**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: Kelsey Thompson**

**Date: July 25, 2024**

**Final Replit Program Share Link:**

https://replit.com/join/woctutlwbb-kelseythompson4

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? |
| The problem I am planning to solve involves the shortage of available time for parents to read with their children. Busy parents desire to spend more quality and educational time with their kids. By tracking a daily schedule, users will better manage and calculate their time, addressing the current issue of overlapping activities and incomplete tasks. Reading in the evening is particularly affected. This issue needs addressing to enhance the child's reading development and family bonding time. The program will need to interact with the user continuously.  To manage and further your child's reading, the program will utilize input data such as activity type, time, date and length as well as day start/bed time. This data will create a small database of information. This will include activities that the parent completes throughout the day.  The program's outputs will include a calculation of the time remaining, that can be devoted to reading with the child. By calculating this data, the program aims to offer a clear picture of how time management impacts the time remaining, that can be devoted to reading with the child.  Both the parent and the program will supply the necessary data. The parent will provide schedule information and details about reading sessions, while the program will calculate potential reading time based on this input. The end product will be the calculated time remaining, to read with one's child. This is the output and completion of the program. |

## 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? |
| To solve the problem of tracking reading time with your child, I need to identify each activity and its duration, calculate the total time spent on these activities, and then determine the remaining time available for reading. This process will ensure that we can allocate sufficient time for reading without overlapping other activities.  Here are the following steps for two distinct examples:  Step 1: Input Data Required The necessary input data for this problem includes the Day Start Time activity type, time, date, length of each activity and Day Bed time.  Example 1:   * Day Start Time: 7:00 A.M * Activities:   + Jogging: 10:00 A.M to 11:00 A.M (1 hour)   + Cleaning: 12:00 P.M to 2:00 P.M (2 hours)   + Cooking: 3:00 P.M to 4:00 P.M (1 hour) * Day Bed time: 9:00 P.M   Example 2:   * Day Start Time: 8:00 A.M * Day Bed time: 10:00 P.M (14 hours total) * Activities:   + Grocery Shopping: 10:00 A.M to 11:00 A.M (1 hour)   + Cooking: 4:00 P.M to 6:00 P.M (2 hours)   + Reading to Child: 7:00 P.M to 8:00 P.M (1 hour)   Step 2: Calculate Time Remaining for Reading For each example, we will log and calculate the total time spent on activities and then determine the leftover time available for reading.  Example 1 Calculation:   * Total Day Time: 14 hours (from 7:00 A.M to 9:00 P.M) * Activity Time:   + Jogging: 1 hour   + Cleaning: 2 hours   + Cooking: 1 hour * Total Activity Time: 4 hours * Time Left for Reading: 14 hours - 4 hours = 10 hours   Example 2 Calculation:   * Total Day Time: 14 hours (from 8:00 A.M to 10:00 P.M) * Activity Time:   + Grocery Shopping: 1 hour   + Cooking: 2 hours   + Reading to Child: 1 hour * Total Activity Time: 4 hours * Time Left for Reading: 14 hours - 4 hours = 10 hours   Step 3: Time Left for Reading is calculated. Also, consider Time Validation and Handling Edge Cases. I need to ensure that the entered times are valid. If a user enters an invalid time format (e.g., 0800 instead of 8:00 A.M), the program should throw an 'Invalid Time Exception' and prompt the user to correct the input.  Example with Time Validation:   * User Input: 0800 instead of 8:00 A.M * Exception Handling: The program will detect the incorrect format, throw an 'Invalid Time Exception,' and prompt the user to enter the correct time format.   Now, by following these steps, I can effectively manage and track the time spent on various activities and ensure there is adequate time allocated for reading with your child. This structured approach will help enhance family bonding and support your child's educational progress by making sure reading time is not overlooked. |

## 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. |
| To develop the general sequence for my program, I considered a variety of user inputs and designed pseudocode to manage these inputs effectively (including my previous input examples). This pseudocode aims to handle various activities throughout the day, calculate the total activity time, and determine the remaining hours available for reading with the child. It also includes handling invalid time formats, as previously contemplated, through exception handling.  Here's the general sequence of the program using pseudocode:  // Prompt the user to enter the start and bed times for the day  day\_start\_time = input("Please enter day start time (MM/DD/YYYY): ")  day\_end\_time = input("Please enter day bed time (MM/DD/YYYY): ")  // Initialize an empty list to store activities and a variable for the total activity time  list = []  total\_activity\_time = 0  // Calculate the total hours available in the day  total\_hours\_of\_day = day\_end\_time - day\_start\_time  // Loop to collect activities from the user until they indicate they are done  while activities are different than x:  try:  // Prompt the user to enter an activity  activity = input("Please enter an activity or press x to finish: ")    // Add the activity to the list and update the total activity time  list.append(activity)  total\_activity\_time = total\_activity\_time + activity.getTime()  catch(InvalidTimeException e):  // Handle invalid time format by prompting the user to enter the correct format  print("Invalid Time format (MM/DD/YYYY)")  // Calculate the remaining hours available for reading with the child  remaining\_hours\_for\_the\_day = total\_hours\_of\_day - total\_activity\_time  // Print the remaining hours available for reading with the child  print(f"You have {remaining\_hours\_for\_the\_day} hours to read with your child")  --------------------------------------------------------------------------------------------------------------------------------------In this pseudocode, the program begins by prompting the user to enter the start and bed times of their day. It initializes an empty list to store the activities and a variable to keep track of the total activity time. The total hours available in the day are calculated by subtracting the start time from the bed time.  The program then enters a loop where it continuously prompts the user to enter activities until they choose to finish by pressing 'x'. Each activity is added to the list, and its duration is added to the total activity time.  Finally, the program calculates the remaining hours available for reading with the child by subtracting the total activity time from the total hours of the day and prints this information to the user.  This approach ensures a clear and organized structure, making it easier to convert the pseudocode into actual program code later in this project. |

## 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? |
| Throughout my revisions I focused on input validation and handling exceptions for time input. I resolved errors by modifying the code, specifically switching from generic to concrete types and simplifying the constructor from two parameters to one, allowing it to handle a single string input. Additionally, I fixed several compile errors and tested input prompts to ensure a smooth user experience.  Here is a compilation of corrected errors with the corresponding screen shots:   1. Errors:     I corrected the errors in deleting various lines. This achieved an alteration between generic type, to concrete type. Here is the updated code snippet:     1. Continued errors, requiring correction:     To correct this, I modified a constructor in my Java project (another file of the program) to accept a single string parameter instead of two individual inputs. Here, the errors are relieved:     1. Here is one example, of many compile errors, that I corrected throughout the project.     I simply added the semicolon to correct the compile error:     1. Here, I’m testing the input prompts for the user:      * + 1. I correct ‘invalid time”      * + 1. Here are additional corrections after I ran the test. Proceeding, is the first attempt:      * + 1. Below is my second attempt, with added alterations:      * + 1. I considered the result and tested further:      1. Here is another example of corrected code:      * + 1. Here is the corrected version with the corresponding, tests run. I added the necessary semicolons. I also added the data type ‘string’ to the activity name, which in turn, corrected the last error.      1. Here are the latest tests that I ran to ensure proficient functionality of the program, as a whole. 2. getActivityName method:      1. Here is a full run of the program, correctly calculating a remaining time for potential reading time. Therefore, my program fulfills the goal and purpose of it’s functionality:     Again, I view here that the calculations are correct (which are hughlighted) and the program is running smoothly. |

## 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. |
| Here is the link to my program  <https://replit.com/join/woctutlwbb-kelseythompson4>  Please follow the link, as the program code is too extensive for a Word document. I have enjoyed applying my previous knowledge of Java over the past month while writing this program.  Also, note all of the files in the program. All of which contain comments that were reviewed by peers: |

## 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. |
| My program is running fully and may be accessed on replit, here:  https://replit.com/join/woctutlwbb-kelseythompson4 |