Program Assignments – While Loops. Develop an IPO for each of the problems below. Place the IPO into your repository. Then write code for each problem and place those files (py) into your repository. Paste the link to your repository into the Assignment Completion Link on Blackboard.

1. Display the odd numbers starting at 1 and ending with 25. Use a while loop structure for this problem.

Input:

- There are no inputs required from the user.

Process:

1. Initialize a variable num to 1.
2. While num is less than or equal to 25:
    - Print num.
    - Increment number by 2 to get the next odd number.

Output:

- The odd numbers from 1 to 25, inclusive.

2. Allow the user to enter a start value, stop value and increment value from the keyboard. Display all the numbers from the start value to stop value using the increment value as you proceed. Use a while loop structure for this problem.

Input:

- Start value (integer) entered by the user.
- Stop value (integer) entered by the user.
- Increment value (integer) entered by the user.

Process:

1. Prompt the user to enter the start value, stop value, and increment value.
2. Initialize a variable current_value to the start value.
3. While current_value is less than or equal to the stop value:
    - Print current_value.
    - Increment current_value by the increment value.

Output:

All numbers from the start value to the stop value, inclusive, incremented by the increment value.

3. Prompt the user on whether they want to do this program **(just before the while loop**). “Yes” entry means they want to continue. Any other response indicates they will stop the program. This response is the loop control variable. If the user answers “Yes “then go into the while loop.

   Once in the while loop. You are to prompt the user for their last name and two exam scores. Compute the average exam score. Display last name and average. After the loop, display a count of the number of students who entered data.

   Finally, the **last statements within the while loop** will ask the user if they want to do this loop again. In other words the user needs to be prompted again. The reason is that the end of the loop takes execution to the while condition to be evaluated again. It can not take us to the first few lines of code that prompt the user for the first time. That code is out of the loop. Therefore, we need a second prompt at the bottom, inside the loop.

Input

1. Initial user prompt to decide whether to start the program.
2. User's last name.
3. Two exam scores.
4. User prompt to decide whether to continue after each loop iteration.

Process

1. Check if the user wants to start the program.
2. If "Yes", enter a while loop:
   - Prompt for the last name and two exam scores.
   - Calculate the average of the two exam scores.
   - Display the last name and the average score.
   - Increment the count of students.
   - Ask if the user wants to continue with another iteration.
3. If the user does not want to continue, exit the loop.
4. After the loop, display the count of students who entered data.

Output

1. Last name and average exam score for each student.
2. Total number of students who entered data.

4. Prompt the user on whether they want to do this program **(just before the while loop**). Yes means they want to continue. Any other response indicates they will stop the program. This response is the loop control variable. If the user answers Yes then go into the while loop.

Once in the while loop. You are to prompt the user for employee last name, hours worked and rate of pay. Compute gross pay. Give the employee time and a half for hours worked over 40. Sum the gross pay and count the number of employees.

For each employee display their last name and gross pay.

After the loop (all data entered) display the sum of all the gross pays, and count of the number of employees. Compute and display the average pay.

Finally, the **last statements within the while loop** will ask the user if they want to do this loop again. In other words the user needs to be prompted again. The reason is that the end of the loop takes execution to the while condition to be evaluated again. It can not take us to the first few lines of code that prompt the user for the first time. That code is out of the loop. Therefore, we need a second prompt at the bottom, inside the loop.

Input:

- Prompt user whether they want to continue (continue_prompt)
- Inside loop:
    - Employee last name (last_name)
    - Hours worked (hours_worked)
    - Rate of pay (rate_of_pay)
    - Prompt to continue loop (loop_prompt)

Process:

- Initialize variables for total gross pay (total_gross_pay) and employee count (employee_count).
- Use a while loop to repeatedly prompt the user for employee information until they choose to stop.
- Calculate gross pay:
    - Regular pay for hours up to 40 hours.
    - Overtime pay (time and a half) for hours over 40.
- Sum up the gross pay and increment the employee count for each iteration.
- Compute the average pay after exiting the loop.

Output:

- For each employee entered, display their last name and gross pay.
- After exiting the loop, display:
    - Sum of all gross pays (total_gross_pay)
    - Number of employees (employee_count)
    - Average pay (average_pay)

5. Prompt the user on whether they want to do this program **(just before the while loop**). Yes means they want to continue. Any other response indicates they will stop the program. This response is the loop control variable. If the user answers Yes then go into the while loop.

Once in the while loop. You are to prompt the user for quantity and price of an item. Compute extended price (quantity times price of an item. If the extended price is greater than 10000.00 compute a discount of 25%. All other orders get a 10% discount. For each order display extended price, discount amount (extended price x discount percent), total (extended price – discount amount).

For each order sum the discount amount.

After the loop (all data entered) display the sum of all the discounts.

Finally, the **last statements within the while loop** will ask the user if they want to do this loop again. In other words the user needs to be prompted again. The reason is that the end of the loop takes execution to the while condition to be evaluated again. It can not take us to the first few lines of code that prompt the user for the first time. That code is out of the loop. Therefore, we need a second prompt at the bottom, inside the loop.

Input:

- Prompt user whether they want to continue (continue_prompt)
- Inside loop:
    - Quantity of an item (quantity)
    - Price of an item (price)
    - Prompt to continue loop (loop_prompt)

Process:

- Initialize variables for total discount (total_discount).
- Use a while loop to repeatedly prompt the user for item information until they choose to stop.
- Calculate extended price (extended_price) as quantity * price.
- Determine discount based on extended price:
    - 25% discount if extended price > 10000.00.
    - 10% discount otherwise.
- Calculate discount amount (discount_amount).
- Compute total (total) as extended_price - discount_amount.
- Sum up the discount amount (total_discount) for each iteration.

Output:

- For each item entered, display:
    - Extended price
    - Discount amount
    - Total (extended price - discount amount)
- After exiting the loop, display:
    - Sum of all discount amounts (total_discount)