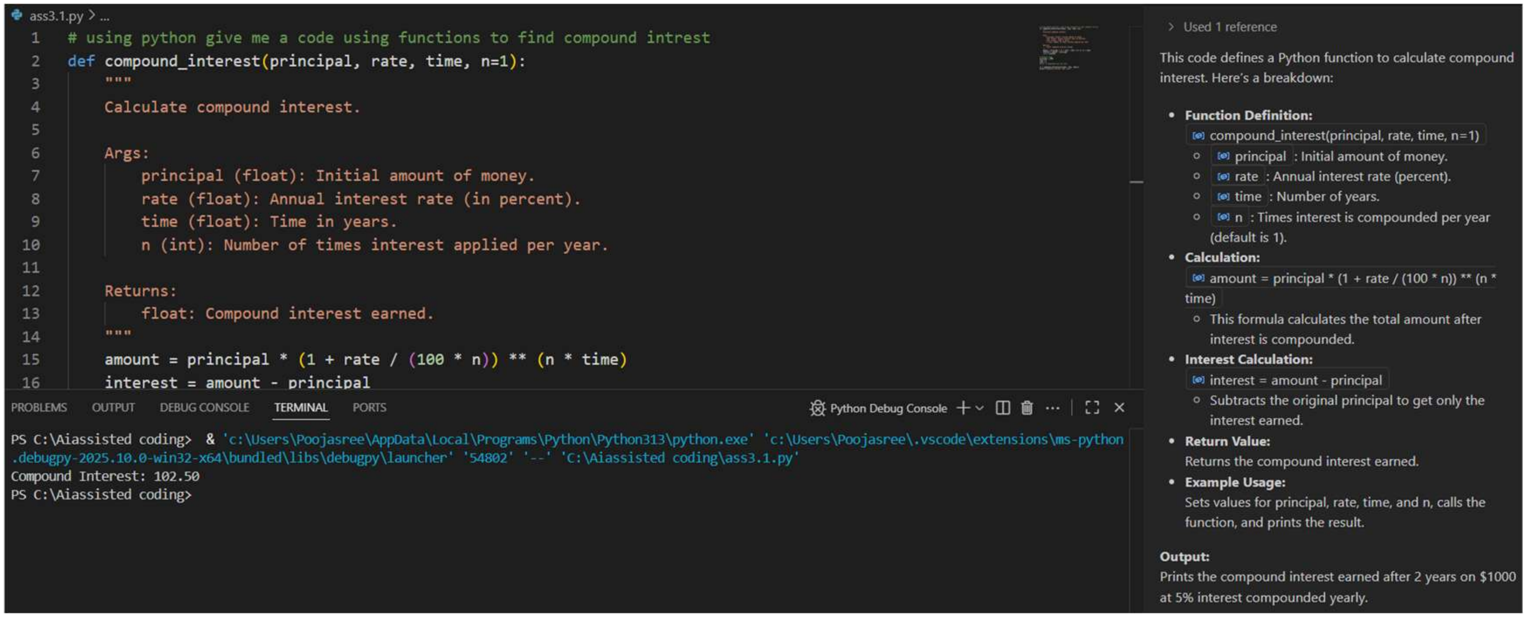
Assignment 3

Task-1:

Prompt: Using python give me a code using functions to find compound interest.



```python
# using python give me a code using functions to find compound intrest
def compound_interest(principal, rate, time, n=1):
    """
    Calculate compound interest.

    Args:
        principal (float): Initial amount of money.
        rate (float): Annual interest rate (in percent).
        time (float): Time in years.
        n (int): Number of times interest applied per year.

    Returns:
        float: Compound interest earned.
    """
    amount = principal * (1 + rate / (100 * n)) ** (n * time)
    interest = amount - principal
```

```
PS C:\Aiassisted coding>  & 'c:\Users\Poojasree\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\Poojasree\.vscode\extensions\ms-python
.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '54802' '--' 'C:\Aiassisted coding\ass3.1.py'
Compound Interest: 102.50
PS C:\Aiassisted coding>
```

> Used 1 reference

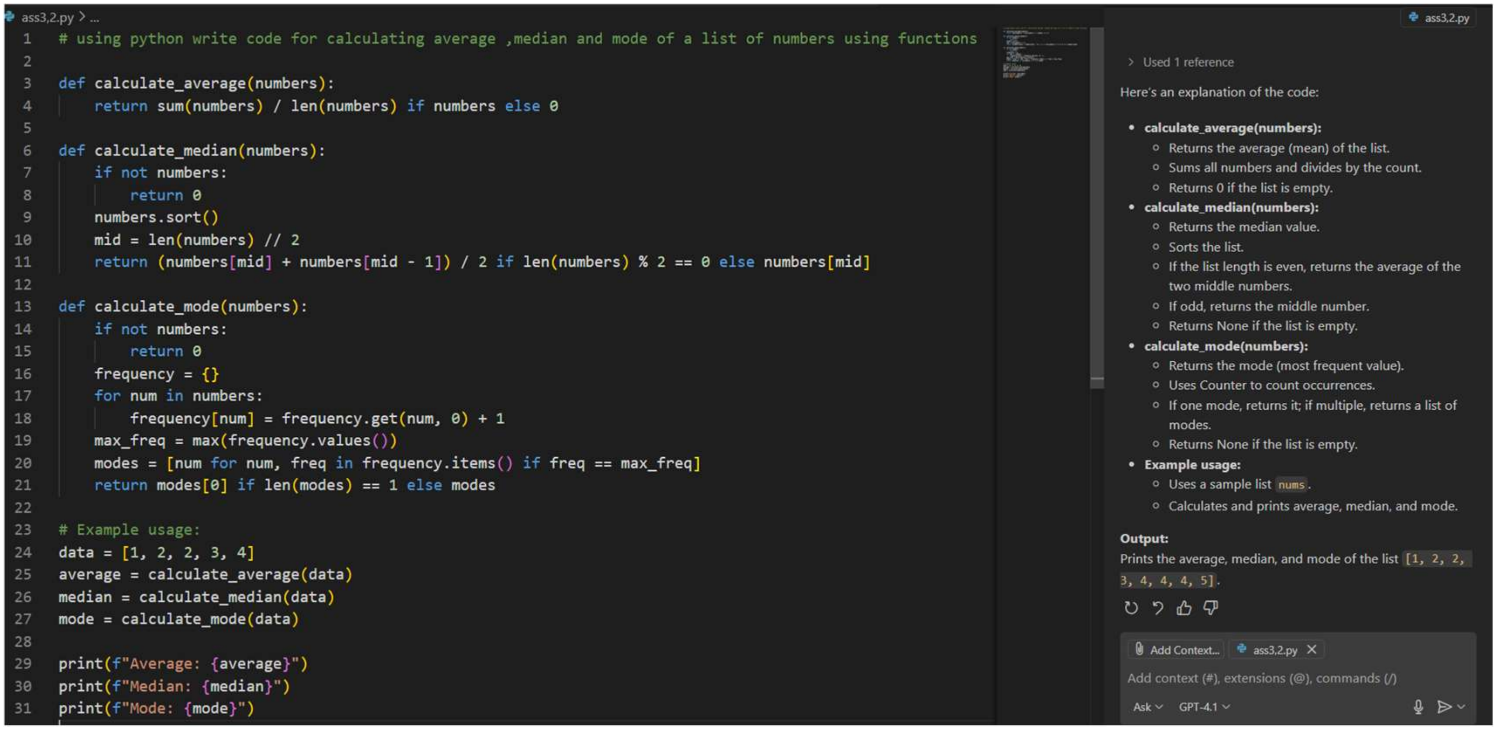
This code defines a Python function to calculate compound interest. Here's a breakdown:

- **Function Definition:**
  `compound_interest(principal, rate, time, n=1)`
  - `principal` : Initial amount of money.
  - `rate` : Annual interest rate (percent).
  - `time` : Number of years.
  - `n` : Times interest is compounded per year (default is 1).
- **Calculation:**
  `amount = principal * (1 + rate / (100 * n)) ** (n * time)`
  - This formula calculates the total amount after interest is compounded.
- **Interest Calculation:**
  `interest = amount - principal`
  - Subtracts the original principal to get only the interest earned.
- **Return Value:**
  Returns the compound interest earned.
- **Example Usage:**
  Sets values for principal, rate, time, and n, calls the function, and prints the result.

Output:
Prints the compound interest earned after 2 years on $1000 at 5% interest compounded yearly.

Task-2:

#using python write code for calculating average ,median and mode of a list of numbers using functions



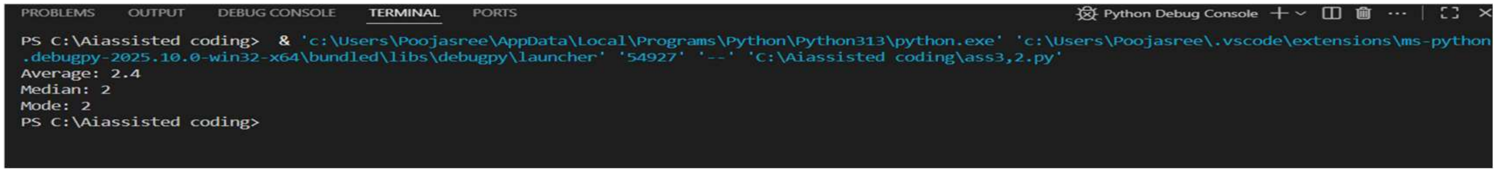
```python
# using python write code for calculating average ,median and mode of a list of numbers using functions

def calculate_average(numbers):
    return sum(numbers) / len(numbers) if numbers else 0

def calculate_median(numbers):
    if not numbers:
        return 0
    numbers.sort()
    mid = len(numbers) // 2
    return (numbers[mid] + numbers[mid - 1]) / 2 if len(numbers) % 2 == 0 else numbers[mid]

def calculate_mode(numbers):
    if not numbers:
        return 0
    frequency = {}
    for num in numbers:
        frequency[num] = frequency.get(num, 0) + 1
    max_freq = max(frequency.values())
    modes = [num for num, freq in frequency.items() if freq == max_freq]
    return modes[0] if len(modes) == 1 else modes

# Example usage:
data = [1, 2, 2, 3, 4]
average = calculate_average(data)
median = calculate_median(data)
mode = calculate_mode(data)

print(f"Average: {average}")
print(f"Median: {median}")
print(f"Mode: {mode}")
```

> Used 1 reference

Here's an explanation of the code:

- **calculate_average(numbers):**
  - Returns the average (mean) of the list.
  - Sums all numbers and divides by the count.
  - Returns 0 if the list is empty.
- **calculate_median(numbers):**
  - Returns the median value.
  - Sorts the list.
  - If the list length is even, returns the average of the two middle numbers.
  - If odd, returns the middle number.
  - Returns None if the list is empty.
- **calculate_mode(numbers):**
  - Returns the mode (most frequent value).
  - Uses Counter to count occurrences.
  - If one mode, returns it; if multiple, returns a list of modes.
  - Returns None if the list is empty.
- **Example usage:**
  - Uses a sample list `nums`.
  - Calculates and prints average, median, and mode.

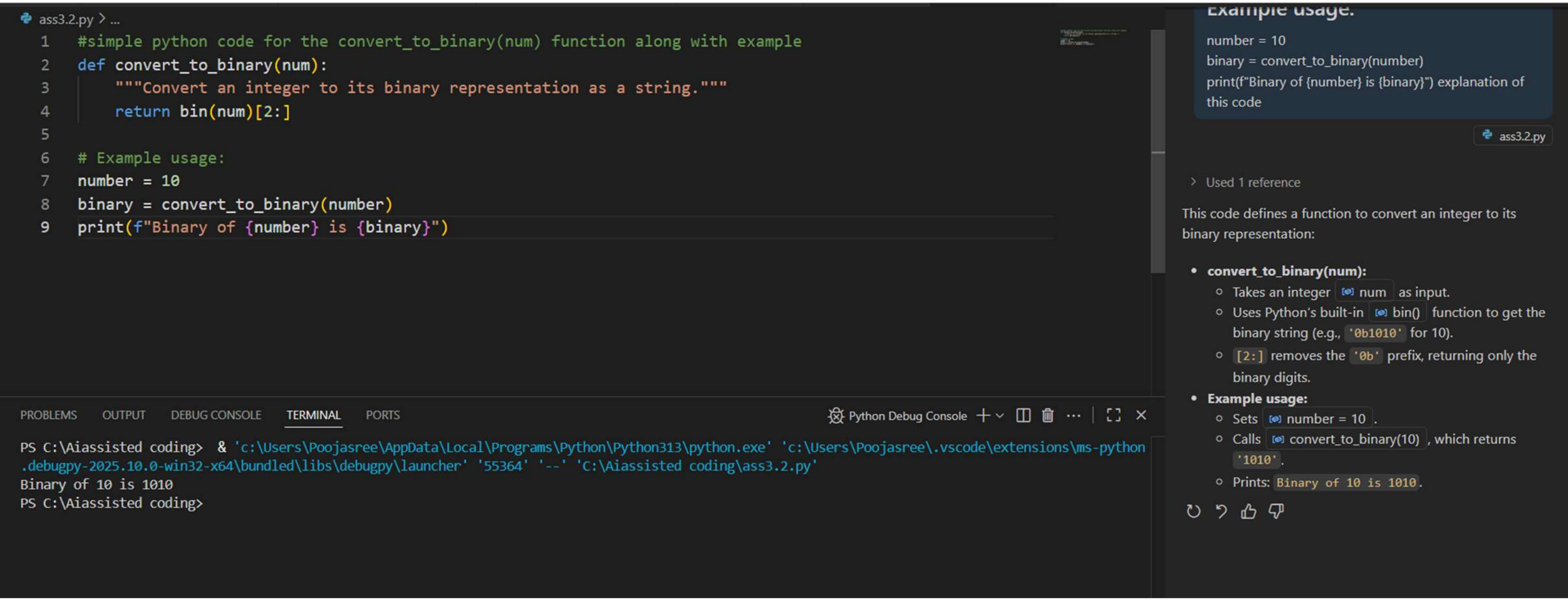
Output:
Prints the average, median, and mode of the list `[1, 2, 2, 3, 4, 4, 4, 5]`.



```
PS C:\Aiassisted coding>  & 'c:\Users\Poojasree\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\Poojasree\.vscode\extensions\ms-python
.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '54927' '--' 'C:\Aiassisted coding\ass3,2.py'
Average: 2.4
Median: 2
Mode: 2
PS C:\Aiassisted coding>
```

Task-3:

#simple python code for the convert_to_binary(num) function along with example



```python
#simple python code for the convert_to_binary(num) function along with example
def convert_to_binary(num):
    """Convert an integer to its binary representation as a string."""
    return bin(num)[2:]

# Example usage:
number = 10
binary = convert_to_binary(number)
print(f"Binary of {number} is {binary}")
```

```
PS C:\Aiassisted coding>  & 'c:\Users\Poojasree\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\Poojasree\.vscode\extensions\ms-python
.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '55364' '--' 'C:\Aiassisted coding\ass3.2.py'
Binary of 10 is 1010
PS C:\Aiassisted coding>
```

Example usage:

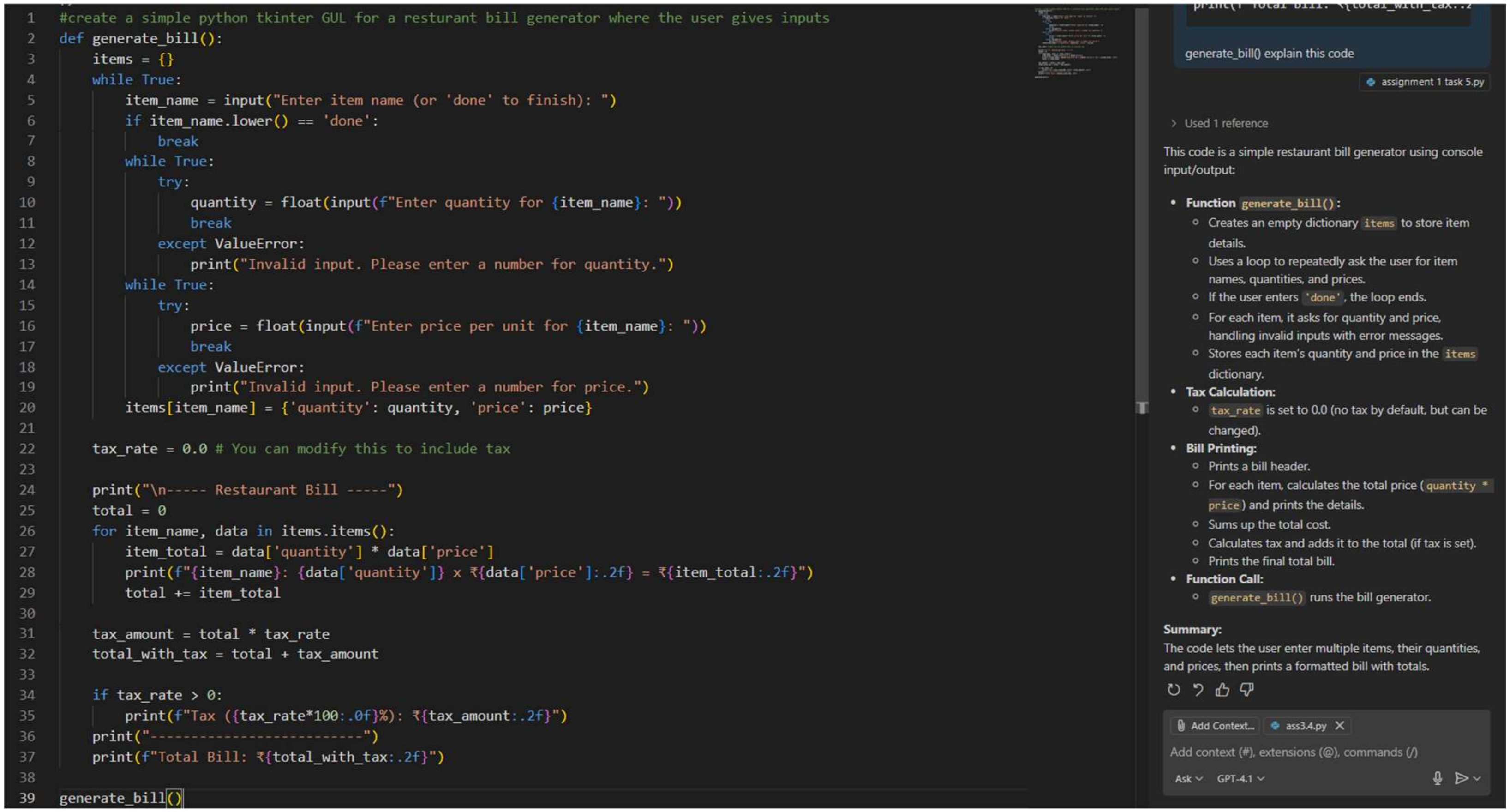
number = 10
binary = convert_to_binary(number)
print(f"Binary of {number} is {binary}") explanation of this code

This code defines a function to convert an integer to its binary representation:

- **convert_to_binary(num):**
  - Takes an integer `num` as input.
  - Uses Python's built-in `bin()` function to get the binary string (e.g., `'0b1010'` for 10).
  - `[2:]` removes the `'0b'` prefix, returning only the binary digits.
- **Example usage:**
  - Sets `number = 10` .
  - Calls `convert_to_binary(10)` , which returns `'1010'`.
  - Prints: `Binary of 10 is 1010`.

Task-4:

# using python generate restutant bill in text format



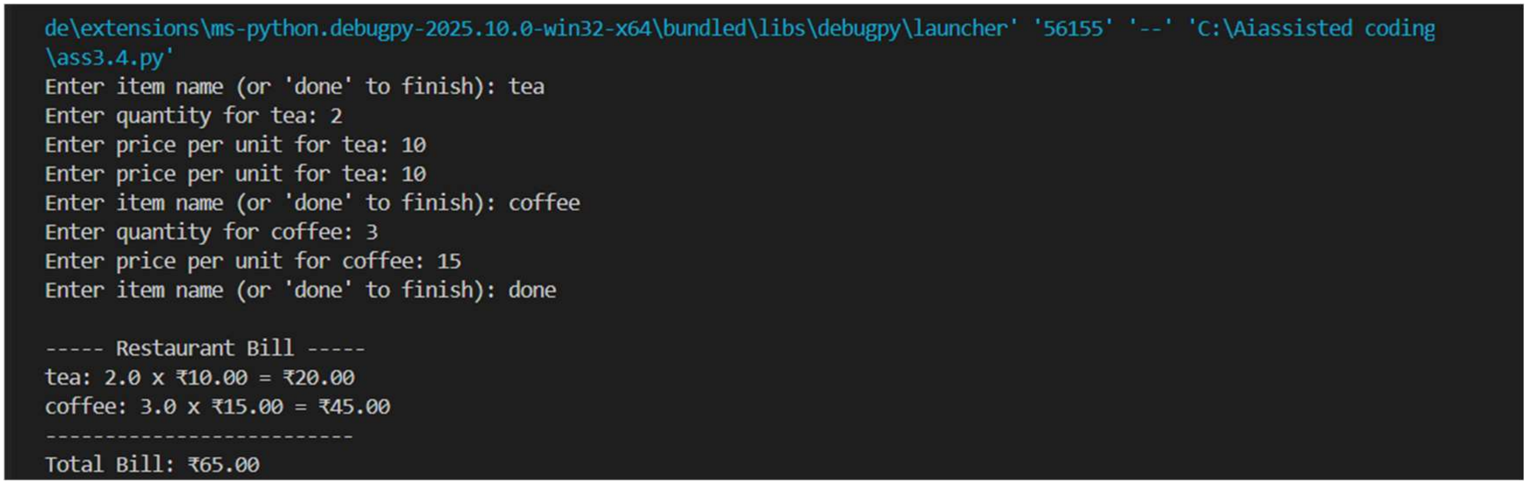
```python
#create a simple python tkinter GUL for a resturant bill generator where the user gives inputs
def generate_bill():
    items = {}
    while True:
        item_name = input("Enter item name (or 'done' to finish): ")
        if item_name.lower() == 'done':
            break
        while True:
            try:
                quantity = float(input(f"Enter quantity for {item_name}: "))
                break
            except ValueError:
                print("Invalid input. Please enter a number for quantity.")
        while True:
            try:
                price = float(input(f"Enter price per unit for {item_name}: "))
                break
            except ValueError:
                print("Invalid input. Please enter a number for price.")
        items[item_name] = {'quantity': quantity, 'price': price}

    tax_rate = 0.0 # You can modify this to include tax

    print("\n----- Restaurant Bill -----")
    total = 0
    for item_name, data in items.items():
        item_total = data['quantity'] * data['price']
        print(f"{item_name}: {data['quantity']} x ₹{data['price']:.2f} = ₹{item_total:.2f}")
        total += item_total

    tax_amount = total * tax_rate
    total_with_tax = total + tax_amount

    if tax_rate > 0:
        print(f"Tax ({tax_rate*100:.0f}%): ₹{tax_amount:.2f}")
    print("--------------------------")
    print(f"Total Bill: ₹{total_with_tax:.2f}")

generate_bill()
```

generate_bill() explain this code

This code is a simple restaurant bill generator using console input/output:

- **Function `generate_bill()`:**
    - Creates an empty dictionary `items` to store item details.
    - Uses a loop to repeatedly ask the user for item names, quantities, and prices.
    - If the user enters `'done'`, the loop ends.
    - For each item, it asks for quantity and price, handling invalid inputs with error messages.
    - Stores each item's quantity and price in the `items` dictionary.
- **Tax Calculation:**
    - `tax_rate` is set to 0.0 (no tax by default, but can be changed).
- **Bill Printing:**
    - Prints a bill header.
    - For each item, calculates the total price (`quantity * price`) and prints the details.
    - Sums up the total cost.
    - Calculates tax and adds it to the total (if tax is set).
    - Prints the final total bill.
- **Function Call:**
    - `generate_bill()` runs the bill generator.

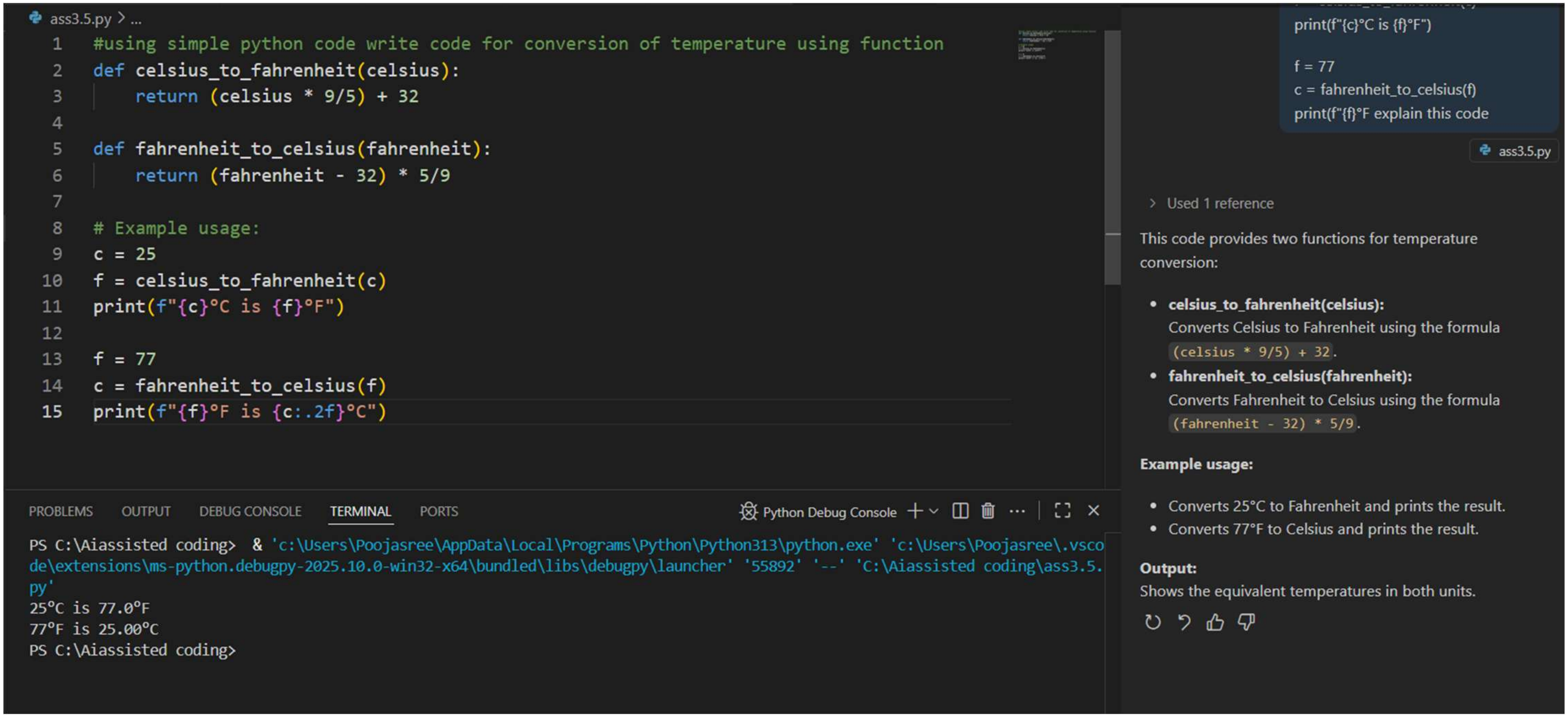
Summary:
The code lets the user enter multiple items, their quantities, and prices, then prints a formatted bill with totals.



```
de\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '56155' '--' 'C:\Aiassisted coding
\ass3.4.py'
Enter item name (or 'done' to finish): tea
Enter quantity for tea: 2
Enter price per unit for tea: 10
Enter price per unit for tea: 10
Enter item name (or 'done' to finish): coffee
Enter quantity for coffee: 3
Enter price per unit for coffee: 15
Enter item name (or 'done' to finish): done

----- Restaurant Bill -----
tea: 2.0 x ₹10.00 = ₹20.00
coffee: 3.0 x ₹15.00 = ₹45.00
--------------------------
Total Bill: ₹65.00
```

Task-5:

#using simple python code write code for conversion of temperature using function



```python
#using simple python code write code for conversion of temperature using function
def celsius_to_fahrenheit(celsius):
    return (celsius * 9/5) + 32

def fahrenheit_to_celsius(fahrenheit):
    return (fahrenheit - 32) * 5/9

# Example usage:
c = 25
f = celsius_to_fahrenheit(c)
print(f"{c}°C is {f}°F")

f = 77
c = fahrenheit_to_celsius(f)
print(f"{f}°F is {c:.2f}°C")
```

```
PS C:\Aiassisted coding>  & 'c:\Users\Poojasree\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\Poojasree\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '55892' '--' 'C:\Aiassisted coding\ass3.5.py'
25°C is 77.0°F
77°F is 25.00°C
PS C:\Aiassisted coding>
```

```
print(f"{c}°C is {f}°F")

f = 77
c = fahrenheit_to_celsius(f)
print(f"{f}°F explain this code
```

This code provides two functions for temperature conversion:

- **celsius_to_fahrenheit(celsius):** Converts Celsius to Fahrenheit using the formula `(celsius * 9/5) + 32`.
- **fahrenheit_to_celsius(fahrenheit):** Converts Fahrenheit to Celsius using the formula `(fahrenheit - 32) * 5/9`.

Example usage:

- Converts 25°C to Fahrenheit and prints the result.
- Converts 77°F to Celsius and prints the result.

Output:
Shows the equivalent temperatures in both units.