**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:**

**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? |
| <Write your journal entry response here>  The problem to solve is a blackjack game between 1 player and 1 dealer with 1 deck of cards, no bets.  The inputs will be the player choosing to “hit” (get another card), “stand” (don’t get another card), and if they draw an ace, get to choose whether it’s worth 1 or 11. The player will also be able to change that that choice if they would like to draw higher. It will also have to account whether player or dealer get a  The main challenge will likely be in getting the “dealer”, aka computer to do this as well, most likely choosing randomly between either hit, stand, and between 1 or 11. It will also have to account whether player or dealer get a natural 21.  The outputs will be the player and dealers deck of cards and their sum, and the final result, which is either a win, loss, or tie between both. If the player or dealer get a 21, it’s an automatic win, otherwise it’s an automatic tie if they both get it.  The program will distribute two cards at random to the player and dealer. The player can access both their cards, while they can only see one of the dealers cards. The program will continue to prompt the player to hit or stand until they choose to stand. In between, if the player draws an ace, they get to choose whether it is 1 or 11. In the background, the “dealer” will also be doing this, following the same behavior. The program will also keep track of which cards have been distributed, so no more than one of each card is distributed to player or dealer. Once both dealer and player have chosen to stay, it will evaluate which of the decks have greater value, then outputs the winner. If the value is equal, it’s a tie. If either drew a natural 21, then that’s an automatic win. |

## 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? |
| <Write your journal entry response here>  **NATURAL 21 CASES**  If player gets a natural 21, then player automatically wins.  If dealer gets a natural 21, then dealer automatically wins  If dealer and player both get natural 21, then it is a tie.  **BOTH SIDES STAND, AND ARE 21 OR LESS**  If player total is greater than dealer total, then player wins  If dealer total is greater than player total, then player dealer wins  If player total and dealer total are equal, then it’s a tie.  **HIT LOSSES**  If player total is greater than 21, then player loses  If dealer total is greater than 21, dealer loses.  **When one draws an ACE**  If 1 or 11 is chosen, then add that to deck total.  These are the categories of results. Note that Natural 21 cases and “both sides stand” cases can end in a win, loss, *or* tie. But the hits can only result in a win or a loss. |

## 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. |
| <Write your pseudocode here> |

## 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? |
| <Record your errors and fixes here> |

## 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. |
| <Copy your full program code here, including comments> |

## 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. |
| <Provide the link to your program here> |