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	*Expected Time 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

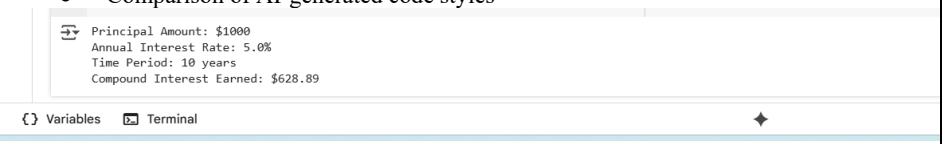
Task Description#1

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

- **Prompt :** Write a funtion to calculate compound interest, starting with only the function name

Expected Output#1

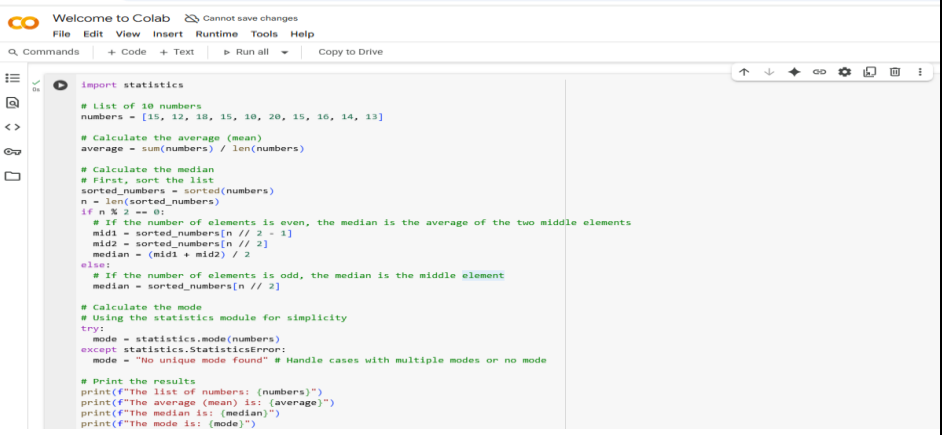
- Comparison of AI-generated code styles



Task Description#2

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

- **Prompt :** Write a function to calculate average, median, and mode of a list of numbers.



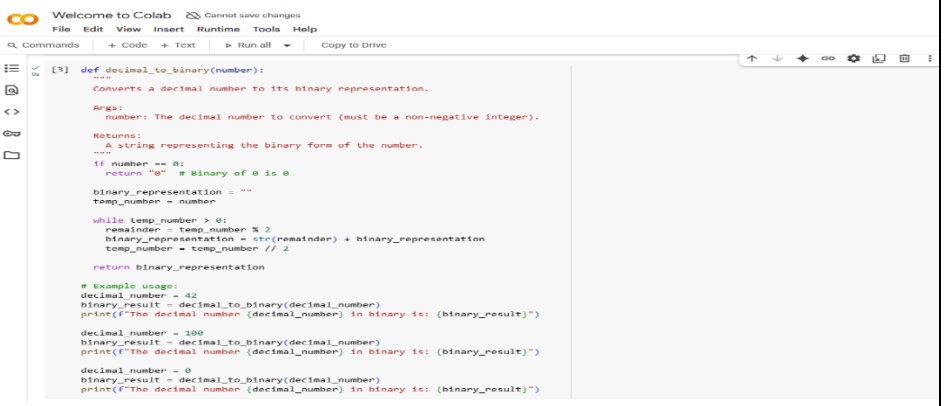
Expected Output#2

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

Task Description#3

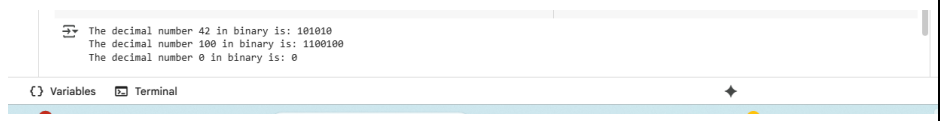
- Provide multiple examples of input-output to the AI for convert_to_binary(num) function. Observe how AI uses few-shot prompting to generalize.

- **Prompt :** Write a progarm in python to convert a number into binary number suing function



Expected Output#3

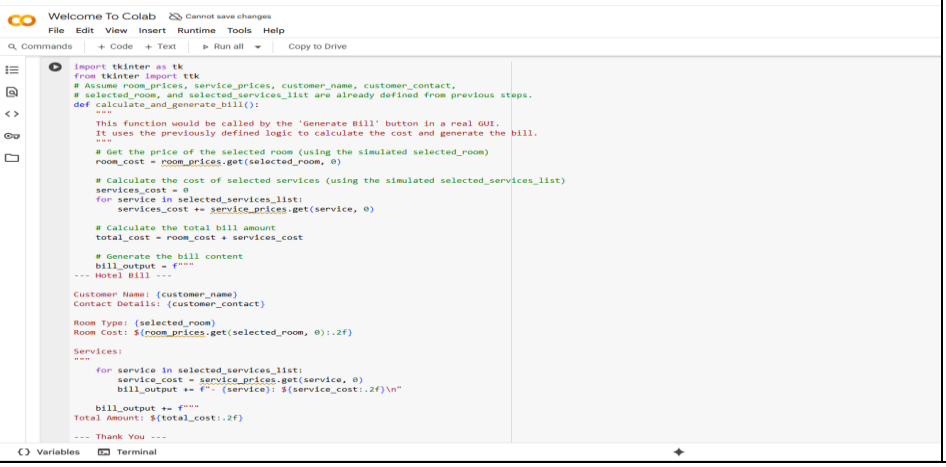
- Enhanced AI output with clearer prompts



Task Description#4

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

- **Prompt :** Generate an user interface for an hotel to generate bill based on customer requirements



```
    """
    # In a real GUI, you would update a text area or label with bill_output
    print("--- Simulated Bill Display in GUI ---")
    print(bill_output)
    print("-------------------------------------")
# --- Simplified Tkinter Structure (Illustrative only) ---
print("Simulating the 'Generate Bill' button click...")
calculate_and_generate_bill()
```

Expected Output#4

- Consistent functions with shared logic

```
--- Simulated Bill Display in GUI ---

--- Hotel Bill ---

Customer Name: John Doe
Contact Details: john.doe@example.com

Room Type: Deluxe
Room Cost: $150.00

Services:
- Room Service: $30.00
- WiFi: $10.00

Total Amount: $190.00

--- Thank You ---

-------------------------------------
```

Task Description#5

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

- **Prompt :** Write a program in python to create a function to convert temperatures between celsius and fahrenheit

Generate the python code for analysing prompt specificity : improving temperatures conversion function with clear instructions

Okay, I understand you'd like to switch to a new task: analyzing prompt specificity to improve a temperature conversion function with clear instructions. This is a good approach to understand how different prompts affect the output of a model.

Here is a plan to tackle this:

- **Define the temperature conversion task**: Clearly state the goal: create a Python function to convert temperatures between Celsius, Fahrenheit, and Kelvin.
- **Create different prompts**: Write several different prompts, varying in specificity and clarity, that ask for this temperature conversion function.
- **Analyze prompt specificity**: Examine each prompt to identify what makes it more or less specific and how that might influence the generated code and instructions.
- **Generate code based on prompts**: (This step would ideally involve using an LLM, but since I am the LLM, I will generate the improved code directly based on the analysis of what a specific prompt would likely yield.) Generate the Python code for the temperature conversion function with clear instructions, reflecting the understanding gained from analyzing prompt specificity.
- **Evaluate the generated code and instructions**: Review the generated code and instructions for correctness, clarity, and completeness.
- **Finish task**: Summarize the findings on prompt specificity and present the

Welcome To Colab Cannot save changes

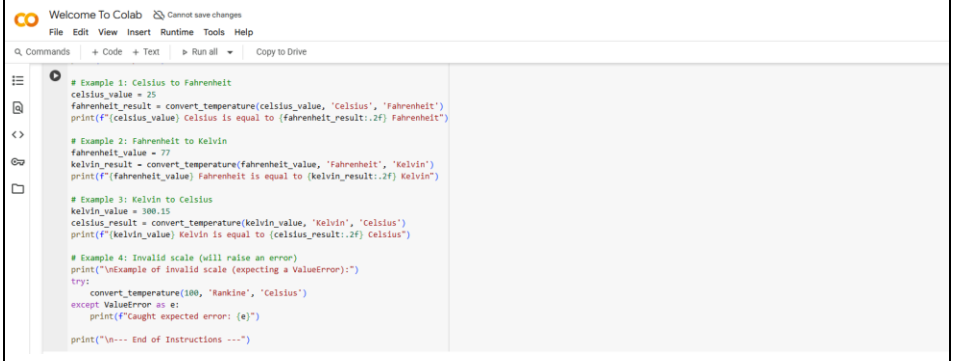
File Edit View Insert Runtime Tools Help

Commands + Code + Text ▶ Run all Copy to Drive

```
def convert_temperature(value, from_scale, to_scale):
    supported_scales = ['Celsius', 'Fahrenheit', 'Kelvin']

    if from_scale not in supported_scales:
        raise ValueError(f"Invalid 'from_scale': {from_scale}. Supported scales are {supported_scales}")
    if to_scale not in supported_scales:
        raise ValueError(f"Invalid 'to_scale': {to_scale}. Supported scales are {supported_scales}")

    # Conversion logic
    if from_scale == 'Celsius':
        if to_scale == 'Fahrenheit':
            return (value * 9/5) + 32
        elif to_scale == 'Kelvin':
            return value + 273.15
        else: # to_scale == 'Celsius'
            return value
    elif from_scale == 'Fahrenheit':
        if to_scale == 'Celsius':
            return (value - 32) * 5/9
        elif to_scale == 'Kelvin':
            return (value - 32) * 5/9 + 273.15
        else: # to_scale == 'Fahrenheit'
            return value
    elif from_scale == 'Kelvin':
        if to_scale == 'Celsius':
            return value - 273.15
        elif to_scale == 'Fahrenheit':
            return (value - 273.15) * 9/5 + 32
        else: # to_scale == 'Kelvin'
            return value
# Instructions on how to use the function:
print("--- How to use the convert_temperature function ---")
print("This function converts a temperature value between Celsius, Fahrenheit, and Kelvin.")
print("It takes three arguments:")
print("1. value: The numerical temperature (e.g., 25, 77, 300.15)")
print("2. from_scale: The original scale as a string ('Celsius', 'Fahrenheit', or 'Kelvin')")
print("3. to_scale: The target scale as a string ('Celsius', 'Fahrenheit', or 'Kelvin')")
print("\nExamples:")
```



Expected Output#5

- Code quality difference analysis for various prompts

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**