**Part 1: Meal Cost Calculation**

1. Input:

Ask the user to enter the charge for the food.

1. Process:
2. Create a list to store the meal cost components.
3. Calculate the tip, which is 18% of the food charge) and store it in the list.
4. Calculate the sales tax, which is 7% of the food charge and store it in the list.
5. Calculate the total amount (sum of food charge, tip, and tax) and store it in the list.
6. Output:

Access and display the food charge, tip amount, tax amount, and total amount from the list.

**Source code:**

# Part 1: Meal Cost Calculation

# Input: Request the food charge from the user

food\_charge = float(input("Enter the charge for the food: $"))

# Calculate tip and tax

tip = food\_charge \* 0.18

tax = food\_charge \* 0.07

total = food\_charge + tip + tax

# Store results in a list

costs = [food\_charge, tip, tax, total]

# Display the results using the list

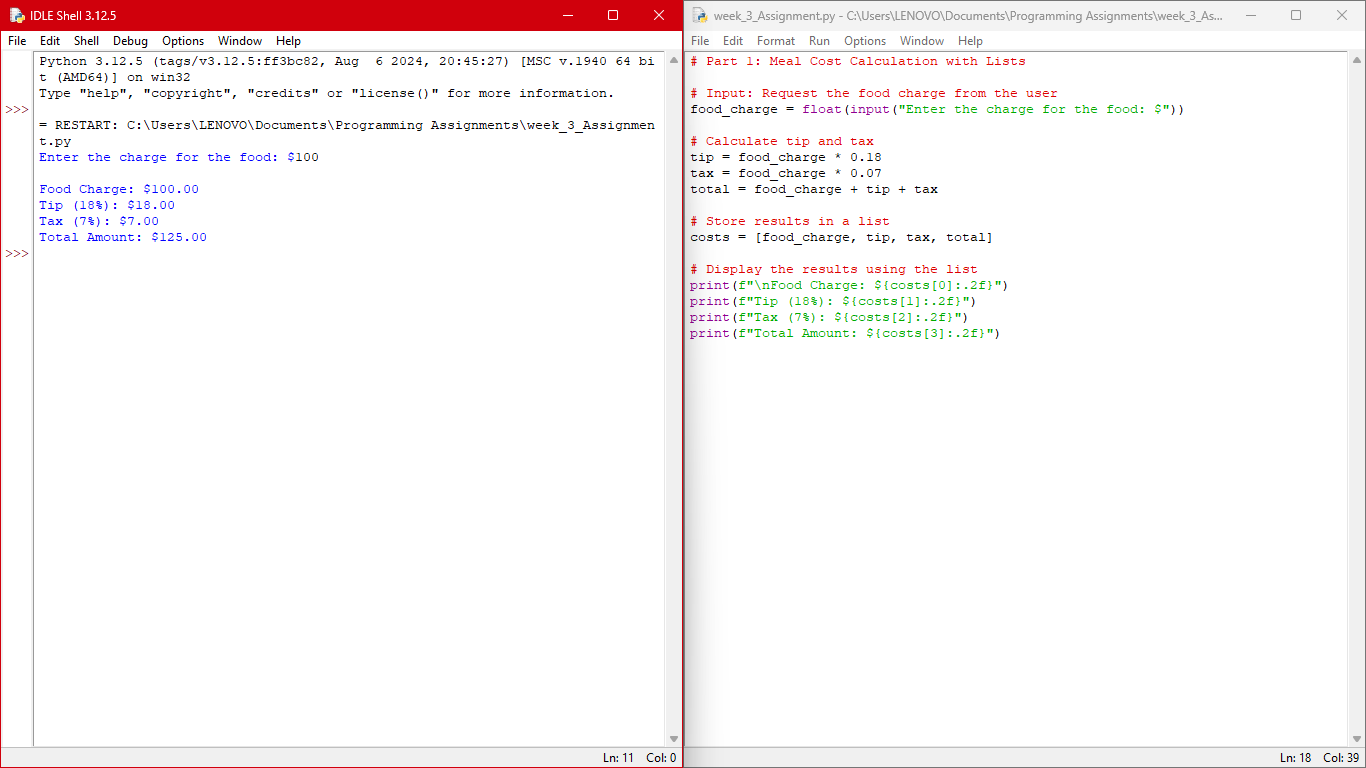
print(f"\nFood Charge: ${costs[0]:.2f}")

print(f"Tip (18%): ${costs[1]:.2f}")

print(f"Tax (7%): ${costs[2]:.2f}")

print(f"Total Amount: ${costs[3]:.2f}")

**Screenshot of the application executing the code from Part 1:**



**Part 2: 24-Hour Clock Alarm**

Pseudocode:

1. **Input**: Request the current time in hours from the user and the number of hours to wait for the alarm to activate.
2. **Process**:
   1. Create a list to store the current time, wait time, and the calculated alarm time.
   2. Calculate the time the alarm will activate using the modulus operation to ensure the time stays within 0-23 hours. The 24-hour clock has hours ranging from 0 to 23.
   3. When calculating the time at which the alarm will go off, if the sum of the current time and the wait time exceeds 23 (the highest hour in a 24-hour clock), the clock has to wrap around to the correct hour. **Example:** Suppose the current time is 13 (1 PM) and the alarm has to go off in 50 hours. Without the modulo operation, the time would be 13 + 50 = 63, but since there are only 24 hours in a day, 63 is not a valid hour. Here, 63 % 24 = 15. This calculation effectively wraps the time around to the correct hour, which is 15:00 (3 PM).
3. **Output**: Access and display the current time, wait time, and alarm time from the list.

**Source code:**

# Part 2: 24-Hour Clock Alarm

# Input: Request the current time in hours from the user and the number of hours to wait for the alarm to activate.

current\_time = int(input("What is the current time in hours (0-23): "))

time\_to\_wait = int(input("Input the number of hours to wait for the alarm: "))

# Calculate time the alarm will go off

alarm\_time = (current\_time + time\_to\_wait) % 24

# % 24 ensures that the resulting time is always within the range of 0 to 23

# Store the inputs and result in a list

time\_data = [current\_time, time\_to\_wait, alarm\_time]

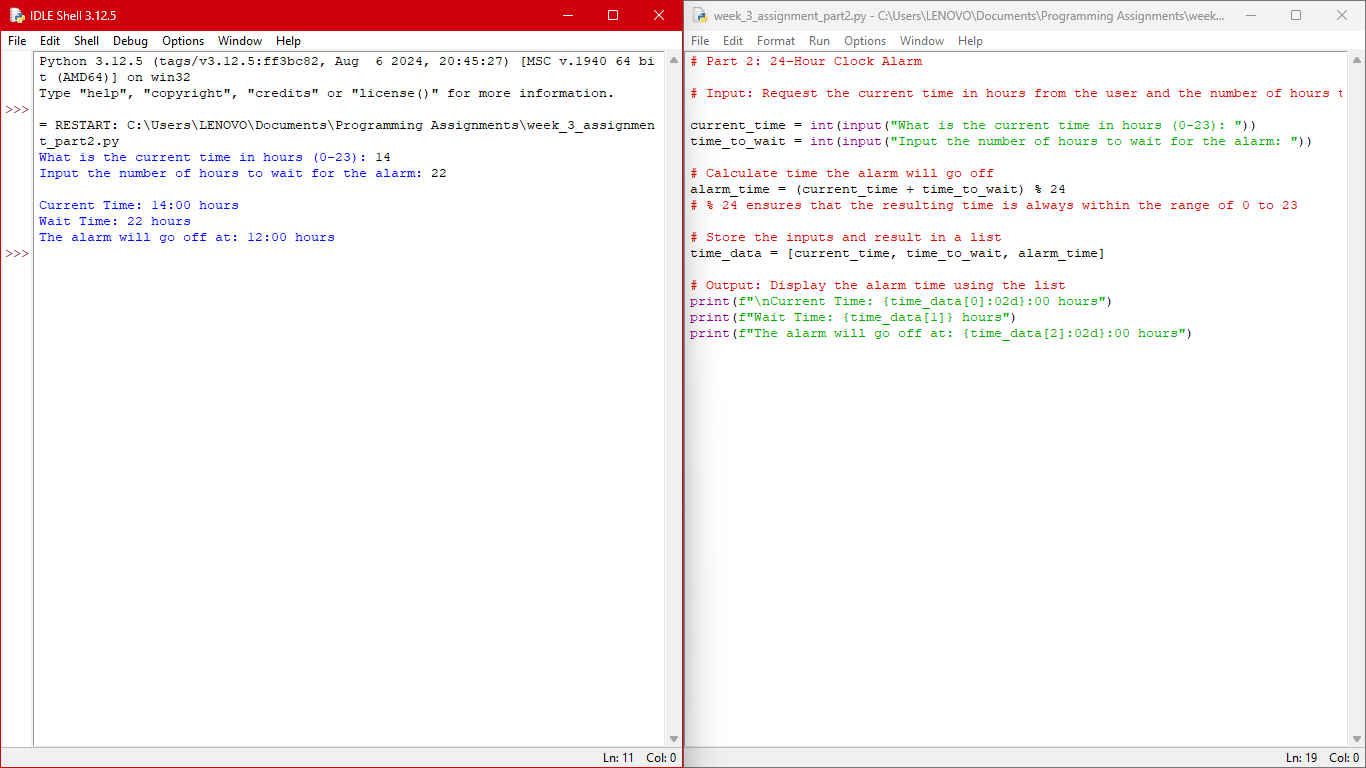
# Output: Display the alarm time using the list

print(f"\nCurrent Time: {time\_data[0]:02d}:00 hours")

print(f"Wait Time: {time\_data[1]} hours")

print(f"The alarm will go off at: {time\_data[2]:02d}:00 hours")

**Screenshot of the application executing the code from Part 2:**



**GIT repository link:**

<https://github.com/giftataylor/week_3_Assignment>