.nextLine().toLowerCase();  if (!(P1.equals("rock") || P1.equals("paper") || P1.equals("scissors"))) {  System.out.println("Not a valid choice."); //Enforce a valid choice  }  } while (!(P1.equals("rock") || P1.equals("paper") || P1.equals("scissors")));  System.out.println("You chose " + P1 + ".");  int rand = random.nextInt(3) + 1; // Generates a number between 1 and 3 to represent rock, paper, or scissors  //Let player know what player 2 chose  if (rand == 1) {  System.out.println("Player 2 chose rock.");  }  else if (rand == 2) {  System.out.println("Player 2 chose paper.");  }  else if (rand == 3) {  System.out.println("Player 2 chose scissors.");  }  //Update score  if (  (P1.equals("rock") && rand == 1) ||  (P1.equals("paper") && rand == 2) ||  (P1.equals("scissors") && rand == 3)  ) {  System.out.println("It's a tie!");} //Tie condition  else if (  (P1.equals("rock") && rand == 2) ||  (P1.equals("paper") && rand == 3) ||  (P1.equals("scissors") && rand == 1) //Lose round condition  ) {  lose += 1;  System.out.println("You lose this round.");}  else if (  (P1.equals("rock") && rand == 3) ||  (P1.equals("paper") && rand == 1) ||  (P1.equals("scissors") && rand == 2) //Win round condition  ) {  win += 1;  System.out.println("You win this round.");}  else {  System.out.println("Error!");  }  round -= 1;  System.out.println("\nScore: P1 has " + win + " wins. P2 has " + lose + " wins."); //Let player know score  System.out.println("Rounds left: " + round); //Let player know rounds left  checkWinner(win,lose,round);  }  //Check rounds to see if game is over and declare winner  public static void checkWinner(int win, int lose, int round) {  if (round != 0) {  playGame(round,"",win,lose); //Keep playing if the rounds aren't finished  }  else if (round == 0 && win == lose) {  System.out.println("Tie breaker!"); //Play another round if it's a tie  round += 1;  playGame(round,"",win,lose);  }  else if (round == 0 && win > lose) {  System.out.println("\n" + win + " vs. " + lose);  System.out.println("You won the game!!!"); //Declare the player as the winner  }  else if (round == 0 && lose > win)  {  System.out.println("\n" + win + " vs. " + lose);  System.out.println("You lost the game!!! :("); //Declare player as the loser  }  else {  System.out.println("Error!");  }  }  } |

## PART 6: Your Completed Program

|  |
| --- |
| **Task**  Provide the IDE 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/sjpdbpkior-whitneyperrypro> |