Python Journal NAJD

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: NAJD FARIS A ALEID

Date: 01/06/2023

Final Replit Program Share Link:

https://replit.com/@NAJDALEID/SteelblueFirsthandInternet

https://replit.com/@NAJDALEID/HomelyShadyArtificialintelligence

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?

Problem Statement:

I am planning to develop a Python program that assists users in managing their daily water intake and promotes healthy hydration habits.
Requirements:
1. Problem Description: The problem at hand is the difficulty many individuals face in maintaining adequate daily water intake. Dehydration can have adverse effects on health and well-being, and people often struggle to keep track of their water consumption. This program aims to address this issue by providing a tool to monitor and manage daily water intake.
2. Input Data: The program will require input data related to the user's water consumption. This data may include the amount of water consumed at different times of the day, such as morning, afternoon, and evening. The user will enter this data interactively.
3. Program Functionality: The program will allow users to enter their water intake throughout the day and track their progress towards achieving the recommended daily water intake. It will provide reminders or notifications to encourage regular hydration. The program may also offer personalized recommendations based on factors like age, weight, and activity level to ensure optimal hydration.

4. Outputs and Results: The program will provide several outputs and results, including:
    - Daily water intake summary, showing the total amount of water consumed.
    - Progress towards the recommended daily water intake, displayed as a percentage.
    - Visual representations, such as charts or graphs, illustrating the user's hydration patterns over time.
    - Reminders or notifications to encourage regular water consumption.
    - Personalized recommendations for staying hydrated based on individual factors.

Inspiration:

- I want to solve this problem because dehydration can lead to various health issues, and maintaining proper hydration is essential for overall well-being.
- The input data required will be entered by the user interactively. The program may prompt the user to enter the amount of water consumed at different times of the day.
- The program will need to interact with the user throughout the day, reminding them to drink water and allowing them to input their water intake.
- The expected result is improved awareness and management of daily water intake. The user should be able to track their progress, receive reminders, and receive personalized recommendations to maintain optimal hydration.

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?

Example 1: Tracking Daily Water Intake
Input Data:

- Amount of water consumed at different times of the day (morning, afternoon, evening).

Steps to Solve the Problem:

1. Start the program.
2. Display a welcome message and instructions on how to use the program.
3. Initialize variables for total water intake and recommended daily water intake.
4. Prompt the user to enter the amount of water consumed in the morning.
5. Add the entered amount to the total water intake variable.
6. Prompt the user to enter the amount of water consumed in the afternoon.
7. Add the entered amount to the total water intake variable.
8. Prompt the user to enter the amount of water consumed in the evening.
9. Add the entered amount to the total water intake variable.
10. Calculate the percentage of the recommended daily water intake achieved by dividing the total water intake by the recommended daily water intake and multiplying by 100.
11. Display the total water intake and the percentage achieved.

12. If the percentage achieved is less than 100%, display a reminder or notification to drink more water.
13. End the program.

Result of Example 1: Suppose the user enters 250 ml of water in the morning, 350 ml in the afternoon, and 200 ml in the evening. The program will calculate the total water intake (800 ml) and the percentage achieved based on the recommended daily water intake. It will display the total water intake as well as the percentage achieved, which can be used by the user to track their progress towards proper hydration.

Example 2: Hydration Reminder and Recommendation

Input Data:

- Recommended daily water intake.
- User's age, weight, and activity level.

Steps to Solve the Problem:

1. Start the program.
2. Display a welcome message and instructions on how to use the program.
3. Prompt the user to enter their age, weight, and activity level.
4. Calculate the recommended daily water intake based on the user's age, weight, and activity level.
5. Initialize variables for total water intake and current percentage achieved.
6. Set a reminder interval for hydration reminders (e.g., every 2 hours).
7. Display the recommended daily water intake to the user.
8. Enter a loop that repeats every reminder interval: a. Display a reminder to drink water. b. Prompt the user to enter the amount of water consumed since the last reminder. c. Add the entered amount to the total water intake variable. d. Calculate the current percentage achieved. e. Display the total water intake and the current percentage achieved. f. If the percentage achieved is less than 100%, provide personalized recommendations for maintaining hydration based on the user's age, weight, and activity level. g. If the user chooses to end the program, exit the loop.
9. End the program.

Result of Example 2: Suppose the user is recommended to consume 2 liters of water daily based on their age, weight, and activity level. The program will remind the user to drink water every 2 hours and prompt them to enter the amount of water consumed since the last reminder. It will calculate the total water intake and the current percentage achieved. If the percentage achieved is less than 100%, the program will provide personalized recommendations for maintaining hydration based on the user's individual factors. The user can continue receiving reminders and recommendations until they choose to end the program.

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.

```
# Example 1: Tracking Daily Water Intake

# Initialize variables
total_water_intake = 0
recommended_daily_intake = 2000

# Prompt the user to enter the amount of water consumed in the morning
morning_water = get_user_input("Enter the amount of water consumed in the morning")

# Add the morning water intake to the total
total_water_intake += morning_water

# Prompt the user to enter the amount of water consumed in the afternoon
afternoon_water = get_user_input("Enter the amount of water consumed in the afternoon")

# Add the afternoon water intake to the total
total_water_intake += afternoon_water

# Prompt the user to enter the amount of water consumed in the evening
evening_water = get_user_input("Enter the amount of water consumed in the evening")

# Add the evening water intake to the total
total_water_intake += evening_water

# Calculate the percentage of recommended daily intake achieved
percentage_achieved = (total_water_intake / recommended_daily_intake) * 100

# Display the total water intake and percentage achieved
display_output("Total water intake:", total_water_intake)
display_output("Percentage achieved:", percentage_achieved)

# Example 2: Hydration Reminder and Recommendation

# Prompt the user to enter their age, weight, and activity level
age = get_user_input("Enter your age")
weight = get_user_input("Enter your weight")
activity_level = get_user_input("Enter your activity level")

# Calculate the recommended daily water intake based on user input
recommended_daily_intake = calculate_daily_intake(age, weight, activity_level)

# Initialize variables
total_water_intake = 0
current_percentage_achieved = 0
```

```
reminder_interval = 2  # hours

# Display the recommended daily water intake
display_output("Recommended daily intake:", recommended_daily_intake)

# Enter a loop for hydration reminders
while True:
    # Display reminder to drink water
    display_output("Reminder: Drink water")

    # Prompt the user to enter the amount of water consumed since the last reminder
    consumed_water = get_user_input("Enter the amount of water consumed since the last
reminder")

    # Add the consumed water to the total
    total_water_intake += consumed_water

    # Calculate the current percentage achieved
    current_percentage_achieved = (total_water_intake / recommended_daily_intake) * 100

    # Display the total water intake and current percentage achieved
    display_output("Total water intake:", total_water_intake)
    display_output("Current percentage achieved:", current_percentage_achieved)

    # Check if percentage achieved is less than 100%
    if current_percentage_achieved < 100:
        # Provide personalized recommendations for hydration
        recommendations = get_hydration_recommendations(age, weight, activity_level)
        display_output("Recommendations:", recommendations)

    # Check if user wants to end the program
    if user_wants_to_end():
        break

# End of program
```

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?

PART 4: Testing Your Program
During the testing of the program, I will focus on different scenarios and inputs to ensure the correctness and robustness of the code. Here are some test cases I will consider:
Test Case 1: Tracking Daily Water Intake

- Input:
    - Morning water: 500
    - Afternoon water: 700
    - Evening water: 800
- Expected Output:
    - Total water intake: 2000
    - Percentage achieved: 100

Description: This test case aims to verify if the code correctly calculates the total water intake and the percentage achieved based on the provided inputs. By comparing the expected output with the actual output, we can determine if the calculations are accurate.
Test Case 2: Hydration Reminder and Recommendation

- Input:
    - Age: 30
    - Weight: 70
    - Activity level: Moderate
- Expected Output:
    - Recommended daily intake: Calculated based on age, weight, and activity level

Description: This test case focuses on checking if the code can accurately calculate the recommended daily water intake based on user input. By comparing the calculated recommended intake with the expected value, we can ensure that the calculation is correct.
Error Handling Test: Incorrect User Input

- Input:
    - Morning water: "500"
- Expected Output:
    - Error message: "Invalid input. Please enter a numeric value."

Description: This test case checks how the program handles incorrect user input. In this case, the user enters a string instead of a numeric value for the morning water intake. The program should detect this error and display an appropriate error message to the user. The expected output should be the error message mentioned above.
If any errors occur during the testing, I will carefully examine the error messages provided by the system and the code logic to identify the cause. I will review the corresponding sections of the code that are related to the error and attempt to fix it. If necessary, I will debug the code by adding print statements or using a debugger to trace the flow of execution and identify any logical errors or incorrect variable values.
I will also make sure to handle potential exceptions or edge cases, such as division by zero, invalid user inputs, or unexpected behavior due to certain conditions. By thoroughly testing these scenarios and refining the code, I can ensure that it is robust and capable of handling various situations.

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

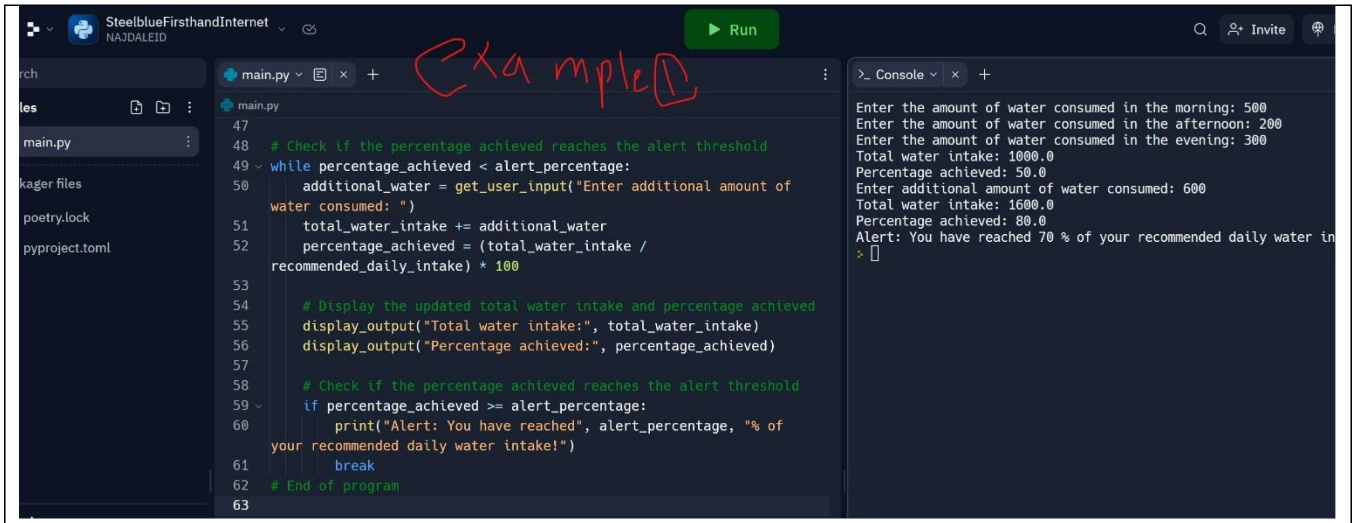
- The purpose of the program and each of its parts should be clear to a reader that does not know the Python programming language.

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.

```
# Example 1: Tracking Daily Water Intake
# Function to get user input and validate the input format
def get_user_input(prompt):
    while True:
        try:
            value = float(input(prompt))
            if value >= 0:
                return value
            else:
                print("Please enter a non-negative value.")
        except ValueError:
            print("Please enter a valid number.")

# Function to display output
def display_output(label, value):
    print(label, value)

# Initialize variables
total_water_intake = 0
recommended_daily_intake = 2000
alert_percentage = 70  # Set the percentage at which you want to trigger the alert
```

```
# Prompt the user to enter the amount of water consumed in the morning
morning_water = get_user_input("Enter the amount of water consumed in the morning: ")

# Add the morning water intake to the total
total_water_intake += morning_water

# Prompt the user to enter the amount of water consumed in the afternoon
afternoon_water = get_user_input("Enter the amount of water consumed in the afternoon: ")

# Add the afternoon water intake to the total
total_water_intake += afternoon_water

# Prompt the user to enter the amount of water consumed in the evening
evening_water = get_user_input("Enter the amount of water consumed in the evening: ")

# Add the evening water intake to the total
total_water_intake += evening_water

# Calculate the percentage of recommended daily intake achieved
percentage_achieved = (total_water_intake / recommended_daily_intake) * 100

# Display the total water intake and percentage achieved
display_output("Total water intake:", total_water_intake)
display_output("Percentage achieved:", percentage_achieved)

# Check if the percentage achieved reaches the alert threshold
while percentage_achieved < alert_percentage:
    additional_water = get_user_input("Enter additional amount of water consumed: ")
    total_water_intake += additional_water
    percentage_achieved = (total_water_intake / recommended_daily_intake) * 100

    # Display the updated total water intake and percentage achieved
    display_output("Total water intake:", total_water_intake)
    display_output("Percentage achieved:", percentage_achieved)

    # Check if the percentage achieved reaches the alert threshold
    if percentage_achieved >= alert_percentage:
        print("Alert: You have reached", alert_percentage, "% of your recommended daily water
intake!")
        break
# End of program
```

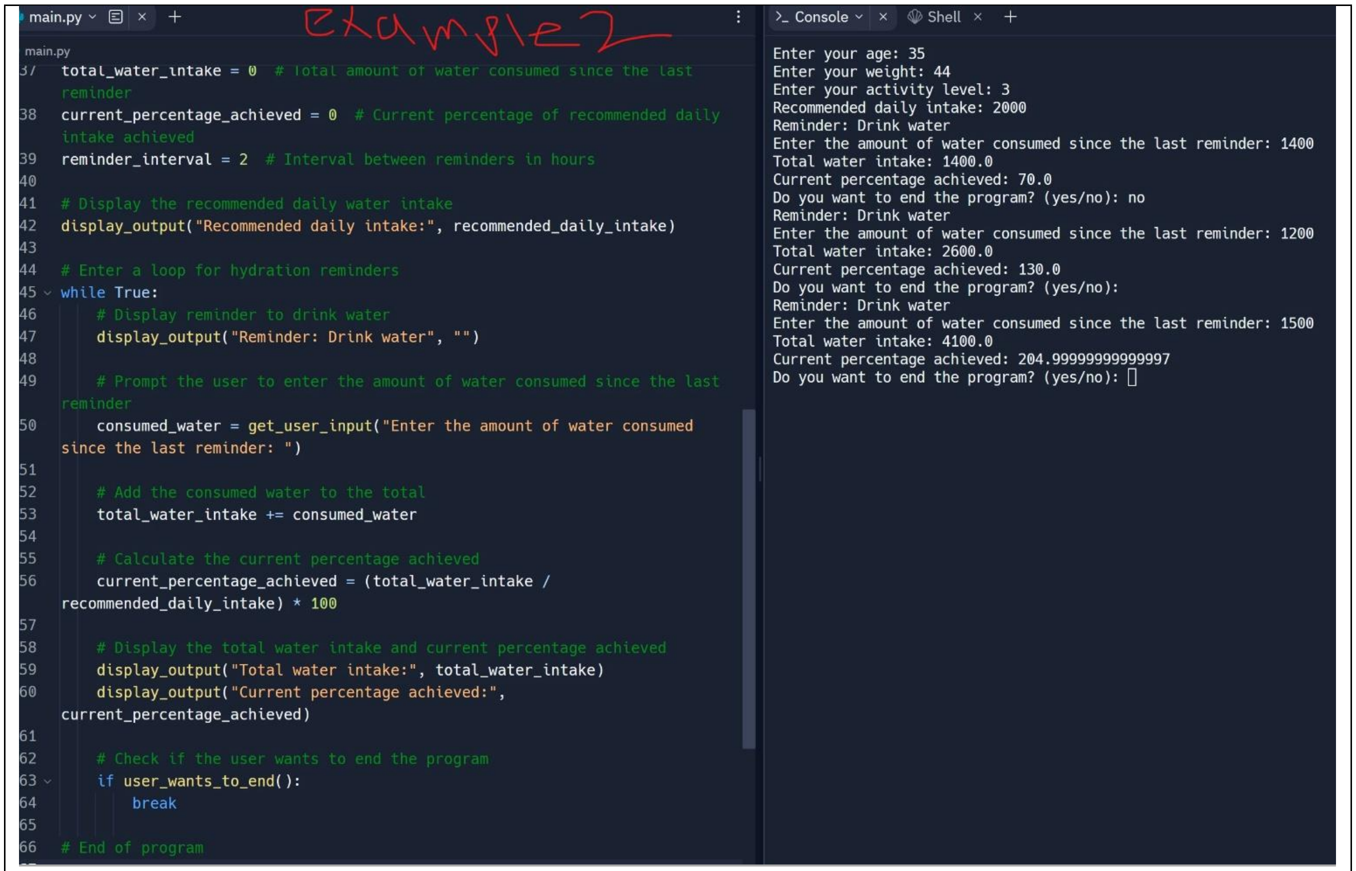


```python

# Check if the percentage achieved reaches the alert threshold
while percentage_achieved < alert_percentage:
    additional_water = get_user_input("Enter additional amount of
water consumed: ")
    total_water_intake += additional_water
    percentage_achieved = (total_water_intake /
recommended_daily_intake) * 100

    # Display the updated total water intake and percentage achieved
    display_output("Total water intake:", total_water_intake)
    display_output("Percentage achieved:", percentage_achieved)

    # Check if the percentage achieved reaches the alert threshold
    if percentage_achieved >= alert_percentage:
        print("Alert: You have reached", alert_percentage, "% of
your recommended daily water intake!")
        break
# End of program

```

```
Enter the amount of water consumed in the morning: 500
Enter the amount of water consumed in the afternoon: 200
Enter the amount of water consumed in the evening: 300
Total water intake: 1000.0
Percentage achieved: 50.0
Enter additional amount of water consumed: 600
Total water intake: 1600.0
Percentage achieved: 80.0
Alert: You have reached 70 % of your recommended daily water in
> 
```

```python
# Example 2: Hydration Reminder and Recommendation
# Function to calculate the recommended daily water intake
def calculate_daily_intake(age, weight, activity_level):
    # Add your logic to calculate the recommended daily water intake based on the provided
parameters
    # Placeholder implementation
    return 2000

# Function to get user input and validate the input format
def get_user_input(prompt):
    while True:
        try:
            value = float(input(prompt))
            if value >= 0:
                return value
            else:
                print("Please enter a non-negative value.")
        except ValueError:
            print("Please enter a valid number.")

# Function to display output
def display_output(label, value):
    print(label, value)

# Function to check if the user wants to end the program
def user_wants_to_end():
    response = input("Do you want to end the program? (yes/no): ")
    return response.lower() == "yes"

# Prompt the user to enter their age, weight, and activity level
```

```
age = get_user_input("Enter your age: ")
weight = get_user_input("Enter your weight: ")
activity_level = get_user_input("Enter your activity level: ")

# Calculate the recommended daily water intake based on user input
recommended_daily_intake = calculate_daily_intake(age, weight, activity_level)

# Initialize variables
total_water_intake = 0  # Total amount of water consumed since the last reminder
current_percentage_achieved = 0  # Current percentage of recommended daily intake
achieved
reminder_interval = 2  # Interval between reminders in hours

# Display the recommended daily water intake
display_output("Recommended daily intake:", recommended_daily_intake)

# Enter a loop for hydration reminders
while True:
    # Display reminder to drink water
    display_output("Reminder: Drink water", "")

    # Prompt the user to enter the amount of water consumed since the last reminder
    consumed_water = get_user_input("Enter the amount of water consumed since the last
reminder: ")

    # Add the consumed water to the total
    total_water_intake += consumed_water

    # Calculate the current percentage achieved
    current_percentage_achieved = (total_water_intake / recommended_daily_intake) * 100

    # Display the total water intake and current percentage achieved
    display_output("Total water intake:", total_water_intake)
    display_output("Current percentage achieved:", current_percentage_achieved)

    # Check if the user wants to end the program
    if user_wants_to_end():
        break

# End of program
```



```python
total_water_intake = 0  # Total amount of water consumed since the last reminder
current_percentage_achieved = 0  # Current percentage of recommended daily intake achieved
reminder_interval = 2  # Interval between reminders in hours

# Display the recommended daily water intake
display_output("Recommended daily intake:", recommended_daily_intake)

# Enter a loop for hydration reminders
while True:
    # Display reminder to drink water
    display_output("Reminder: Drink water", "")

    # Prompt the user to enter the amount of water consumed since the last reminder
    consumed_water = get_user_input("Enter the amount of water consumed since the last reminder: ")

    # Add the consumed water to the total
    total_water_intake += consumed_water

    # Calculate the current percentage achieved
    current_percentage_achieved = (total_water_intake / recommended_daily_intake) * 100

    # Display the total water intake and current percentage achieved
    display_output("Total water intake:", total_water_intake)
    display_output("Current percentage achieved:", current_percentage_achieved)

    # Check if the user wants to end the program
    if user_wants_to_end():
        break

# End of program
```

```
Enter your age: 35
Enter your weight: 44
Enter your activity level: 3
Recommended daily intake: 2000
Reminder: Drink water
Enter the amount of water consumed since the last reminder: 1400
Total water intake: 1400.0
Current percentage achieved: 70.0
Do you want to end the program? (yes/no): no
Reminder: Drink water
Enter the amount of water consumed since the last reminder: 1200
Total water intake: 2600.0
Current percentage achieved: 130.0
Do you want to end the program? (yes/no):
Reminder: Drink water
Enter the amount of water consumed since the last reminder: 1500
Total water intake: 4100.0
Current percentage achieved: 204.99999999999997
Do you want to end the program? (yes/no):
```

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.
1 https://replit.com/@NAJDALEID/SteelblueFirsthandInternet   2 https://replit.com/@NAJDALEID/HomelyShadyArtificialintelligence