

NAME: B.VYSHNAVI
ROLL NO.: 2403A510C8
BATCH : 04

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
ProgramName: B. Tech		Assignment Type: Lab	AcademicYear:2025-2026
CourseCoordinatorName		Venkataramana Veeramsetty	
Instructor(s)Name		Dr. Mohammed Ali Shaik Dr. T Sampath Kumar Mr. S Naresh Kumar Dr. V. Rajesh Dr. Brij Kishore Dr Pramoda Patro Dr. Venkataramana Dr. Ravi Chander Dr. Jagjeeth Singh	
CourseCode	24CS002PC215	CourseTitle	AI Assisted Coding
Year/Sem	II/I	Regulation	R24
Date and Day of Assignment	Week2-Tuesday	Time(s)	
Duration	2 Hours	Applicableto Batches	24CSBTB01 To 24CSBTB39
AssignmentNumber: 3.2(Present assignment number)/24(Total number of assignments)			
Q.No.	Question	Expected Time to complete	
1	<p>Lab 3: Prompt Engineering – Improving Prompts and Context Management</p> <p>Lab Objectives:</p> <p>To understand how prompt structure and wording influence AI-generated code. To explore how context (like comments and function names) helps AI generate relevant output. To evaluate the quality and accuracy of code based on prompt clarity. To develop effective prompting strategies for AI-assisted programming.</p> <p>Lab Outcomes (LOs): After completing this lab, students will be able to:</p> <p>Generate Python code using Google Gemini in Google Colab. Analyze the effectiveness of code explanations and suggestions by Gemini. Set up and use Cursor AI for AI-powered coding assistance.</p>	03.08.2025 EOD	

NAME: B.VYSHNAVI
ROLL NO.: 2403A510C8
BATCH : 04

- Evaluate and refactor code using Cursor AI features.
- Compare AI tool behavior and code quality across different platforms.

Task Description#1

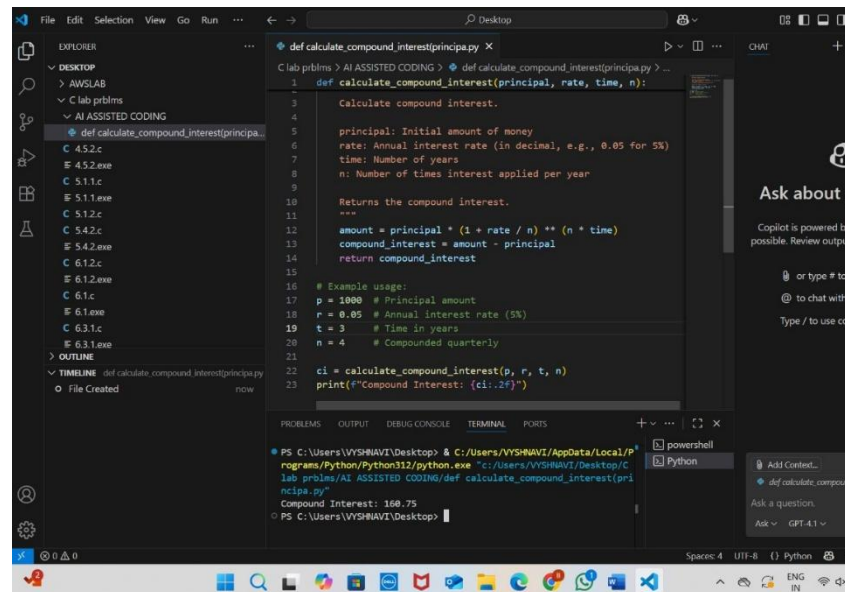
- Ask AI to write a function to calculate compound interest, starting with only the function name. Then add a docstring, then input-output example

Expected Output#1

- Comparison of AI-generated code styles

PROMPT1:

Give a python code to calculate the compound interest using functions starting with only function name.



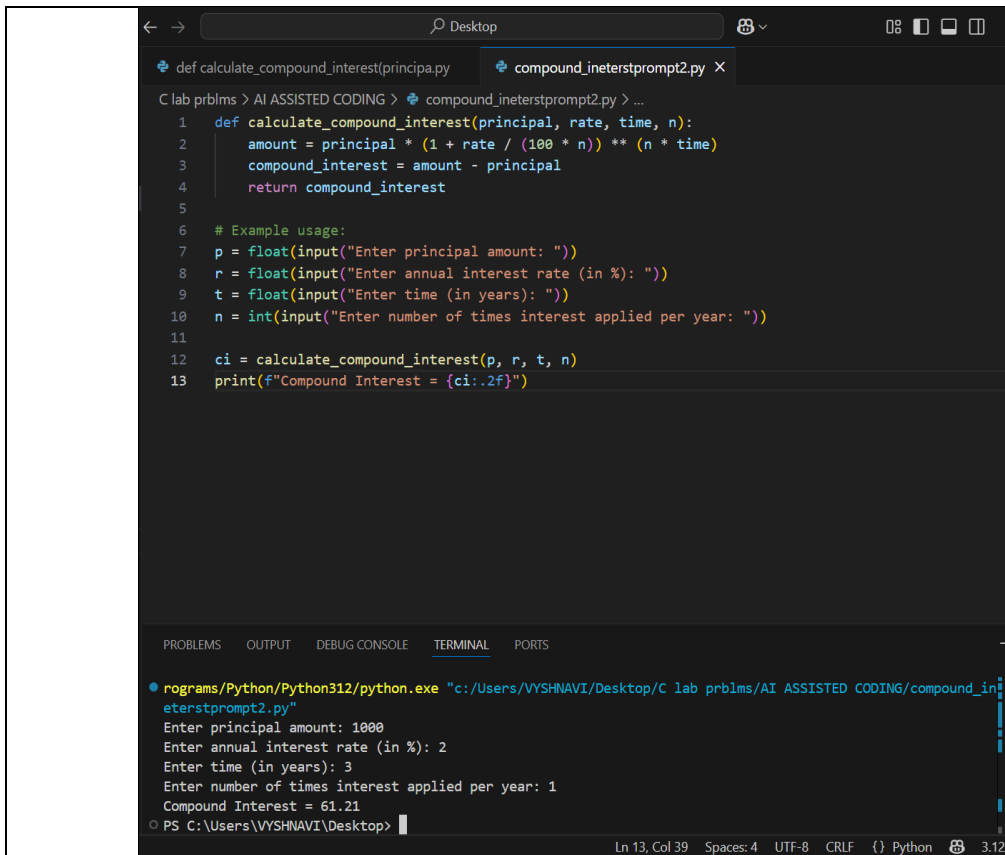
The screenshot shows a code editor with a file explorer on the left and a chat panel on the right. The main editor displays a Python function named `calculate_compound_interest` with a docstring and example usage. The chat panel shows a prompt and the AI's response.

```
def calculate_compound_interest(principal, rate, time, n):  
    """  
    Calculate compound interest.  
    principal: Initial amount of money  
    rate: Annual interest rate (in decimal, e.g., 0.05 for 5%)  
    time: Number of years  
    n: Number of times interest applied per year  
    Returns the compound interest.  
    """  
    amount = principal * (1 + rate / n) ** (n * time)  
    compound_interest = amount - principal  
    return compound_interest  
  
# Example usage:  
p = 1000 # Principal amount  
r = 0.05 # Annual interest rate (5%)  
t = 3 # Time in years  
n = 4 # Compounded quarterly  
  
ci = calculate_compound_interest(p, r, t, n)  
print(f"Compound Interest: {ci:.2f}")
```

PROMPT2:

write a python code to calculate compound interest using function name

NAME: B.VYSHNAVI
ROLL NO.: 2403A510C8
BATCH : 04

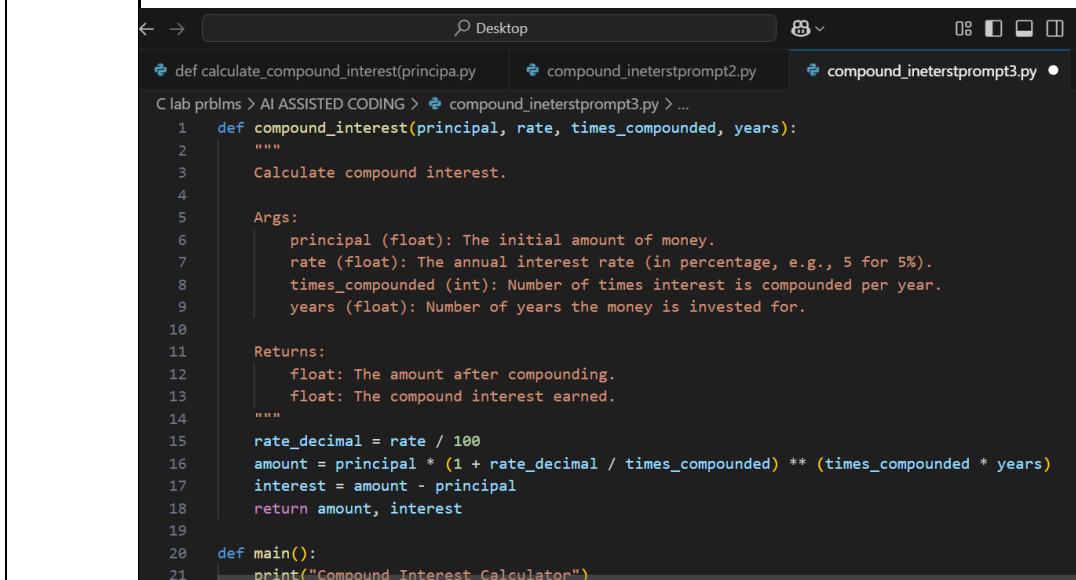


The screenshot shows a code editor with a file named `compound_ineterstprompt2.py`. The code defines a function `calculate_compound_interest` that takes `principal`, `rate`, `time`, and `n` as arguments. It calculates the compound interest by first determining the total amount using the formula $\text{amount} = \text{principal} * (1 + \text{rate} / (100 * n)) ** (n * \text{time})$, then subtracting the principal to get the compound interest. An example usage is provided with input prompts for principal amount, annual interest rate, time in years, and number of times interest is applied per year. The terminal output shows the execution with inputs 1000, 2, 3, and 1, resulting in a compound interest of 61.21.

```
def calculate_compound_interest(principal, rate, time, n):  
    amount = principal * (1 + rate / (100 * n)) ** (n * time)  
    compound_interest = amount - principal  
    return compound_interest  
  
# Example usage:  
p = float(input("Enter principal amount: "))  
r = float(input("Enter annual interest rate (in %): "))  
t = float(input("Enter time (in years): "))  
n = int(input("Enter number of times interest applied per year: "))  
  
ci = calculate_compound_interest(p, r, t, n)  
print(f"Compound Interest = {ci:.2f}")
```

Terminal Output:
C:\Users\B.VYSHNAVI\Desktop> python.exe "c:/Users/B.VYSHNAVI/Desktop/C lab prblms/AI ASSISTED CODING/compound_ineterstprompt2.py"
Enter principal amount: 1000
Enter annual interest rate (in %): 2
Enter time (in years): 3
Enter number of times interest applied per year: 1
Compound Interest = 61.21

PROMPT3:
write a complex code to calculate the compound interest using functions in python.



The screenshot shows a code editor with a file named `compound_ineterstprompt3.py`. The code defines a function `compound_interest` with a detailed docstring. The docstring includes a description of the function, its arguments (`principal`, `rate`, `times_compounded`, `years`), and its returns (the amount after compounding and the compound interest earned). The function calculates the compound interest by first determining the total amount using the formula $\text{amount} = \text{principal} * (1 + \text{rate_decimal} / \text{times_compounded}) ** (\text{times_compounded} * \text{years})$, then subtracting the principal to get the compound interest. A `main` function is also defined to print the title of the calculator.

```
def compound_interest(principal, rate, times_compounded, years):  
    """  
    Calculate compound interest.  
  
    Args:  
        principal (float): The initial amount of money.  
        rate (float): The annual interest rate (in percentage, e.g., 5 for 5%).  
        times_compounded (int): Number of times interest is compounded per year.  
        years (float): Number of years the money is invested for.  
  
    Returns:  
        float: The amount after compounding.  
        float: The compound interest earned.  
    """  
    rate_decimal = rate / 100  
    amount = principal * (1 + rate_decimal / times_compounded) ** (times_compounded * years)  
    interest = amount - principal  
    return amount, interest  
  
def main():  
    print("Compound Interest Calculator")
```

NAME: B.VYSHNAVI
ROLL NO.: 2403A510C8
BATCH : 04

```
def calculate_compound_interest(principa.py)  compound_ineterstprompt2.py  compound_ineterstprompt3.py
C lab prblms > AI ASSISTED CODING > compound_ineterstprompt3.py > ...
20 def main():
21     print("Compound Interest Calculator")
22     try:
23         principal = float(input("Enter the principal amount: "))
24         rate = float(input("Enter the annual interest rate (in %): "))
25         times_compounded = int(input("Enter the number of times interest is compounded per year: "))
26         years = float(input("Enter the number of years: "))
27     except ValueError:
28         print("Invalid input. Please enter numeric values.")
29         return
30
31     amount, interest = compound_interest(principal, rate, times_compounded, years)
32     print(f"\nAfter {years} years, the amount will be: {amount:.2f}")
33     print(f"Compound interest earned: {interest:.2f}")
34
35 if __name__ == "__main__":
36     main()
```

Task Description#2

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

Expected Output#2

- AI-generated function evolves from unclear to accurate multi-statistical operation.

PROMPT 1:

```
import-statistics1.py X
C lab prblms > AI ASSISTED CODING > import-statistics1.py > ...
1 import statistics
2
3 def calculate_stats(numbers):
4     average = sum(numbers) / len(numbers)
5     median = statistics.median(numbers)
6     mode = statistics.mode(numbers)
7     return average, median, mode
8
9 # Example usage:
10 nums = [1, 2, 2, 3, 4]
11 avg, med, mod = calculate_stats(nums)
12 print(f"Average: {avg}")
13 print(f"Median: {med}")
14 print(f"Mode: {mod}")

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

Enter the number of times interest is compounded per year: 2
Enter the number of years: 4

After 4.0 years, the amount will be: 10828.57
Compound interest earned: 828.57
PS C:\Users\VYSHNAVI\Desktop> & C:/Users/VYSHNAVI/AppData/Local/Programs/Python/Python312/python.exe C:\Users\VYSHNAVI\Desktop/C lab prblms/AI ASSISTED CODING/import-statistics1.py
Average: 2.4
Median: 2
Mode: 2
PS C:\Users\VYSHNAVI\Desktop>
```

PROMPT 2:

Write a function to calculate average, median, and mode of a list of numbers.

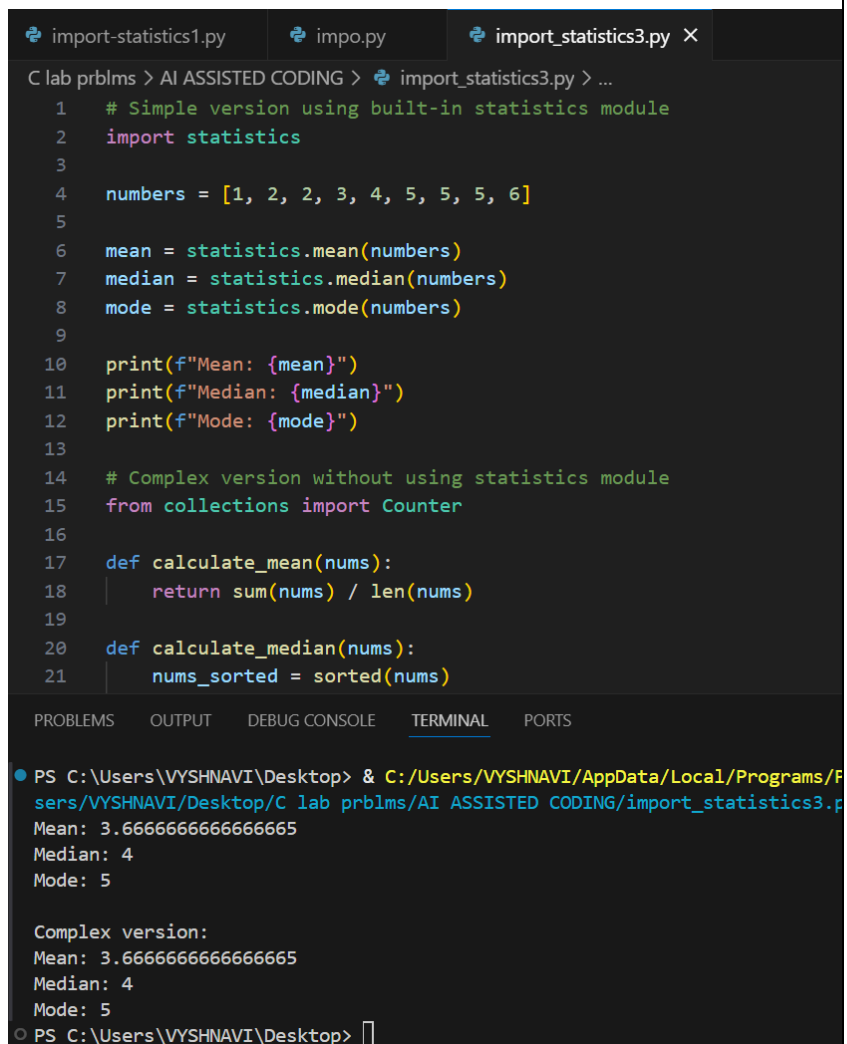
NAME: B.VYSHNAVI
ROLL NO.: 2403A510C8
BATCH : 04

```
import-statistics1.py  impo.py X
C lab prblms > AI ASSISTED CODING > impo.py > calculate_stats
1  def calculate_stats(numbers):
2      from collections import Counter
3
4      if not numbers:
5          return {'average': None, 'median': None, 'mode': None}
6
7      n = len(numbers)
8      sorted_nums = sorted(numbers)
9
10     # Average
11     average = sum(numbers) / n
12
13     # Median
14     if n % 2 == 1:
15         median = sorted_nums[n // 2]
16     else:
17         median = (sorted_nums[n // 2 - 1] + sorted_nums[n // 2]) / 2
18
19     # Mode
20     counts = Counter(sorted_nums)
21     max_count = max(counts.values())
22
23     PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
24
25     import-statistics1.py  impo.py X
26     C lab prblms > AI ASSISTED CODING > impo.py > calculate_stats
27     1  def calculate_stats(numbers):
28         median = sorted_nums[n // 2]
29     else:
30         median = (sorted_nums[n // 2 - 1] + sorted_nums[n // 2]) / 2
31
32     # Mode
33     counts = Counter(sorted_nums)
34     max_count = max(counts.values())
35     mode = [num for num, count in counts.items() if count == max_count]
36     if len(mode) == n:
37         mode = None # No mode if all values are unique
38     elif len(mode) == 1:
39         mode = mode[0]
40
41     return {'average': average, 'median': median, 'mode': mode}
```

PROMPT 3:

write a complex and simple python code to calculate the mean,median,mode for the list of numbers

NAME: B.VYSHNAVI
ROLL NO.: 2403A510C8
BATCH : 04



```
import-statistics1.py  impo.py  import_statistics3.py X
C lab prblms > AI ASSISTED CODING > import_statistics3.py > ...
1  # Simple version using built-in statistics module
2  import statistics
3
4  numbers = [1, 2, 2, 3, 4, 5, 5, 5, 6]
5
6  mean = statistics.mean(numbers)
7  median = statistics.median(numbers)
8  mode = statistics.mode(numbers)
9
10 print(f"Mean: {mean}")
11 print(f"Median: {median}")
12 print(f"Mode: {mode}")
13
14 # Complex version without using statistics module
15 from collections import Counter
16
17 def calculate_mean(nums):
18     return sum(nums) / len(nums)
19
20 def calculate_median(nums):
21     nums_sorted = sorted(nums)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
● PS C:\Users\VYSHNAVI\Desktop> & C:/Users/VYSHNAVI/AppData/Local/Programs/Python/Python310/python.exe C:/Users/VYSHNAVI/Desktop/C lab prblms/AI ASSISTED CODING/import_statistics3.py
Mean: 3.6666666666666665
Median: 4
Mode: 5

Complex version:
Mean: 3.6666666666666665
Median: 4
Mode: 5
○ PS C:\Users\VYSHNAVI\Desktop> 
```

Task Description#3

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

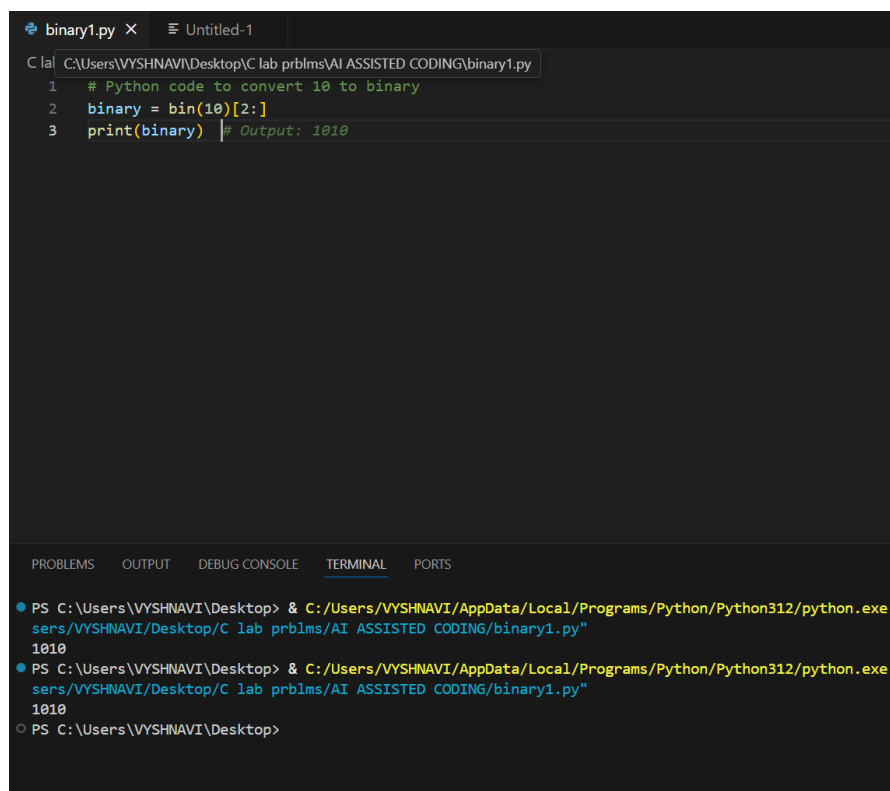
Expected Output#3

- Enhanced AI output with clearer prompts

PROMPT 1:

"Can you convert 10 to binary?"

NAME: B.VYSHNAVI
ROLL NO.: 2403A510C8
BATCH : 04



The screenshot shows a code editor with a file named `binary1.py` and an untitled window. The code in `binary1.py` is as follows:

```
1 # Python code to convert 10 to binary
2 binary = bin(10)[2:]
3 print(binary) # Output: 1010
```

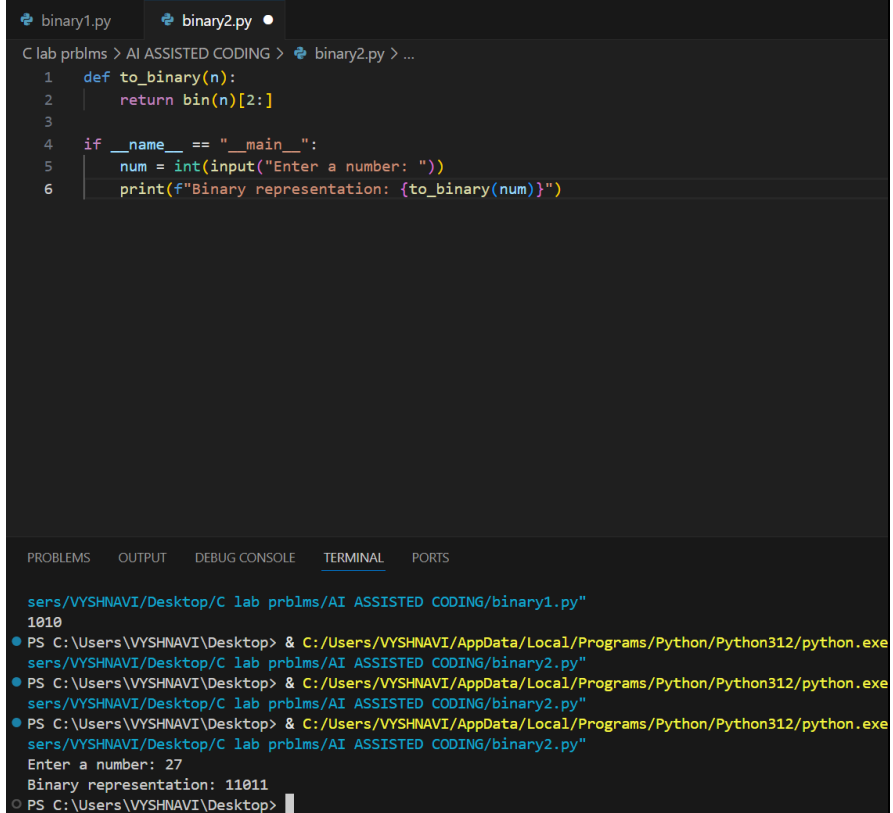
Below the code editor is a terminal window with the following output:

```
PS C:\Users\VYSHNAVI\Desktop> & C:/Users/VYSHNAVI/AppData/Local/Programs/Python/Python312/python.exe
sers/VYSHNAVI/Desktop/C lab prblms/AI ASSISTED CODING/binary1.py"
1010
PS C:\Users\VYSHNAVI\Desktop> & C:/Users/VYSHNAVI/AppData/Local/Programs/Python/Python312/python.exe
sers/VYSHNAVI/Desktop/C lab prblms/AI ASSISTED CODING/binary1.py"
1010
PS C:\Users\VYSHNAVI\Desktop>
```

PROMPT 2:

write a python function code to convert numbers to binary

NAME: B.VYSHNAVI
ROLL NO.: 2403A510C8
BATCH : 04



```
binary1.py binary2.py
C lab prblms > AI ASSISTED CODING > binary2.py > ...
1 def to_binary(n):
2     return bin(n)[2:]
3
4 if __name__ == "__main__":
5     num = int(input("Enter a number: "))
6     print(f"Binary representation: {to_binary(num)}")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

sers\VYSHNAVI\Desktop\C lab prblms\AI ASSISTED CODING\binary1.py"
1010
● PS C:\Users\VYSHNAVI\Desktop> & C:/Users/VYSHNAVI/AppData/Local/Programs/Python/Python312/python.exe
sers\VYSHNAVI\Desktop\C lab prblms\AI ASSISTED CODING\binary2.py"
● PS C:\Users\VYSHNAVI\Desktop> & C:/Users/VYSHNAVI/AppData/Local/Programs/Python/Python312/python.exe
sers\VYSHNAVI\Desktop\C lab prblms\AI ASSISTED CODING\binary2.py"
● PS C:\Users\VYSHNAVI\Desktop> & C:/Users/VYSHNAVI/AppData/Local/Programs/Python/Python312/python.exe
sers\VYSHNAVI\Desktop\C lab prblms\AI ASSISTED CODING\binary2.py"
Enter a number: 27
Binary representation: 11011
○ PS C:\Users\VYSHNAVI\Desktop>
```

PROMPT 3:

"I want to train an AI model using few-shot prompting to learn how to convert integers to binary strings

NAME: B.VYSHNAVI
ROLL NO.: 2403A510C8
BATCH : 04

```
binary1.py binary2.py binary3.py X
C lab prblms > AI ASSISTED CODING > binary3.py > ...
1 # Few-shot prompting example for integer to binary string conversion
2
3 # Few-shot examples
4 examples = [
5     (5, '101'),
6     (8, '1000'),
7     (15, '1111'),
8     (2, '10'),
9     (0, '0')
10 ]
11
12 def int_to_binary(n):
13     """Convert integer to binary string without '0b' prefix."""
14     return bin(n)[2:]
15
16 # Test the function with few-shot examples
17 for num, expected in examples:
18     result = int_to_binary(num)
19     print(f"{num} -> {result} (expected: {expected})")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

sers\VYSHNAVI\Desktop\C lab prblms\AI ASSISTED CODING\binary2.py"
Enter a number: 27
Binary representation: 11011
● PS C:\Users\VYSHNAVI\Desktop> & C:/Users/VYSHNAVI/AppData/Local/Programs/Python/Python312/python.exe sers\VYSHNAVI\Desktop\C lab prblms\AI ASSISTED CODING\binary3.py"
5 -> 101 (expected: 101)
8 -> 1000 (expected: 1000)
15 -> 1111 (expected: 1111)
2 -> 10 (expected: 10)
0 -> 0 (expected: 0)
○ PS C:\Users\VYSHNAVI\Desktop>
```

Task Description#4

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

Expected Output#4

- Consistent functions with shared logic

PROMPT 1:

"Can you make a simple hotel bill generator?"

NAME: B.VYSHNAVI
ROLL NO.: 2403A510C8
BATCH : 04

```
hotel1.py X
C lab prblms > AI ASSISTED CODING > hotel1.py > ...
3 def generate_bill(guest_name, room_rate, nights, extras):
9     print(f"Guest Name: {guest_name}")
10    print(f"Room Rate: ${room_rate:.2f} x {nights} nights")
11    print(f"Extras: ${extras:.2f}")
12    print(f"Subtotal: ${subtotal:.2f}")
13    print(f"Tax (10%): ${tax:.2f}")
14    print(f"Total: ${total:.2f}")
15    print("=====")
16
17 # Example usage:
18 if __name__ == "__main__":
19     name = input("Enter guest name: ")
20     rate = float(input("Enter room rate per night: "))
21     nights = int(input("Enter number of nights: "))
22     extras = float(input("Enter extras (e.g., minibar, laundry): "))
23     generate_bill(name, rate, nights, extras)

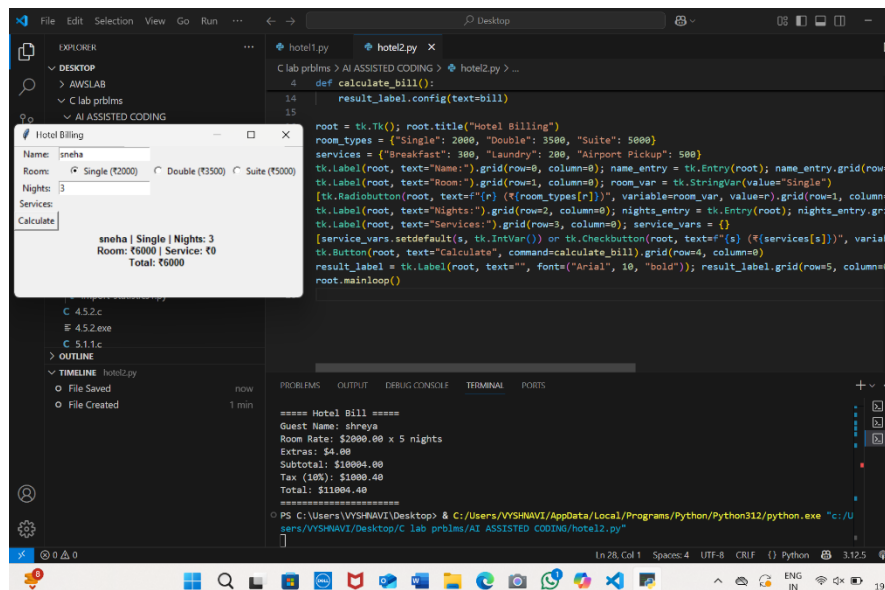
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Enter guest name: shreya
Enter room rate per night: 2000
Enter number of nights: 5
Enter extras (e.g., minibar, laundry): 4
===== Hotel Bill =====
Guest Name: shreya
Room Rate: $2000.00 x 5 nights
Extras: $4.00
Subtotal: $10004.00
Tax (10%): $1000.40
Total: $11004.40

Ln 23, Col 46 Spaces: 4 UTF-8 CRLF {} Python
```

PROMPT 2:

"I need a user interface for a hotel billing system that calculates charges based on customer preferences like room type and services used. Can you help?"



```
hotel2.py X
C lab prblms > AI ASSISTED CODING > hotel2.py > ...
4 def calculate_bill():
14     result_label.config(text=bill)
15
root = tk.Tk(); root.title("Hotel Billing")
room_types = {"Single": 2000, "Double": 3500, "Suite": 5000}
services = {"Breakfast": 200, "Laundry": 200, "Airport Pickup": 500}
tk.Label(root, text="Name:").grid(row=0, column=0); name_entry = tk.Entry(root); name_entry.grid(row=1, column=0)
tk.Label(root, text="Room:").grid(row=1, column=0); room_var = tk.StringVar(value="Single")
tk.Radiobutton(root, text=f"{r} ({room_types[r]})", variable=room_var, value=r).grid(row=1, column=1)
tk.Label(root, text="Nights:").grid(row=2, column=0); nights_entry = tk.Entry(root); nights_entry.grid(row=2, column=1)
tk.Label(root, text="Services:").grid(row=3, column=0); service_vars = {}
[service_vars.setdefault(s, tk.IntVar()) or tk.Checkbutton(root, text=f"{s} ({services[s]})", variable=service_vars[s]).grid(row=3, column=1) for s in services]
tk.Button(root, text="Calculate", command=calculate_bill).grid(row=4, column=0)
result_label = tk.Label(root, text="", font=("Arial", 10, "bold")); result_label.grid(row=5, column=0)
root.mainloop()

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

===== Hotel Bill =====
Guest Name: sneha
Room Rate: $2000.00 x 5 nights
Extras: $4.00
Subtotal: $10004.00
Tax (10%): $1000.40
Total: $11004.40
=====

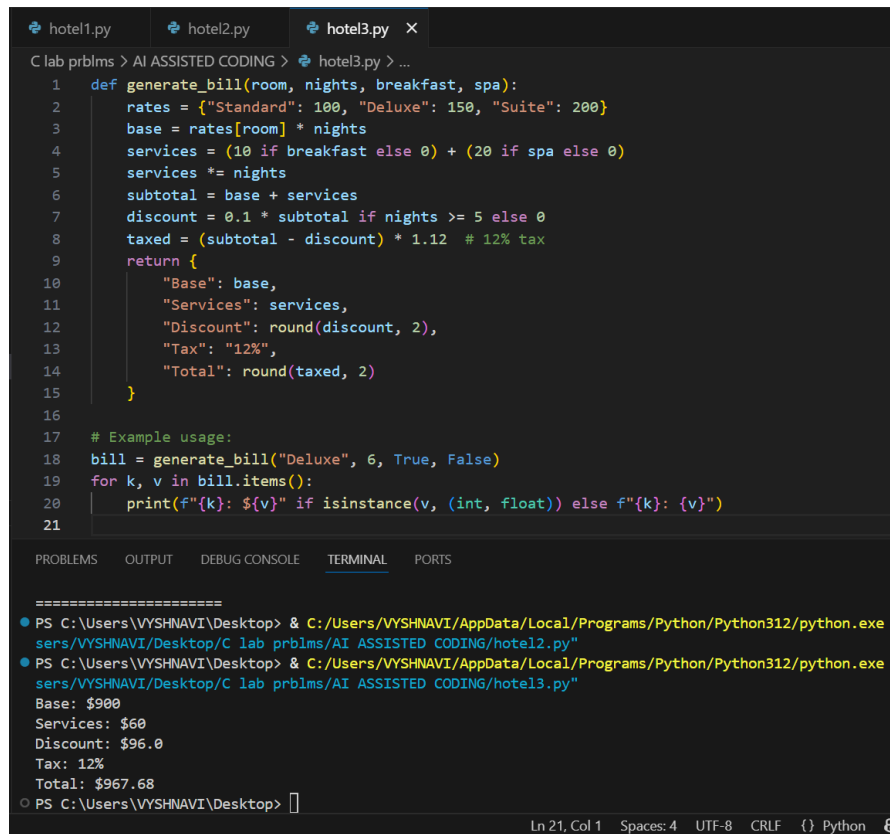
PS C:\Users\VYSHNAVI\Desktop> & C:\Users\VYSHNAVI\AppData\Local\Programs\Python\Python312\python.exe "C:\Users\VYSHNAVI\Desktop\C lab prblms\AI ASSISTED CODING\hotel2.py"
```

PROMPT 3:

"Design a responsive front-end interface for a hotel management system that generates itemized bills based on customer inputs such as room category, duration of stay, and optional services. Include modular components for discounts, taxes, and final billing

NAME: B.VYSHNAVI
ROLL NO.: 2403A510C8
BATCH : 04

summary."



```
hotel1.py hotel2.py hotel3.py X
C lab prblms > AI ASSISTED CODING > hotel3.py > ...
1 def generate_bill(room, nights, breakfast, spa):
2     rates = {"Standard": 100, "Deluxe": 150, "Suite": 200}
3     base = rates[room] * nights
4     services = (10 if breakfast else 0) + (20 if spa else 0)
5     services *= nights
6     subtotal = base + services
7     discount = 0.1 * subtotal if nights >= 5 else 0
8     taxed = (subtotal - discount) * 1.12 # 12% tax
9     return {
10         "Base": base,
11         "Services": services,
12         "Discount": round(discount, 2),
13         "Tax": "12%",
14         "Total": round(taxed, 2)
15     }
16
17 # Example usage:
18 bill = generate_bill("Deluxe", 6, True, False)
19 for k, v in bill.items():
20     print(f"{k}: ${v}" if isinstance(v, (int, float)) else f"{k}: {v}")
21
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
=====
● PS C:\Users\VYSHNAVI\Desktop> & C:/Users/VYSHNAVI/AppData/Local/Programs/Python/Python312/python.exe
sers/VYSHNAVI/Desktop/C lab prblms/AI ASSISTED CODING/hotel2.py"
● PS C:\Users\VYSHNAVI\Desktop> & C:/Users/VYSHNAVI/AppData/Local/Programs/Python/Python312/python.exe
sers/VYSHNAVI/Desktop/C lab prblms/AI ASSISTED CODING/hotel3.py"
Base: $900
Services: $60
Discount: $96.0
Tax: 12%
Total: $967.68
○ PS C:\Users\VYSHNAVI\Desktop>
Ln 21, Col 1 Spaces: 4 UTF-8 CRLF {} Python 6
```

Task Description#5

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

Expected Output#5

- Code quality difference analysis for various prompts

PROMPT 1:

write a code temperature conversion function with clear instruction by using python.

NAME: B.VYSHNAVI
ROLL NO.: 2403A510C8
BATCH : 04

```
def temperature_conversion():  
    """  
    Temperature Conversion Tool  
  
    Instructions:  
    1. Choose the conversion type:  
       a. Celsius to Fahrenheit  
       b. Fahrenheit to Celsius  
    2. Enter the temperature value.  
    3. The converted temperature will be displayed.  
  
    Example:  
       Enter 1 for Celsius to Fahrenheit, 2 for Fahrenheit to Celsius: 1  
       Enter temperature: 25  
       25°C is 77.0°F  
    """  
    print("Temperature Conversion Tool")  
    print("1: Celsius to Fahrenheit")  
    print("2: Fahrenheit to Celsius")  
    choice = input("Enter 1 or 2: ")  
  
    if choice == '1':  
        celsius = float(input("Enter temperature in Celsius: "))  
        fahrenheit = (celsius * 9/5) + 32  
        print(f"{celsius}°C is {fahrenheit}°F")  
    elif choice == '2':  
        fahrenheit = float(input("Enter temperature in Fahrenheit: "))  
        celsius = (fahrenheit - 32) * 5/9  
        print(f"{fahrenheit}°F is {celsius}°C")  
    else:  
        print("Invalid choice. Please enter 1 or 2.")  
  
if __name__ == "__main__":  
    temperature_conversion()
```

PROMPT 2:
give a python code to convert the improving temperature conversion funtion with clear instruction.

```
def convert_temperature(value, from_unit, to_unit):  
    """  
    Convert temperature between Celsius, Fahrenheit, and Kelvin.  
  
    Parameters:  
        value (float): The temperature value to convert.  
        from_unit (str): The unit of the input value ('C', 'F', or 'K').  
        to_unit (str): The unit to convert to ('C', 'F', or 'K').  
  
    Returns:  
        float: The converted temperature value.  
  
    Raises:  
        ValueError: If an invalid unit is provided.  
    """  
    from_unit = from_unit.upper()  
    to_unit = to_unit.upper()  
    # Convert input to Celsius  
    if from_unit == 'C':  
        celsius = value  
    elif from_unit == 'F':  
        celsius = (value - 32) * 5/9  
    elif from_unit == 'K':  
        celsius = value - 273.15  
    else:  
        raise ValueError("Invalid from_unit. Use 'C', 'F', or 'K'.")  
    # Convert Celsius to target unit  
    if to_unit == 'C':  
        return celsius  
    elif to_unit == 'F':  
        return celsius * 9/5 + 32  
    elif to_unit == 'K':  
        return celsius + 273.15  
    else:  
        raise ValueError("Invalid to_unit. Use 'C', 'F', or 'K'.")  
  
# To convert 100 Celsius to Fahrenheit:  
result = convert_temperature(100, 'C', 'F')  
print("100°C is", result, "°F")  
# To convert 32 Fahrenheit to Kelvin:  
result = convert_temperature(32, 'F', 'K')  
print("32°F is", result, "°K")
```

PROMPT 3: to improve the temperature write a python code to create a temperatue conventor

NAME: B.VYSHNAVI
ROLL NO.: 2403A510C8
BATCH : 04

```
def celsius_to_fahrenheit(celsius):  
    return (celsius * 9/5) + 32  
  
def fahrenheit_to_celsius(fahrenheit):  
    return (fahrenheit - 32) * 5/9  
  
def main():  
    print("Temperature Converter")  
    print("1. Celsius to Fahrenheit")  
    print("2. Fahrenheit to Celsius")  
    choice = input("Choose conversion (1 or 2): ")  
  
    if choice == '1':  
        c = float(input("Enter temperature in Celsius: "))  
        f = celsius_to_fahrenheit(c)  
        print(f"{c}°C = {f:.2f}°F")  
    elif choice == '2':  
        f = float(input("Enter temperature in Fahrenheit: "))  
        c = fahrenheit_to_celsius(f)  
        print(f"{f}°F = {c:.2f}°C")  
    else:  
        print("Invalid choice.")  
  
if __name__ == "__main__":  
    main()
```

Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots

Evaluation Criteria:

Criteria	Max Marks
Task#1	0.5
Task#2	0.5
Task #3	0.5
Task #4	0.5
Task #5	0.5
Total	2.5 Marks

NAME: B.VYSHNAVI
ROLL NO.: 2403A510C8
BATCH : 04