

SR UNIVERSITY

AI ASSIST CODING

Lab-3.2

E.RAVALI

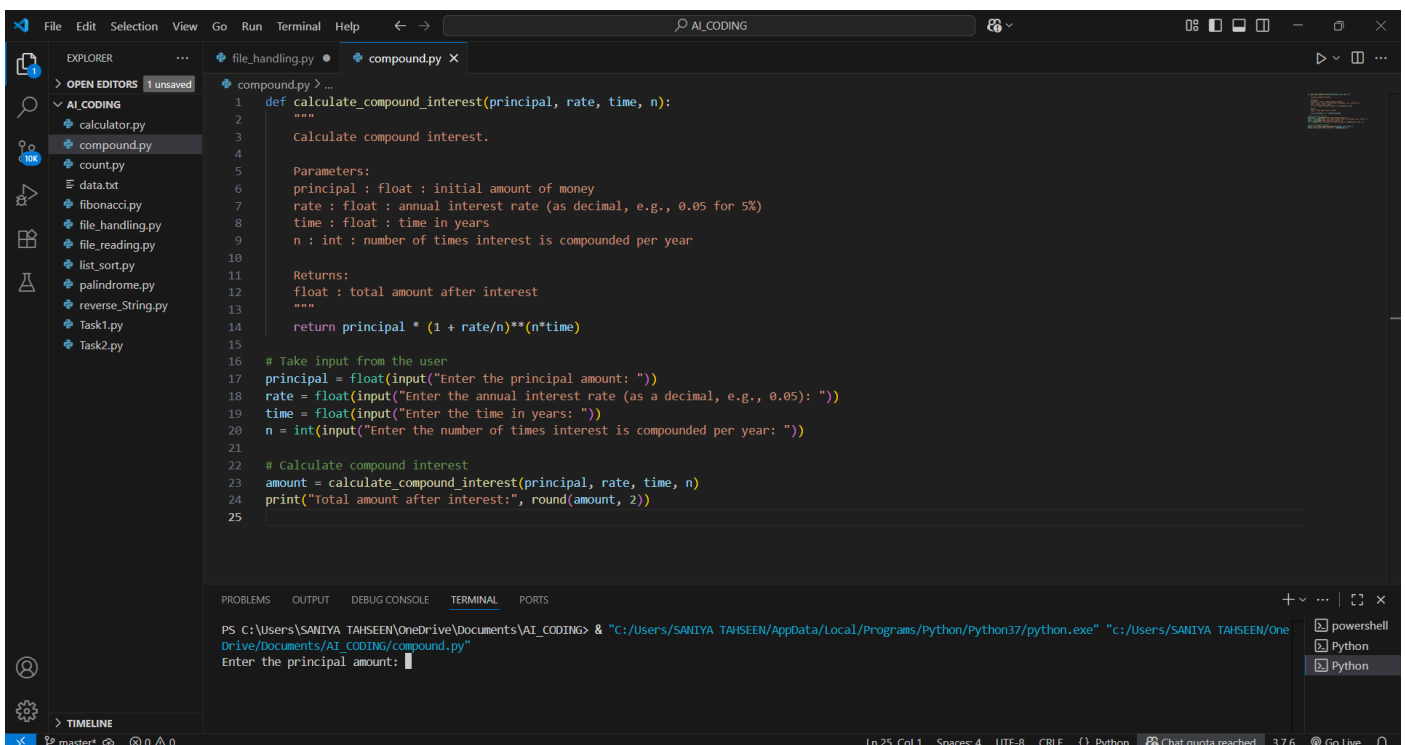
2503A51L18

TASK #1:

Prompt Used:

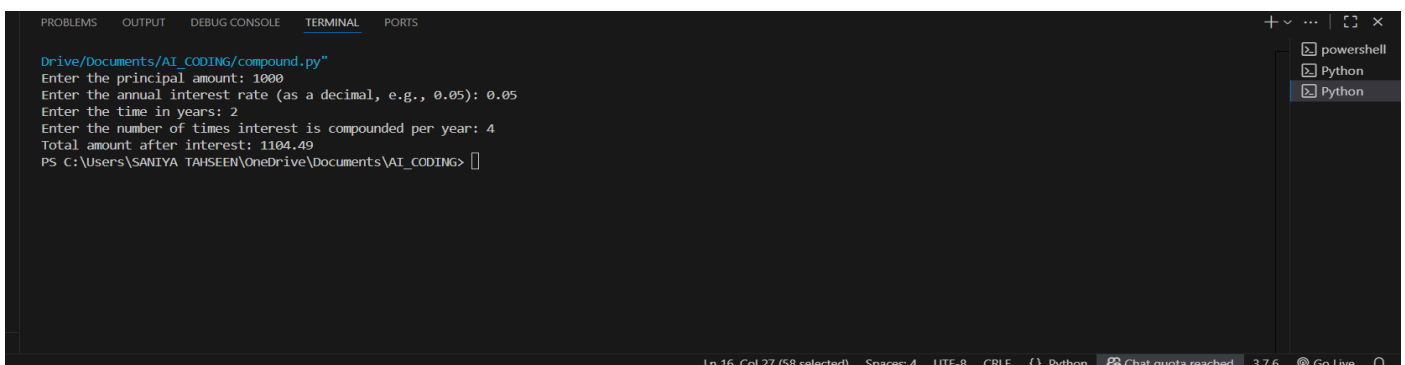
Ask AI to write a function to calculate compound interest, starting with only the function name. Then add a docstring, then input-output example.

Code Generated:



```
File Edit Selection View Go Run Terminal Help  AI_CODING
EXPLORER 1 unsaved
  AI_CODING
    calculator.py
    compound.py
    count.py
    data.txt
    fibonacci.py
    file_handling.py
    file_reading.py
    list_sort.py
    palindrome.py
    reverse_String.py
    Task1.py
    Task2.py
  compound.py x
1 def calculate_compound_interest(principal, rate, time, n):
2     """
3     Calculate compound interest.
4
5     Parameters:
6     principal : float : initial amount of money
7     rate : float : annual interest rate (as decimal, e.g., 0.05 for 5%)
8     time : float : time in years
9     n : int : number of times interest is compounded per year
10
11     Returns:
12     float : total amount after interest
13     """
14     return principal * (1 + rate/n)**(n*time)
15
16 # Take input from the user
17 principal = float(input("Enter the principal amount: "))
18 rate = float(input("Enter the annual interest rate (as a decimal, e.g., 0.05): "))
19 time = float(input("Enter the time in years: "))
20 n = int(input("Enter the number of times interest is compounded per year: "))
21
22 # Calculate compound interest
23 amount = calculate_compound_interest(principal, rate, time, n)
24 print("Total amount after interest:", round(amount, 2))
25
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\SANIYA TAHSEEN\OneDrive\Documents\AI_CODING> "c:/Users/SANIYA TAHSEEN/OneDrive/Documents/AI_CODING/compound.py"
Enter the principal amount: 1000
Enter the annual interest rate (as a decimal, e.g., 0.05): 0.05
Enter the time in years: 2
Enter the number of times interest is compounded per year: 4
Total amount after interest: 1104.49
PS C:\Users\SANIYA TAHSEEN\OneDrive\Documents\AI_CODING>
```

Output After executing Code:



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Drive/Documents/AI_CODING/compound.py"
Enter the principal amount: 1000
Enter the annual interest rate (as a decimal, e.g., 0.05): 0.05
Enter the time in years: 2
Enter the number of times interest is compounded per year: 4
Total amount after interest: 1104.49
PS C:\Users\SANIYA TAHSEEN\OneDrive\Documents\AI_CODING>
```

Observations:

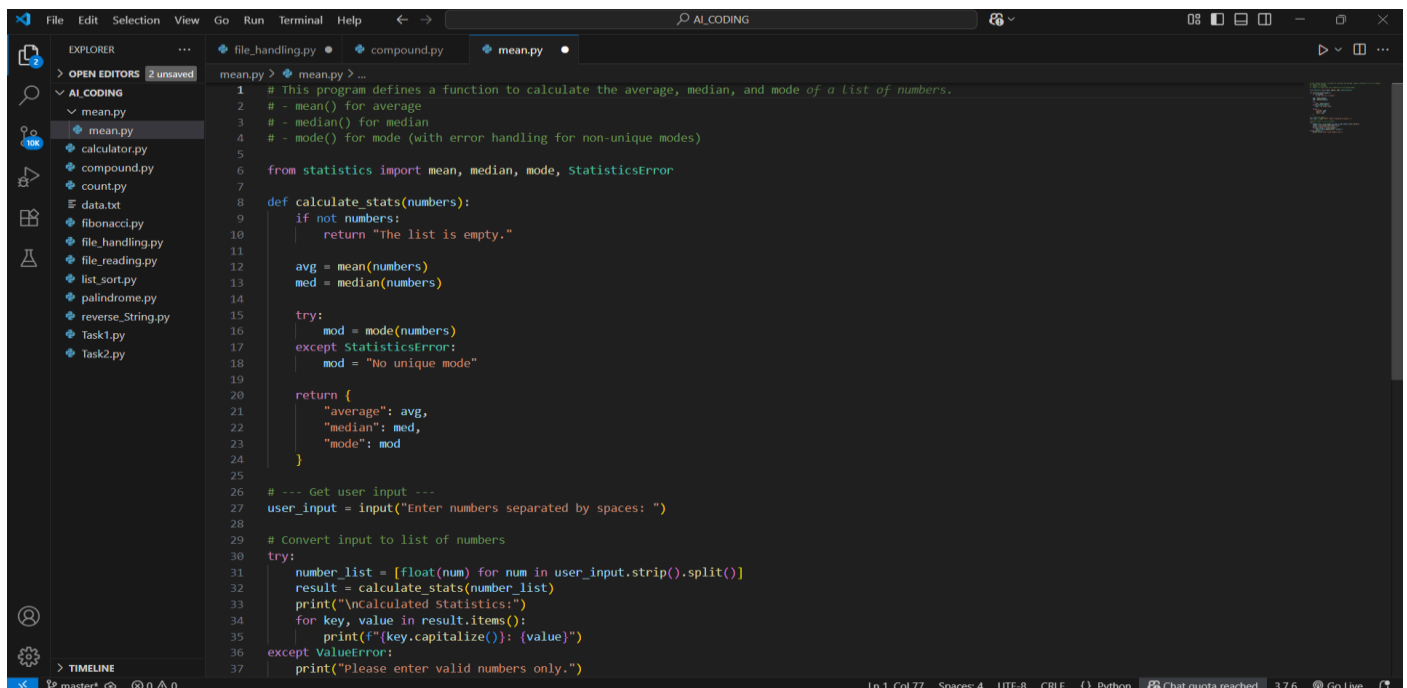
- The code correctly implements the compound interest formula using inputs (principal, rate, time, compounds per year) and returns the calculated final amount.
- The docstring explains the function with parameter details, return type, and an example, making the code clear and professional.

TASK #2:

Prompt Used:

Do math stuff, then refine it to: # Write a function to calculate average, median, and mode of a list of numbers.

Code Generated:



```
1 # This program defines a function to calculate the average, median, and mode of a list of numbers.
2 # - mean() for average
3 # - median() for median
4 # - mode() for mode (with error handling for non-unique modes)
5
6 from statistics import mean, median, mode, StatisticsError
7
8 def calculate_stats(numbers):
9     if not numbers:
10         return "The list is empty."
11
12     avg = mean(numbers)
13     med = median(numbers)
14
15     try:
16         mod = mode(numbers)
17     except StatisticsError:
18         mod = "No unique mode"
19
20     return {
21         "average": avg,
22         "median": med,
23         "mode": mod
24     }
25
26 # --- Get user input ---
27 user_input = input("Enter numbers separated by spaces: ")
28
29 # Convert input to list of numbers
30 try:
31     number_list = [float(num) for num in user_input.strip().split()]
32     result = calculate_stats(number_list)
33     print("\nCalculated Statistics:")
34     for key, value in result.items():
35         print(f"{key.capitalize()}: {value}")
36 except ValueError:
37     print("Please enter valid numbers only.")
```

Output After executing Code:



```
Mode: 3.0
PS C:\Users\SANIYA TAHSEEN\OneDrive\Documents\AI_CODING> & "C:/Users/SANIYA TAHSEEN/AppData/Local/Programs/Python/Python37/python.exe" "c:/Users/SANIYA TAHSEEN/OneDrive/Documents/AI_CODING/mean.py/mean.py"
Enter numbers separated by spaces: 5 3 8 6

Calculated Statistics:
Average: 5.5
Median: 5.5
Mode: No unique mode
PS C:\Users\SANIYA TAHSEEN\OneDrive\Documents\AI_CODING> |
```

Observations:

- The program defines a function to calculate the average, median, and mode of a list of numbers using Python's built-in statistics functions: mean(), median(), and mode().
- A try-except block handles cases where there is no unique mode, preventing errors.

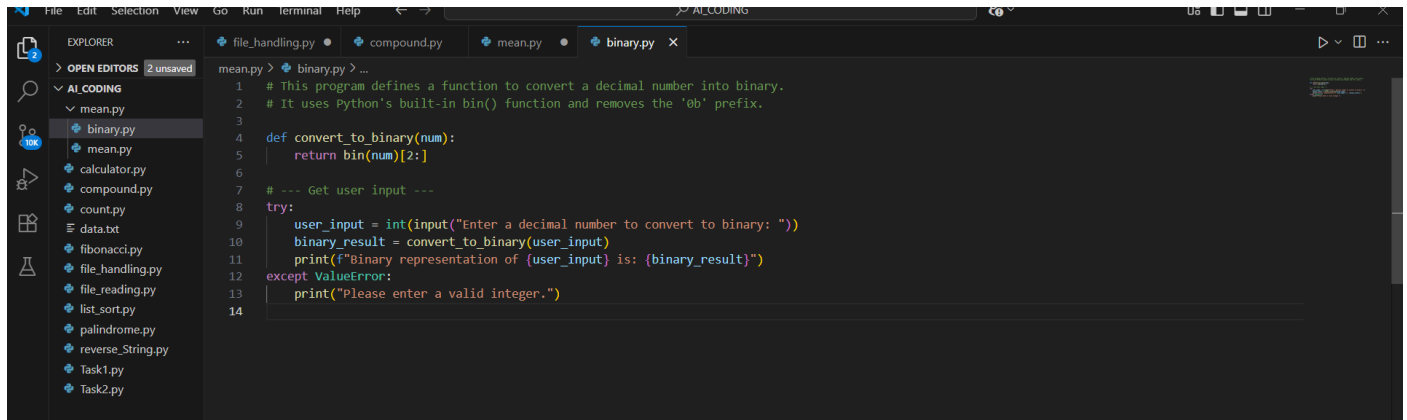
- User input is taken as a space-separated string, converted into floats, and results are displayed in a structured dictionary format.

TASK #3:

Prompt Used:

Provide multiple examples of input-output to the AI for convert_to_binary(num) function. Observe how AI uses few-shot prompting to generalize.

Code Generated:

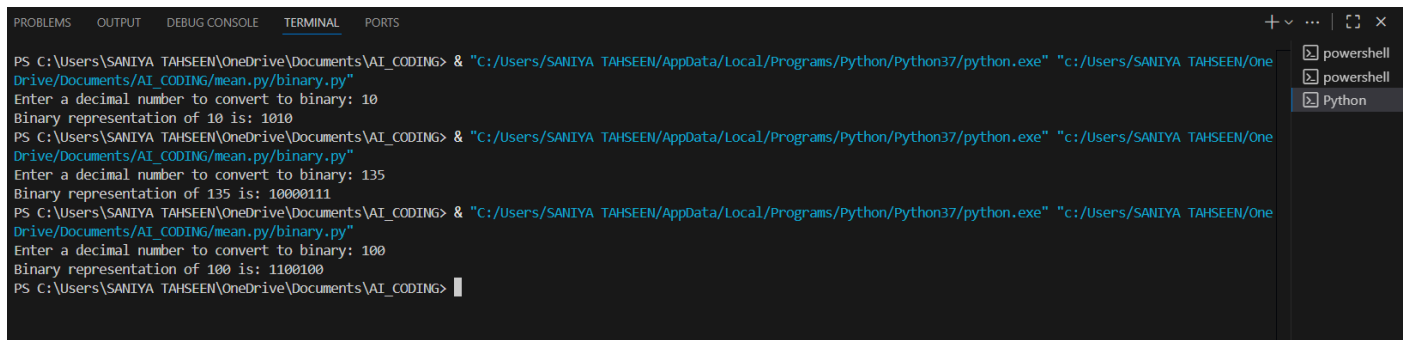


```

1 # This program defines a function to convert a decimal number into binary.
2 # It uses Python's built-in bin() function and removes the '0b' prefix.
3
4 def convert_to_binary(num):
5     return bin(num)[2:]
6
7 # --- Get user input ---
8 try:
9     user_input = int(input("Enter a decimal number to convert to binary: "))
10    binary_result = convert_to_binary(user_input)
11    print(f"Binary representation of {user_input} is: {binary_result}")
12 except ValueError:
13     print("Please enter a valid integer.")
14

```

Output After executing Code:



```

PS C:\Users\SANIYA TAHSEEN\OneDrive\Documents\AI_CODING> & "C:/Users/SANIYA TAHSEEN/AppData/Local/Programs/Python/Python37/python.exe" "C:/Users/SANIYA TAHSEEN/OneDrive/Documents/AI_CODING/mean.py/binary.py"
Enter a decimal number to convert to binary: 10
Binary representation of 10 is: 1010
PS C:\Users\SANIYA TAHSEEN\OneDrive\Documents\AI_CODING> & "C:/Users/SANIYA TAHSEEN/AppData/Local/Programs/Python/Python37/python.exe" "C:/Users/SANIYA TAHSEEN/OneDrive/Documents/AI_CODING/mean.py/binary.py"
Enter a decimal number to convert to binary: 135
Binary representation of 135 is: 10000111
PS C:\Users\SANIYA TAHSEEN\OneDrive\Documents\AI_CODING> & "C:/Users/SANIYA TAHSEEN/AppData/Local/Programs/Python/Python37/python.exe" "C:/Users/SANIYA TAHSEEN/OneDrive/Documents/AI_CODING/mean.py/binary.py"
Enter a decimal number to convert to binary: 100
Binary representation of 100 is: 1100100
PS C:\Users\SANIYA TAHSEEN\OneDrive\Documents\AI_CODING>

```

Observations:

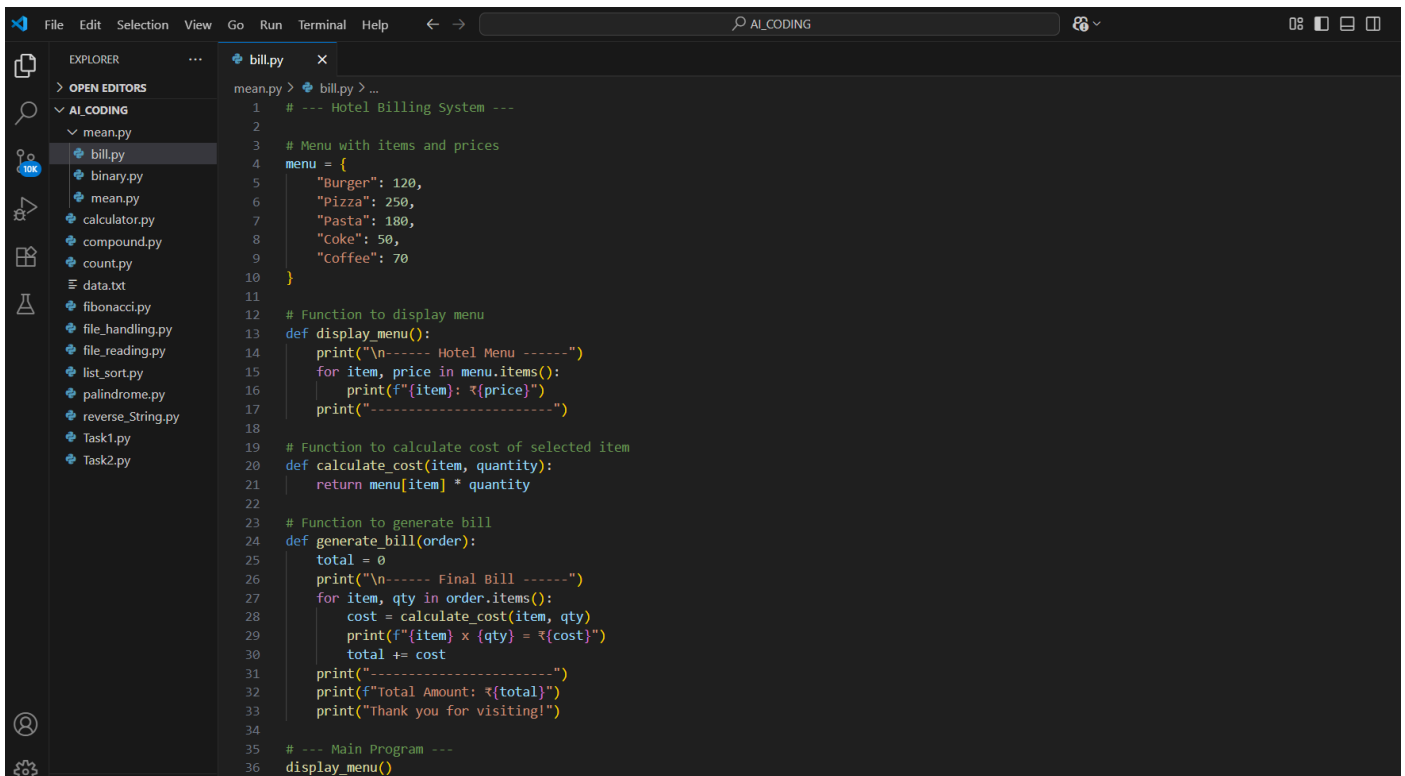
- bin(num) converts a decimal number to a binary string prefixed with '0b'.
- Using [2:] slices off the '0b' to return only the binary digits.
- The function uses a try-except block to check whether the user's input is a valid number and to handle errors if it is not.

TASK #4:

Prompt Used:

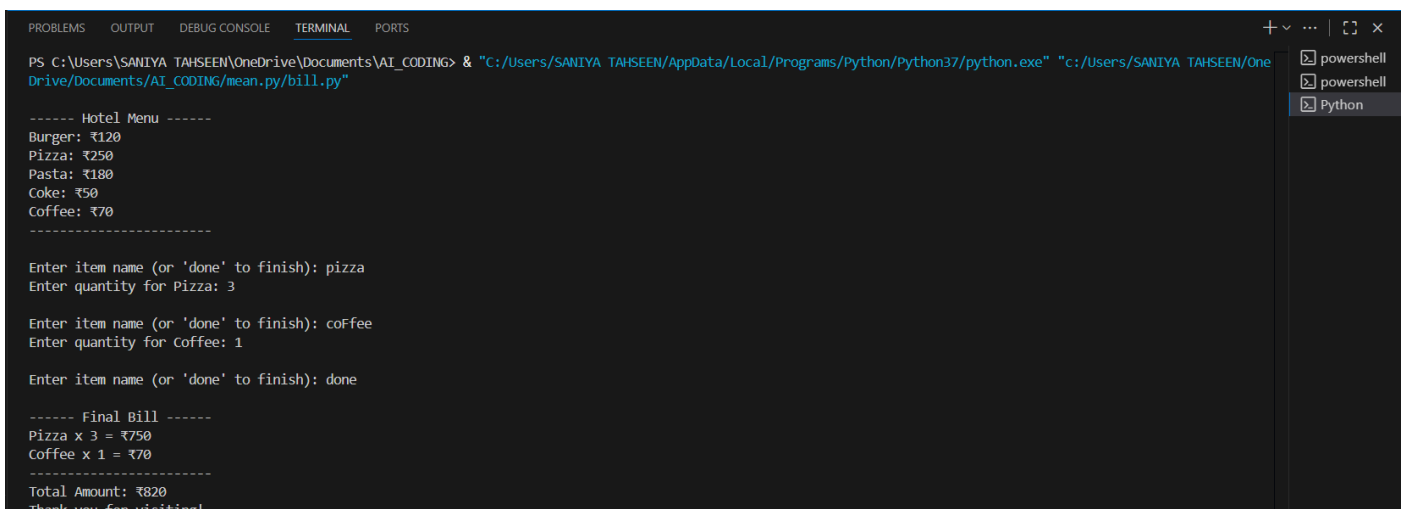
Create an user interface for an hotel to generate bill based on customer requirements

Code Generated:



```
1  # --- Hotel Billing System ---
2
3  # Menu with items and prices
4  menu = {
5      "Burger": 120,
6      "Pizza": 250,
7      "Pasta": 180,
8      "Coke": 50,
9      "Coffee": 70
10 }
11
12 # Function to display menu
13 def display_menu():
14     print("\n----- Hotel Menu -----")
15     for item, price in menu.items():
16         print(f"{item}: ₹{price}")
17     print("-----")
18
19 # Function to calculate cost of selected item
20 def calculate_cost(item, quantity):
21     return menu[item] * quantity
22
23 # Function to generate bill
24 def generate_bill(order):
25     total = 0
26     print("\n----- Final Bill -----")
27     for item, qty in order.items():
28         cost = calculate_cost(item, qty)
29         print(f"{item} x {qty} = ₹{cost}")
30         total += cost
31     print("-----")
32     print(f"Total Amount: ₹{total}")
33     print("Thank you for visiting!")
34
35 # --- Main Program ---
36 display_menu()
```

Output After executing Code:



```
PS C:\Users\SANIYA TAHSEEN\OneDrive\Documents\AI_CODING> & "C:/Users/SANIYA TAHSEEN/AppData/Local/Programs/Python/Python37/python.exe" "c:/Users/SANIYA TAHSEEN/OneDrive/Documents/AI_CODING/mean.py/bill.py"

----- Hotel Menu -----
Burger: ₹120
Pizza: ₹250
Pasta: ₹180
Coke: ₹50
Coffee: ₹70
-----

Enter item name (or 'done' to finish): pizza
Enter quantity for Pizza: 3

Enter item name (or 'done' to finish): coffee
Enter quantity for Coffee: 1

Enter item name (or 'done' to finish): done

----- Final Bill -----
Pizza x 3 = ₹750
Coffee x 1 = ₹70
-----
Total Amount: ₹820
Thank you for visiting!
```

Observations:

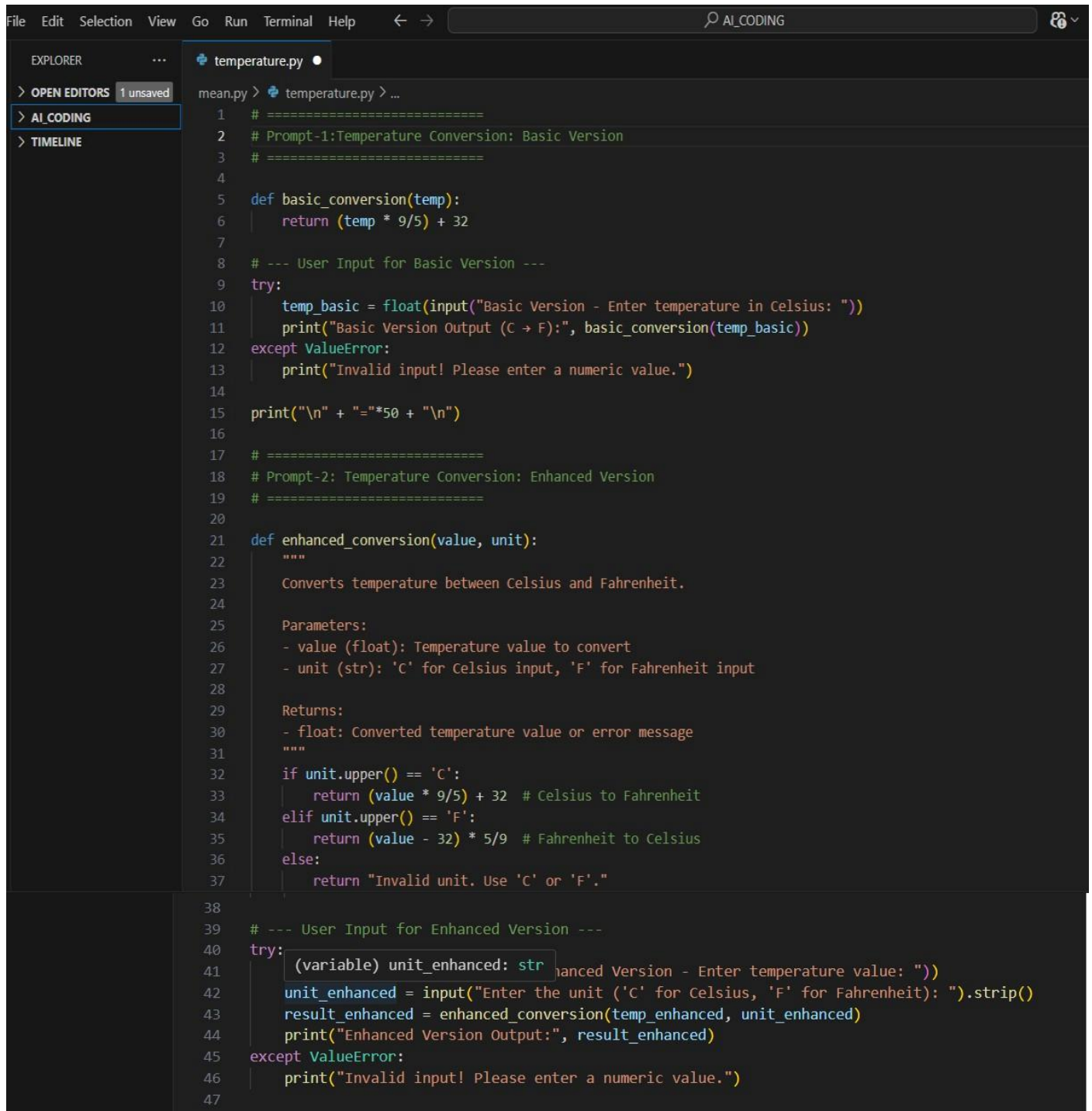
- The menu is stored as a dictionary with items as keys (capitalized) and their prices as values.
- User input is processed with `.capitalize()` so it matches the menu keys, making input case-insensitive for practical purposes.
- Separate functions are defined for repeated tasks like displaying the menu and generating the bill, ensuring code reusability.

TASK #5:

Prompt Used:

Analyzing Prompt Specificity: Improving Temperature Conversion Function with Clear Instructions

Code Generated:

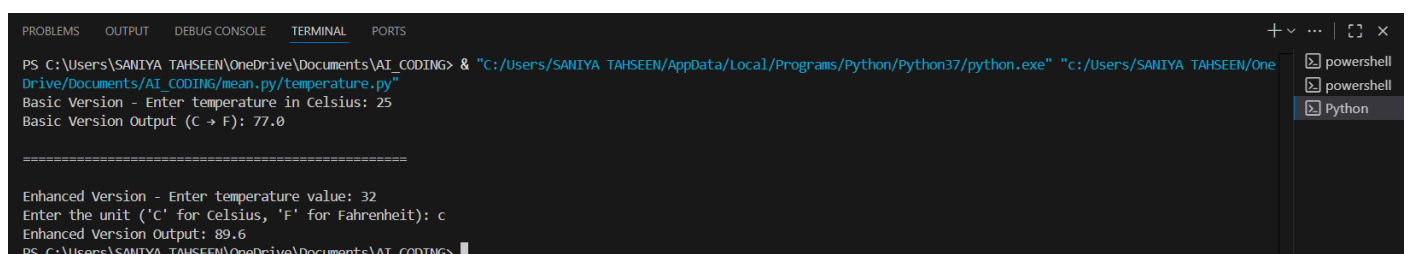


```
File Edit Selection View Go Run Terminal Help ← → AI_CODING

EXPLORER
> OPEN EDITORS 1 unsaved
> AI_CODING
> TIMELINE

mean.py > temperature.py > ...
1 # =====
2 # Prompt-1: Temperature Conversion: Basic Version
3 # =====
4
5 def basic_conversion(temp):
6     return (temp * 9/5) + 32
7
8 # --- User Input for Basic Version ---
9 try:
10     temp_basic = float(input("Basic Version - Enter temperature in Celsius: "))
11     print("Basic Version Output (C → F):", basic_conversion(temp_basic))
12 except ValueError:
13     print("Invalid input! Please enter a numeric value.")
14
15 print("\n" + "="*50 + "\n")
16
17 # =====
18 # Prompt-2: Temperature Conversion: Enhanced Version
19 # =====
20
21 def enhanced_conversion(value, unit):
22     """
23     Converts temperature between Celsius and Fahrenheit.
24
25     Parameters:
26     - value (float): Temperature value to convert
27     - unit (str): 'C' for Celsius input, 'F' for Fahrenheit input
28
29     Returns:
30     - float: Converted temperature value or error message
31     """
32     if unit.upper() == 'C':
33         return (value * 9/5) + 32 # Celsius to Fahrenheit
34     elif unit.upper() == 'F':
35         return (value - 32) * 5/9 # Fahrenheit to Celsius
36     else:
37         return "Invalid unit. Use 'C' or 'F'."
38
39 # --- User Input for Enhanced Version ---
40 try:
41     temp_enhanced = float(input("Enhanced Version - Enter temperature value: "))
42     unit_enhanced = input("Enter the unit ('C' for Celsius, 'F' for Fahrenheit): ").strip()
43     result_enhanced = enhanced_conversion(temp_enhanced, unit_enhanced)
44     print("Enhanced Version Output:", result_enhanced)
45 except ValueError:
46     print("Invalid input! Please enter a numeric value.")
47
```

Output After executing Code:



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\SANIYA TAHSEEN\OneDrive\Documents\AI_CODING> & "C:/Users/SANIYA TAHSEEN/AppData/Local/Programs/Python/Python37/python.exe" "C:/Users/SANIYA TAHSEEN/OneDrive/Documents/AI_CODING/mean.py/temperature.py"
Basic Version - Enter temperature in Celsius: 25
Basic Version Output (C → F): 77.0

=====

Enhanced Version - Enter temperature value: 32
Enter the unit ('C' for Celsius, 'F' for Fahrenheit): c
Enhanced Version Output: 89.6
PS C:\Users\SANIYA TAHSEEN\OneDrive\Documents\AI_CODING>
```

Observations:

Prompt-1: Temperature Conversion: Basic Version

- Handles only Celsius → Fahrenheit conversion
- Code is simple, minimal, and easy to follow
- No error handling

Prompt-2: Temperature Conversion: Enhanced Version

- Converts both Celsius ↔ Fahrenheit based on user input unit.
- Includes docstring, comments, and validation for invalid inputs.
- Robust, readable, and user-friendly.