

AI ASSISTED CODING LAB

ASSIGNMENT 3.2

ENROLLMENT NO :2503A51L15

BATCH NO: 19

NAME: MOHAMMAD KHAJA AFZALUDDIN

TASK1

TASK1 DESCRIPTION:- Ask AI to write a function to calculate compound interest, starting with only the function name. Then add a docstring, then input-output example

PROMPT:- "Write a Python function to calculate compound interest. Start by providing only the function name. Next, add a docstring that explains what the function does, its parameters, and its return value. Then, include an input-output example that demonstrates how to use the function and what result it produces." implement with example

CODE:-

```
1 # Step 1: Function name only
2 def calculate_compound_interest():
3     pass
4
5 # Step 2: Add docstring
6
7 def calculate_compound_interest(principal, rate, times_compounded, years):
8     """
9     Calculate the compound interest for a given principal, interest rate, number of times interest is compounded per year, and number of years.
10
11     Parameters:
12     principal (float): The initial amount of money.
13     rate (float): The annual interest rate (as a decimal, e.g., 0.05 for 5%).
14     times_compounded (int): Number of times interest is compounded per year.
15     years (float): Number of years the money is invested for.
16
17     Returns:
18     float: The amount after interest is applied.
19     """
20     amount = principal * (1 + rate / times_compounded) ** (times_compounded * years)
21     return amount
22
23 # Step 3: Input-output example
24 if __name__ == "__main__":
25     # Example: $1000 at 5% interest, compounded quarterly, for 3 years
26     result = calculate_compound_interest(1000, 0.05, 4, 3)
27     print(f"Compound interest result: ${result:.2f}")
28
```

OUTPUT:-

```
PS C:\Users\khaja\Downloads\a3> & C:/Users/khaja/anaconda3/python.exe c:/Users/khaja/Downloads/a3/t1.py
Compound interest result: $1160.75
PS C:\Users\khaja\Downloads\a3>
```

TASK2

TASK2 DESCRIPTION:- Do math stuff, then refine it to: # Write a function to calculate average, median, and mode of a list of numbers

PROMPT:- generate a python function to calculate average, median, and mode of a list of numbers that should be enter at runtime

CODE:-

```
1  import statistics
2
3  def calculate_average(numbers):
4      """Return the average (mean) of a list of numbers."""
5      return sum(numbers) / len(numbers) if numbers else None
6
7  def calculate_median(numbers):
8      """Return the median of a list of numbers."""
9      return statistics.median(numbers) if numbers else None
10
11 def calculate_mode(numbers):
12     """Return the mode of a list of numbers."""
13     try:
14         return statistics.mode(numbers)
15     except statistics.StatisticsError:
16         return None
17
18 if __name__ == "__main__":
19     user_input = input("Enter numbers separated by spaces: ")
20     numbers = [float(x) for x in user_input.split()]
21     print(f"Average: {calculate_average(numbers)}")
22     print(f"Median: {calculate_median(numbers)}")
23     mode = calculate_mode(numbers)
24     if mode is not None:
25         print(f"Mode: {mode}")
26     else:
27         print("Mode: No unique mode found.")
28
```

OUTPUT:-

```
Enter numbers separated by spaces: 1 2 2 3 4 5
Average: 2.8333333333333335
Median: 2.5
Mode: 2.0
PS C:\Users\khaja\Downloads\ a3> 
```

TASK3

TASK3 DESCRIPTION:- Provide multiple examples of input-output to the AI for convert_to_binary(num) function. Observe how AI uses few-shot prompting to generalize.

PROMPT:- generate a python program that provides multiple input–output examples for a Python function `convert_to_binary(num)` that converts a decimal number into its binary representation

CODE:-

```
1 def convert_to_binary(num):
2     """Convert a decimal number to its binary representation as a string."""
3     return bin(num)[2:]
4
5 # Multiple input-output examples
6 def show_examples():
7     examples = [2, 7, 12, 20, 31, 45]
8     print("Input\tOutput")
9     for n in examples:
10        print(f"{n}\t{convert_to_binary(n)}")
11
12 if __name__ == "__main__":
13     show_examples()
14     # You can also test with your own input:
15     num = int(input("Enter a decimal number: "))
16     print(f"Binary representation: {convert_to_binary(num)}")
17
```

OUTPUT:-

```
PS C:\Users\khaja\Downloads\a3> & C:/Users/khaja/anaconda3/python.exe c:/Users/khaja/Downloads/a3/t3.py
Input    Output
2        10
7        111
12       1100
20       10100
31       11111
45       101101
Enter a decimal number: 7
Binary representation: 111
PS C:\Users\khaja\Downloads\a3> 
```

TASK4

TASK4 DESCRIPTION:-Create an user interface for an hotel to generate bill based on customer requirements

PROMPT:- Generate a python program to Create an user interface for an hotel to generate bill based on customer requirements

CODE:-

```

1  def display_menu():
2      print("Welcome to Pythonic Hotel!")
3      print("Menu:")
4      print("1. Single Room - $100 per night")
5      print("2. Double Room - $180 per night")
6      print("3. Suite - $300 per night")
7      print("4. Breakfast - $20 per person per day")
8      print("5. Dinner - $35 per person per day")
9      print()
10
11 def get_room_price(room_type):
12     if room_type == 1:
13         return 100
14     elif room_type == 2:
15         return 180
16     elif room_type == 3:
17         return 300
18     else:
19         return 0
20
21 def main():
22     display_menu()
23     name = input("Enter customer name: ")
24     nights = int(input("Number of nights: "))
25     room_type = int(input("Room type (1-Single, 2-Double, 3-Suite): "))
26     num_people = int(input("Number of people: "))
27     breakfast = input("Add breakfast? (y/n): ").lower() == 'y'
28     dinner = input("Add dinner? (y/n): ").lower() == 'y'
29
30     room_cost = get_room_price(room_type) * nights
31     breakfast_cost = 20 * num_people * nights if breakfast else 0
32     dinner_cost = 35 * num_people * nights if dinner else 0
33     total = room_cost + breakfast_cost + dinner_cost
34
35     print("\n--- Bill ---")
36     print(f"Customer: {name}")
37     print(f"Room cost: ${room_cost}")
38     if breakfast:
39         print(f"Breakfast: ${breakfast_cost}")
40     if dinner:
41         print(f"Dinner: ${dinner_cost}")
42     print(f"Total: ${total}")
43     print("Thank you for staying with us!")
44
45 if __name__ == "__main__":
46     main()
47

```

```

PS C:\Users\khaja\Downloads\a3> & C:/Users/khaja/anaconda3/python.exe c:/Users/khaja/Downloads/a3/t4.py
Welcome to Pythonic Hotel!
Menu:
1. Single Room - $100 per night
2. Double Room - $180 per night
3. Suite - $300 per night
4. Breakfast - $20 per person per day
5. Dinner - $35 per person per day

Enter customer name: Afzal
Number of nights: 2
Room type (1-Single, 2-Double, 3-Suite): 1
Number of people: 1
Room type (1-Single, 2-Double, 3-Suite): 1
Number of people: 1
Number of people: 1
Add breakfast? (y/n): y
Add breakfast? (y/n): y
Add dinner? (y/n): y
Add dinner? (y/n): y

--- Bill ---
Customer: Afzal
Room cost: $200
Breakfast: $40
Dinner: $70
Total: $310
Room cost: $200
Breakfast: $40
Dinner: $70
Total: $310
Breakfast: $40
Dinner: $70
Total: $310
Dinner: $70
Total: $310
Total: $310
Thank you for staying with us!
PS C:\Users\khaja\Downloads\a3> 

```

TASK5

TASK5 DESCRIPTION:- Analyzing Prompt Specificity: Improving Temperature Conversion Function with Clear Instructions

PROMPT:- Generate a python program to Improving Temperature Conversion Function with Clear Instructions

CODE:-

```
1 def convert_temperature(value, from_unit, to_unit):
2     """
3     Convert temperature between Celsius, Fahrenheit, and Kelvin.
4
5     Parameters:
6     value (float): The temperature value to convert.
7     from_unit (str): The unit to convert from ('C', 'F', 'K').
8     to_unit (str): The unit to convert to ('C', 'F', 'K').
9
10    Returns:
11    float: The converted temperature value.
12    """
13    if from_unit == to_unit:
14        return value
15    # Celsius to Fahrenheit
16    if from_unit == 'C' and to_unit == 'F':
17        return value * 9/5 + 32
18    # Celsius to Kelvin
19    if from_unit == 'C' and to_unit == 'K':
20        return value + 273.15
21    # Fahrenheit to Celsius
22    if from_unit == 'F' and to_unit == 'C':
23        return (value - 32) * 5/9
24    # Fahrenheit to Kelvin
25    if from_unit == 'F' and to_unit == 'K':
26        return (value - 32) * 5/9 + 273.15
27    # Kelvin to Celsius
28    if from_unit == 'K' and to_unit == 'C':
29        return value - 273.15
30    # Kelvin to Fahrenheit
31    if from_unit == 'K' and to_unit == 'F':
32        return (value - 273.15) * 9/5 + 32
33    raise ValueError("Invalid temperature units.")
34
35 if __name__ == "__main__":
36     print("Temperature Converter")
37     print("Enter the temperature value and units.")
38     print("Units: C for Celsius, F for Fahrenheit, K for Kelvin")
39     value = float(input("Temperature value: "))
40     from_unit = input("Convert from (C/F/K): ").strip().upper()
41     to_unit = input("Convert to (C/F/K): ").strip().upper()
42     try:
43         result = convert_temperature(value, from_unit, to_unit)
44         print(f"{value} {from_unit} = {result:.2f} {to_unit}")
45     except ValueError as e:
46         print(f"Error: {e}")
47
```

OUTPUT:-

```
Temperature value: 32
Convert from (C/F/K): c
Temperature value: 32
Convert from (C/F/K): c
Convert from (C/F/K): c
Convert to (C/F/K): k
32.0 C = 305.15 K
PS C:\Users\khaja\Downloads\a3>
```

OBSERVATION:- From this assignment, I observed the practical role of prompt engineering and AI-assisted coding in generating Python programs. By giving different levels of instructions, the AI was able to produce complete implementations, examples, and even user interfaces.

- In Task 1, starting with only a function name and gradually adding docstrings and examples demonstrated how AI understands step-by-step instructions and builds code systematically.
- In Task 2, I observed how runtime inputs can be used for statistical calculations (average, median, mode), showing AI's capability to handle mathematical logic on user-provided data.
- In Task 3, by providing multiple input–output examples for the `convert_to_binary(num)` function, I noticed how AI applied few-shot prompting to generalize and generate correct binary conversions for any decimal input.
- In Task 4, the hotel billing program highlighted how AI can extend beyond simple functions to build user-oriented applications, combining logic with interface design.
- In Task 5, refining the temperature conversion function showed how prompt specificity directly affects the accuracy, clarity, and usability of AI-generated code.