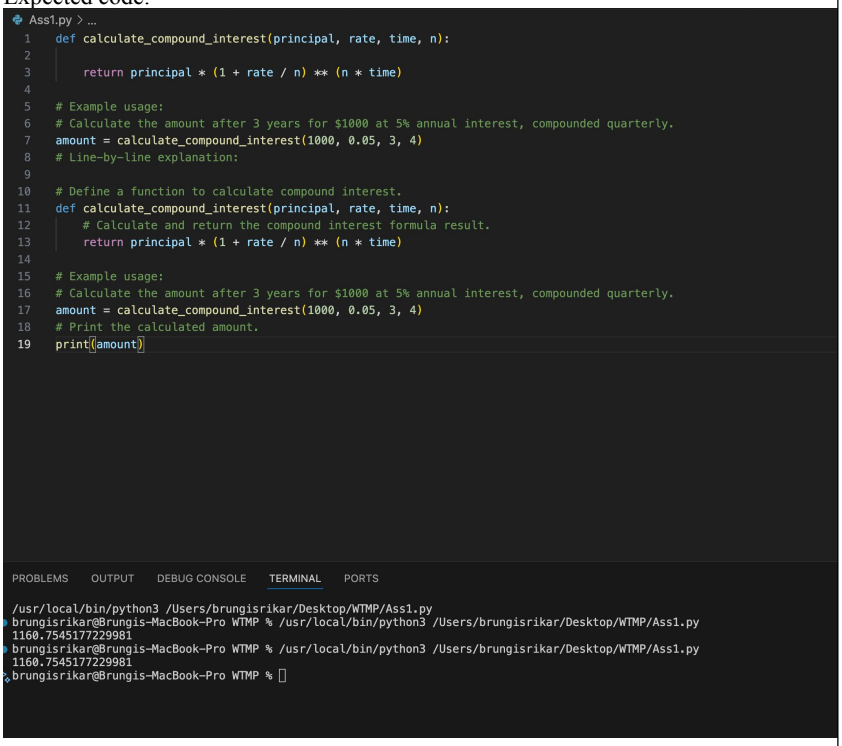
SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
ProgramName: B. Tech		**Assignment Type: Lab**	**AcademicYear:** 2025-2026
CourseCoordinatorName		Venkataramana Veeramsetty	
Instructor(s)Name		1. Dr. Mohammed Ali Shaik 2. Dr. T Sampath Kumar 3. Mr. S Naresh Kumar 4. Dr. V. Rajesh 5. Dr. Brij Kishore 6. Dr Pramoda Patro 7. Dr. Venkataramana 8. Dr. Ravi Chander 9. 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	*ExpectedTime to complete*
1	Lab 3: Prompt Engineering – Improving Prompts and Context Management **Lab Objectives:** ● 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. **Lab Outcomes (LOs):** After completing this lab, students will be able to: ● 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. ● Evaluate and refactor code using Cursor AI features. ● Compare AI tool behavior and code quality across different platforms.	03.08.2025 EOD

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 code:



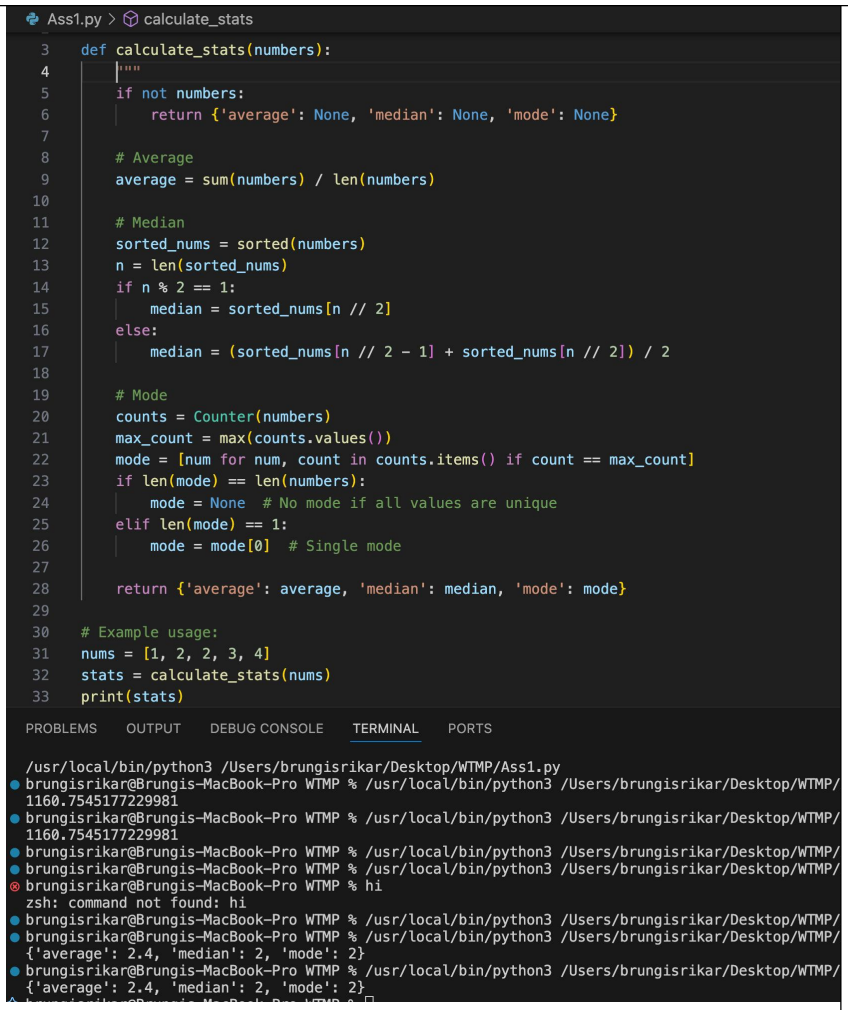
-

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.

- 

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

Ass1.py > ...

```python
# Function to convert a number to binary
def convert_to_binary(num):
    return bin(num)[2:]

# Example inputs and outputs for convert_to_binary(num)
examples = [
    (5, "101"),
    (10, "1010"),
    (0, "0"),
    (255, "11111111"),
    (1, "1"),
    (32, "100000"),
]

# Displaying the examples
for num, expected_output in examples:
    result = convert_to_binary(num)
    print(f"Input: {num} | Expected Output: {expected_output} | Actual Output: {result}")

# Explanation:
# Few-shot prompting means providing several input-output examples to guide the AI in understanding the task.
# Here, we show how different numbers are converted to their binary representations.
# This helps the AI generalize the pattern: for any integer input, output its binary string (without the '0b' pre
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
brungisrikar@Brungis-MacBook-Pro WTMP % hi
zsh: command not found: hi
brungisrikar@Brungis-MacBook-Pro WTMP % /usr/local/bin/python3 /Users/brungisrikar/Desktop/WTMP/Ass1.py
brungisrikar@Brungis-MacBook-Pro WTMP % /usr/local/bin/python3 /Users/brungisrikar/Desktop/WTMP/Ass1.py
{'average': 2.4, 'median': 2, 'mode': 2}
brungisrikar@Brungis-MacBook-Pro WTMP % /usr/local/bin/python3 /Users/brungisrikar/Desktop/WTMP/Ass1.py
{'average': 2.4, 'median': 2, 'mode': 2}
brungisrikar@Brungis-MacBook-Pro WTMP % /usr/local/bin/python3 /Users/brungisrikar/Desktop/WTMP/Ass1.py
Input: 5 | Expected Output: 101 | Actual Output: 101
Input: 10 | Expected Output: 1010 | Actual Output: 1010
Input: 0 | Expected Output: 0 | Actual Output: 0
Input: 255 | Expected Output: 11111111 | Actual Output: 11111111
Input: 1 | Expected Output: 1 | Actual Output: 1
Input: 32 | Expected Output: 100000 | Actual Output: 100000
```

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

Ass1.py > ...

```python
import tkinter as tk
from tkinter import messagebox

# Sample menu and prices
MENU = {
    "Room": 2000,
    "Breakfast": 250,
    "Lunch": 400,
    "Dinner": 500,
    "Laundry": 150,
    "Spa": 800
}

def calculate_bill(selections):
    total = 0
    details = []
    for item, var in selections.items():
        qty = var.get()
        if qty > 0:
            cost = MENU[item] * qty
            details.append(f"{item} x {qty} = ₹{cost}")
            total += cost
    return details, total

def generate_bill():
    details, total = calculate_bill(selections)
    if not details:
        messagebox.showinfo("Bill", "No services selected.")
        return
    bill_text = "\n".join(details) + f"\n\nTotal Bill: ₹{total}"
    messagebox.showinfo("Bill", bill_text)
```

```
Ass1.py > ...
32
33  # UI setup
34  root = tk.Tk()
35  root.title("Hotel Bill Generator")
36
37  tk.Label(root, text="Select Services and Quantity", font=("Arial", 14)).pack(pady=10)
38
39  frame = tk.Frame(root)
40  frame.pack()
41
42  selections = {}
43  for idx, (item, price) in enumerate(MENU.items()):
44      tk.Label(frame, text=f"{item} (₹{price})", width=15, anchor='w').grid(row=idx, column=0, padx=5, pady=5)
45      var = tk.IntVar()
46      tk.Spinbox(frame, from_=0, to=10, textvariable=var, width=5).grid(row=idx, column=1, padx=5)
47      selections[item] = var
48
49  tk.Button(root, text="Generate Bill", command=generate_bill, font=("Arial", 12)).pack(pady=20)
50
51  root.mainloop()
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

brungisrikar@Brungis-MacBook-Pro WTMP % /usr/local/bin/python3 /Users/brungisrikar/Desktop/WTMP/Ass1.py
brungisrikar@Brungis-MacBook-Pro WTMP % /usr/local/bin/python3 /Users/brungisrikar/Desktop/WTMP/Ass1.py
{'average': 2.4, 'median': 2, 'mode': 2}
brungisrikar@Brungis-MacBook-Pro WTMP % /usr/local/bin/python3 /Users/brungisrikar/Desktop/WTMP/Ass1.py
{'average': 2.4, 'median': 2, 'mode': 2}
brungisrikar@Brungis-MacBook-Pro WTMP % /usr/local/bin/python3 /Users/brungisrikar/Desktop/WTMP/Ass1.py
Input: 5 | Expected Output: 101 | Actual Output: 101
Input: 10 | Expected Output: 1010 | Actual Output: 1010
Input: 0 | Expected Output: 0 | Actual Output: 0
Input: 255 | Expected Output: 11111111 | Actual Output: 11111111
Input: 1 | Expected Output: 1 | Actual Output: 1
Input: 32 | Expected Output: 100000 | Actual Output: 100000
brungisrikar@Brungis-MacBook-Pro WTMP % /usr/local/bin/python3 /Users/brungisrikar/Desktop/WTMP/Ass1.py
/usr/local/bin/python3 /Users/brungisrikar/Desktop/WTMP/Ass1.py
```

Task Description#5

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

Expected Output#5

- Code quality difference analysis for various prompts

```
Ass1.py > ...
1   def convert_temperature(value, from_unit, to_unit):
5       elif from_unit == 'F':
6           celsius = (value - 32) * 5 / 9
7       elif from_unit == 'K':
8           celsius = value - 273.15
9       else:
10          raise ValueError("Invalid from_unit. Use 'C', 'F', or 'K'."
11
12      # Convert Celsius to target unit
13      if to_unit == 'C':
14          return celsius
15      elif to_unit == 'F':
16          return celsius * 9 / 5 + 32
17      elif to_unit == 'K':
18          return celsius + 273.15
19      else:
20          raise ValueError("Invalid to_unit. Use 'C', 'F', or 'K'.")
21
22  # Example usage
23  result = convert_temperature(100, 'C', 'F')
24  print(result)
```

- **Explanation:**

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**