

**COURSE: AI Assisted Coding**

**NAME : SANJAY KARUPOTHULA**

**BATCH-34**

**HALLTICKET.NO : 2303A52337**

**AIAC-ASS-3.2**

### **Lab 3:**

**Prompt Engineering – Improving Prompts and Context Management.**

#### **Task Description – 1**

Progressive Prompting for Calculator Design

#### **Objective:**

To observe how AI-generated code improves as prompt clarity and context increase.

#### **AI Output – Level 1**

```
LAB-3.2 > Tasks.py > ...
1  #Create a calculator function
2  def calculator(a, b, operation):
3      if operation == 'add':
4          return a + b
5      elif operation == 'subtract':
6          return a - b
7      elif operation == 'multiply':
8          return a * b
9      elif operation == 'divide':
10         if b != 0:
11             return a / b
12         else:
13             return "Error: Division by zero"
14     else:
15         return "Error: Invalid operation"
16 #Example usage
17 result = calculator(10, 5, 'add')
18 print("Result:", result) # Output: Result: 15
19
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\SANDEEP\OneDrive\Documents\Desktop\AI\_ASS> & C:\Users\SANDEEP\AppData\Local\Programs\Python\Python313\python.exe e c:\Users\SANDEEP\OneDrive\Documents\Desktop\AI\_ASS\LAB-3.2\Tasks.py  
Result: 15  
PS C:\Users\SANDEEP\OneDrive\Documents\Desktop\AI\_ASS>

Ln 19, Col 1 Spaces: 4 UTF-8 CRLF Python 3.13.7 Go Live

## LEVEL-2:

```
LAB-3.2 > Tasks.py > ...
20
21 """LEVEL-2"""
22
23
24 #Create a calculator function that supports addition, subtraction, multiplication, and division.
25 # Add basic error handling.
26 def calculator(a, b, operation):
27     if operation == 'add':
28         return a + b
29     elif operation == 'subtract':
30         return a - b
31     elif operation == 'multiply':
32         return a * b
33     elif operation == 'divide':
34         if b != 0:
35             return a / b
36         else:
37             return "Error: Division by zero"
38     else:
39         return "Error: Invalid operation"

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Result: 15  
PS C:\Users\SANDEEP\OneDrive\Documents\Desktop\AI\_ASS>

Ln 21, Col 11 Spaces: 4 UTF-8 CRLF Python 3.13.7 Go Live

## LEVEL-3:

The screenshot shows the VS Code editor with a file named `Tasks.py` open. The code defines a `calculator` function that takes two numbers `a` and `b`, and a string `operation` as arguments. The function performs basic arithmetic operations: addition, subtraction, multiplication, and division. It includes a docstring that describes the function's purpose, parameters, and returns. Example usage is shown in the form of a series of calls to the `calculator` function, demonstrating its behavior for various inputs and operations, including error handling for division by zero and invalid operations.

```
45
46 #Create a calculator function with docstring and example usage.
47 def calculator(a, b, operation):
48
49     A simple calculator function that performs basic arithmetic operations.
50     Parameters:
51     a (float): The first number.
52     b (float): The second number.
53     operation (str): The operation to perform. Can be 'add', 'subtract', 'multiply', or 'divide'.
54     Returns:
55     float: The result of the arithmetic operation.
56     str: Error message if division by zero or invalid operation is attempted.
57     Example usage:
58     >>> calculator(10, 5, 'add')
59     15
60     >>> calculator(10, 5, 'subtract')
61     5
62     >>> calculator(10, 5, 'multiply')
63     50
64     >>> calculator(10, 5, 'divide')
65     2.0
66     >>> calculator(10, 0, 'divide')
67     'Error: Division by zero'
68     >>> calculator(10, 5, 'modulus')
69     'Error: Invalid operation'
70
71     if operation == 'add':
72         return a + b
73     elif operation == 'subtract':
74         return a - b
75     elif operation == 'multiply':
76         return a * b
77     elif operation == 'divide':
78         if b != 0:
79             return a / b
80         else:
81             return "Error: Division by zero"
82     else:
83         return "Error: Invalid operation"
```

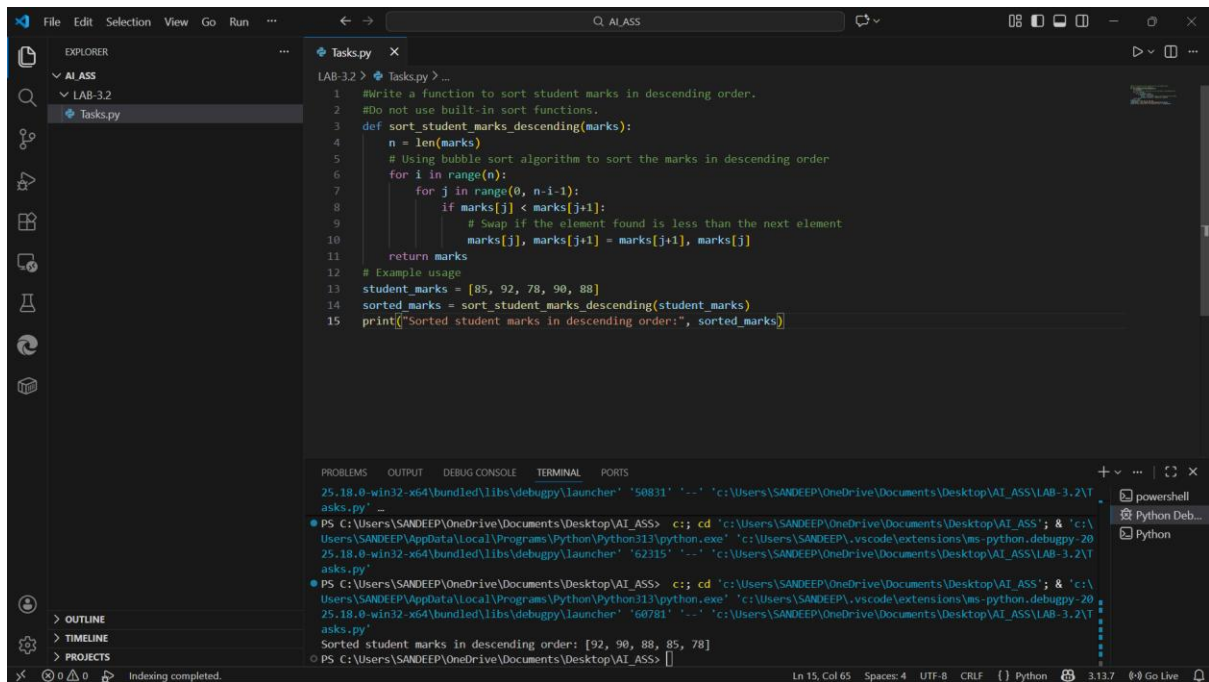
## Task -2:

- Refining Prompts for Sorting Logic: Start with a vague prompt for sorting student marks, then refine it to clearly specify sorting order and constraints.

The screenshot shows the VS Code editor with a file named `Tasks.py` open. The code defines a `sort_student_marks` function that takes a list of marks as input and returns a sorted list. The function uses the `sorted` function to sort the marks in ascending order. Example usage is shown in the form of a call to the `sort_student_marks` function with a list of marks, and the output is printed to the console.

```
1 #Write a function to sort student marks.
2 def sort_student_marks(marks):
3     return sorted(marks)
4
4 # Example usage:
5 student_marks = [88, 92, 79, 85, 95]
6 sorted_marks = sort_student_marks(student_marks)
7 print(f"Sorted student marks:", sorted_marks)
```

The terminal output shows the execution of the code, displaying the sorted marks: `Sorted student marks: [79, 85, 88, 92, 95]`.



```
LAB-3.2 > Tasks.py > ...
1 #Write a function to sort student marks in descending order.
2 #Do not use built-in sort functions.
3 def sort_student_marks_descending(marks):
4     n = len(marks)
5     # Using bubble sort algorithm to sort the marks in descending order
6     for i in range(n):
7         for j in range(0, n-i-1):
8             if marks[j] < marks[j+1]:
9                 # Swap if the element found is less than the next element
10                marks[j], marks[j+1] = marks[j+1], marks[j]
11    return marks
12 # Example usage
13 student_marks = [85, 92, 78, 90, 88]
14 sorted_marks = sort_student_marks_descending(student_marks)
15 print("Sorted student marks in descending order:", sorted_marks)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

25.18.0-win32-x64\bundled\libs\debugpy\launcher '50831' '-' 'c:\Users\SANDEEP\OneDrive\Documents\Desktop\AI\_ASS\LAB-3.2\T  
asks.py' --

● PS C:\Users\SANDEEP\OneDrive\Documents\Desktop\AI\_ASS> c; cd 'c:\Users\SANDEEP\OneDrive\Documents\Desktop\AI\_ASS'; & 'c:\  
Users\SANDEEP\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\SANDEEP\.vscode\extensions\ms-python.debugpy-20  
25.18.0-win32-x64\bundled\libs\debugpy\launcher' '62315' '-' 'c:\Users\SANDEEP\OneDrive\Documents\Desktop\AI\_ASS\LAB-3.2\T  
asks.py'

● PS C:\Users\SANDEEP\OneDrive\Documents\Desktop\AI\_ASS> c; cd 'c:\Users\SANDEEP\OneDrive\Documents\Desktop\AI\_ASS'; & 'c:\  
Users\SANDEEP\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\SANDEEP\.vscode\extensions\ms-python.debugpy-20  
25.18.0-win32-x64\bundled\libs\debugpy\launcher' '60781' '-' 'c:\Users\SANDEEP\OneDrive\Documents\Desktop\AI\_ASS\LAB-3.2\T  
asks.py'

Sorted student marks in descending order: [92, 90, 88, 85, 78]

PS C:\Users\SANDEEP\OneDrive\Documents\Desktop\AI\_ASS> [

Ln 15, Col 65 Spaces: 4 UTF-8 CRLF Python 3.13.7 Go Live

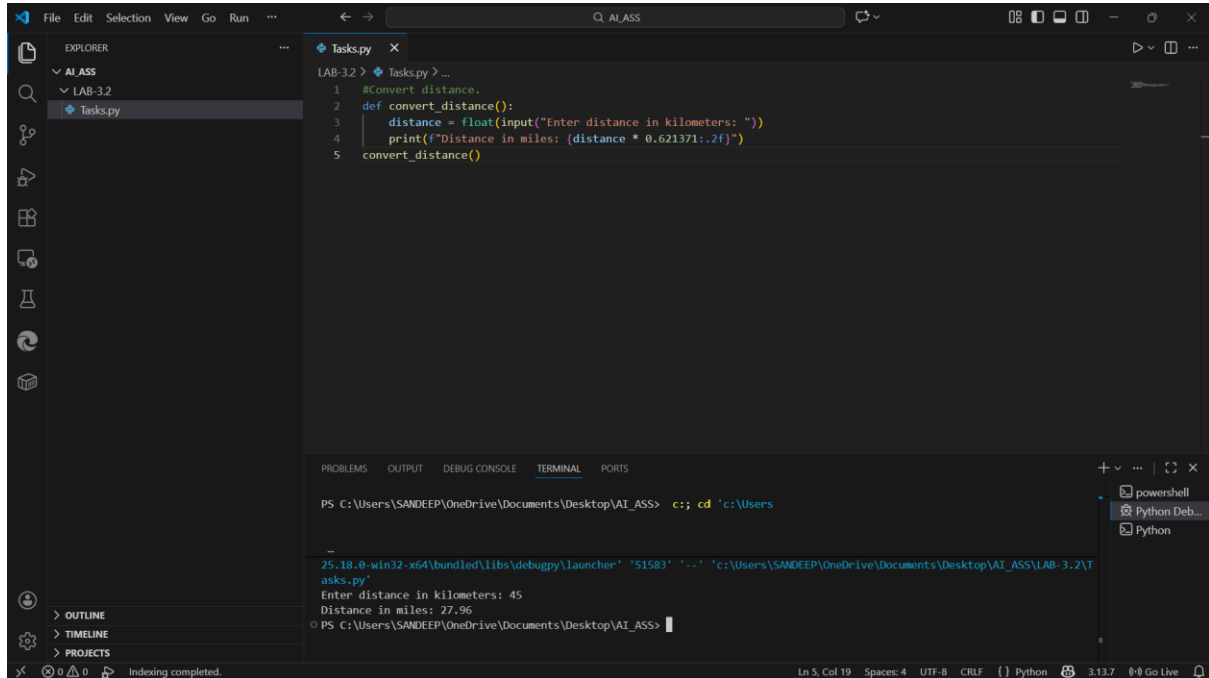
## Task -3

- Few-Shot Prompting for Prime Number Validation: Provide multiple input-output examples for a function that checks whether a number is prime. Observe how few-shot prompting improves correctness.



## Task Description-5

- Analyzing Prompt Specificity in Unit Conversion Functions: Improving a Unit Conversion Function (Kilometers to Miles and Miles to Kilometers) Using Clear Instructions.



The screenshot shows the Visual Studio Code (VS Code) interface. The Explorer pane on the left shows a project named 'AI\_ASS' with a subfolder 'LAB-3.2' containing a file 'Tasks.py'. The main editor window displays the code for 'Tasks.py'.

```
LAB-3.2 > Tasks.py > ...
1 #Convert distance.
2 def convert_distance():
3     distance = float(input("Enter distance in kilometers: "))
4     print(f"Distance in miles: {distance * 0.621371:.2f}")
5 convert_distance()
```

The bottom panel shows the TERMINAL output. The command prompt is at 'PS C:\Users\SANDEEP\OneDrive\Documents\Desktop\AI\_ASS>'. The user has entered 'cd 'c:\Users' and then 'python tasks.py'. The output shows the program running and asking for input: 'Enter distance in kilometers: 45'. The program then outputs: 'Distance in miles: 27.96'.

```
PS C:\Users\SANDEEP\OneDrive\Documents\Desktop\AI_ASS> cd 'c:\Users
...
25.18.0-win32-x64\bundled\libs\debugpy\launcher' '51583' '-' 'c:\Users\SANDEEP\OneDrive\Documents\Desktop\AI_ASS\LAB-3.2\T
asks.py'
Enter distance in kilometers: 45
Distance in miles: 27.96
PS C:\Users\SANDEEP\OneDrive\Documents\Desktop\AI_ASS>
```

- (Kilometers to Miles and Miles to Kilometers)

The image shows a Visual Studio Code (VS Code) editor window with a dark theme. The Explorer sidebar on the left shows a project named 'AI\_ASS' with a subfolder 'LAB-3.2' containing a file 'Tasks.py'. The main editor area displays the contents of 'Tasks.py', which defines two functions for converting kilometers to miles and miles to kilometers, and includes example usage code. The bottom panel shows the 'TERMINAL' tab with two PowerShell commands and their output. The first command runs the script, and the second command runs it with specific arguments. The output shows the results of the conversions: 10 kilometers is equal to 6.21371 miles, and 6.21371 miles is equal to 10.0 kilometers. The status bar at the bottom indicates the current line and column (Ln 18, Col 1), the file encoding (UTF-8), the line ending (CRLF), the active language (Python), and the version (3.13.7).

```
File Edit Selection View Go Run ... Q AI_ASS
EXPLORER
  AI_ASS
  LAB-3.2
  Tasks.py
Tasks.py
LAB-3.2 > Tasks.py > ...
1 #Create two functions:
2 #1. Convert kilometers to miles
3 #2. Convert miles to kilometers
4 #Use correct formulas.
5 def kilometers_to_miles(km):
6     miles = km * 0.621371
7     return miles
8 def miles_to_kilometers(miles):
9     km = miles / 0.621371
10    return km
11 #Example usage:
12 km_value = 10
13 miles_value = kilometers_to_miles(km_value)
14 print(f"{km_value} kilometers is equal to {miles_value} miles.")
15 miles_value = 6.21371
16 km_value = miles_to_kilometers(miles_value)
17 print(f"{miles_value} miles is equal to {km_value} kilometers.")
18

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\SANDEEP\OneDrive\Documents\Desktop\AI_ASS> cd "C:\Users\SANDEEP\OneDrive\Documents\Desktop\AI_ASS"; & "c:\Users\SANDEEP\AppData\Local\Programs\Python\Python313\python.exe" "c:\Users\SANDEEP\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher" "58556" "--" "c:\Users\SANDEEP\OneDrive\Documents\Desktop\AI_ASS\LAB-3.2\Tasks.py"
PS C:\Users\SANDEEP\OneDrive\Documents\Desktop\AI_ASS> cd "c:\Users\SANDEEP\OneDrive\Documents\Desktop\AI_ASS"; & "c:\Users\SANDEEP\AppData\Local\Programs\Python\Python313\python.exe" "c:\Users\SANDEEP\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher" "49686" "--" "c:\Users\SANDEEP\OneDrive\Documents\Desktop\AI_ASS\LAB-3.2\Tasks.py"
10 kilometers is equal to 6.21371 miles.
6.21371 miles is equal to 10.0 kilometers.
PS C:\Users\SANDEEP\OneDrive\Documents\Desktop\AI_ASS>

Ln 18, Col 1 Spaces: 4 UTF-8 CRLF Python 3.13.7 Go Live
```