

ASSIGNMENT

Name:A.Sai Varshith

Roll Number: 2303A51831

Batch - 04

AI Assisted Coding

20-01-2026

Task Description-1

- Progressive Prompting for Calculator Design: Ask the AI to design a simple calculator

program by initially providing only the function name. Gradually enhance the prompt by

adding comments and usage examples.

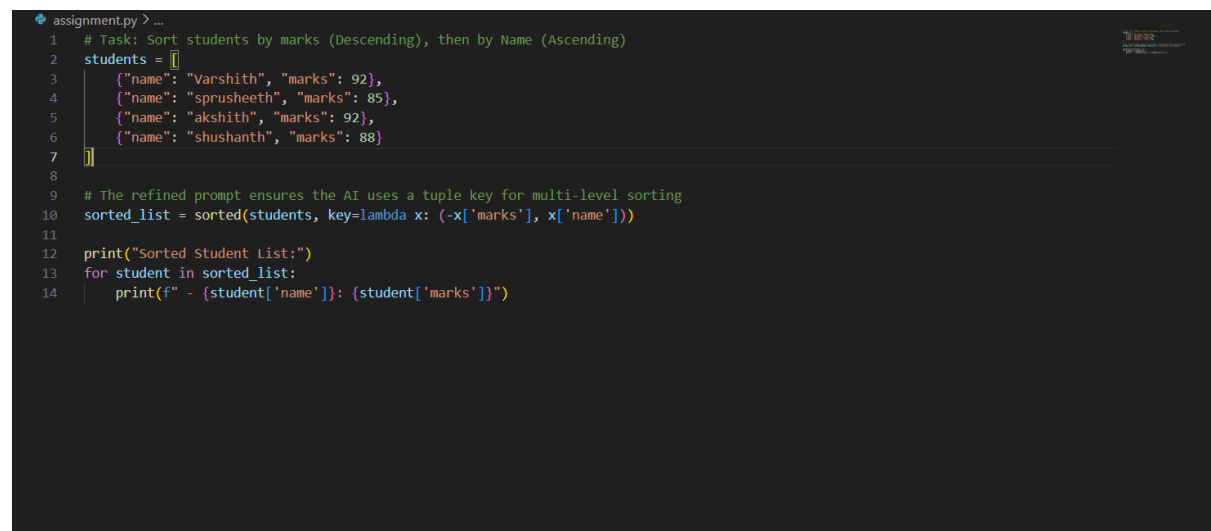
Expected Output-1

- Comparison showing improvement in AI-generated calculator logic and structure.

Task Description-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.

A screenshot of a code editor with a dark background. The code is in Python and is part of a file named 'assignment.py'. It defines a list of student dictionaries and then sorts them by marks in descending order and then by name in ascending order. The code includes comments explaining the task and the sorting logic.

```
1 # Task: Sort students by marks (Descending), then by Name (Ascending)
2 students = [
3     {"name": "Varshith", "marks": 92},
4     {"name": "sprusheeth", "marks": 85},
5     {"name": "akshith", "marks": 92},
6     {"name": "shushanth", "marks": 88}
7 ]
8
9 # The refined prompt ensures the AI uses a tuple key for multi-level sorting
10 sorted_list = sorted(students, key=lambda x: (-x['marks'], x['name']))
11
12 print("Sorted Student List:")
13 for student in sorted_list:
14     print(f" - {student['name']}: {student['marks']}")
```

Expected Output-2

- AI-generated sorting function evolves from ambiguous logic to an accurate and efficient

implementation.

```
top\ai\assignment.py
Sorted Student List:
- Varshith: 92
- akshith: 92
- shushanth: 88
- sprusheeth: 85
PS C:\Users\hp\OneDrive\Desktop\ai> []
```

Task Description-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.

```
assignment.py > ...
1 def is_prime(n):
2     # Prompting with examples (Input: 1 -> Output: False)
3     # ensures these guards are included.
4     if n <= 1:
5         return False
6     if n <= 3:
7         return True
8     if n % 2 == 0 or n % 3 == 0:
9         return False
10
11     i = 5
12     while i * i <= n:
13         if n % i == 0 or n % (i + 2) == 0:
14             return False
15         i += 6
16     return True
17
18 # Testing cases
19 test_values = [-5, 1, 2, 11, 25]
20 for val in test_values:
21     print(f"Is {val} prime? {is_prime(val)}")
```

Expected Output-3

- Improved prime-checking function with better edge-case handling.

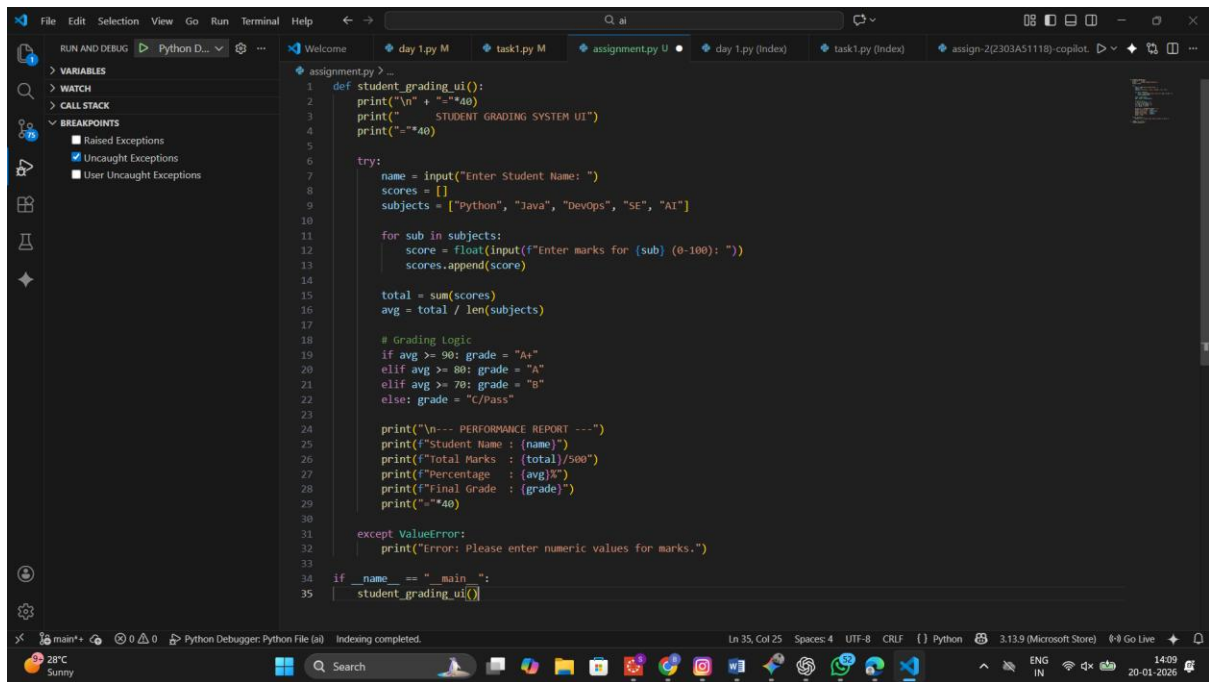
```
PS C:\Users\hp\OneDrive\Desktop\ai> c;; cd 'c:\Users\hp\OneDrive\Desktop\ai'; & 'c:\Users\hp\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\hp\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '52292' '--' 'C:\Users\hp\OneDrive\Desktop\ai\assignment.py'
Is -5 prime? False
Is 1 prime? False
Is 2 prime? True
Is 11 prime? True
Is 25 prime? False
PS C:\Users\hp\OneDrive\Desktop\ai> []
File (ai) Indexing completed. Ln 21, Col 46 Spaces: 4 UTF-8 CRLF {} Python 3.13.9 (Microsoft Store)
```

Task Description-4

- Prompt-Guided UI Design for Student Grading System: Create a user interface for a

student grading system that calculates total marks, percentage, and grade based on user

input.



Expected Output-4

- Well-structured UI code with accurate calculations and clear output display.

```

PS C:\Users\hp\OneDrive\Desktop\ai> c::; cd 'c:\Users\hp\OneDrive\Desktop\ai'; & 'c:\Users\hp\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\hp\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '56742' '--' 'c:\Users\hp\OneDrive\Desktop\ai\assignment.py'

--- PERFORMANCE REPORT ---
Student Name : sai varshith

--- PERFORMANCE REPORT ---
Student Name : sai varshith
--- PERFORMANCE REPORT ---
Student Name : sai varshith
Total Marks : 424.0/500
Percentage : 84.8%
Final Grade : A
Total Marks : 424.0/500
Percentage : 84.8%
Final Grade : A
Final Grade : A
=====
PS C:\Users\hp\OneDrive\Desktop\ai>

```

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.

```
assignment.py > ...
1 def unit_converter(value, mode):
2     """
3     Mode 1: KM to Miles
4     Mode 2: Miles to KM
5     """
6     KM_TO_MILE_FACTOR = 0.621371
7
8     if mode == 1:
9         result = value * KM_TO_MILE_FACTOR
10        return f"{value} KM = {result:.4f} Miles"
11    elif mode == 2:
12        result = value / KM_TO_MILE_FACTOR
13        return f"{value} Miles = {result:.4f} KM"
14    else:
15        return "Invalid Mode Selected"
16
17 print(unit_converter(10, 1)) # Precise Conversion
```

Expected Output-5

- Analysis of code quality and accuracy differences across multiple prompt variations.

```
PS C:\Users\hp\OneDrive\Desktop\ai> c:: cd 'c:\Users\hp\OneDrive\Desktop\ai'; & 'c:\Users\hp\AppData\Local\Microsoft\WindowsApps\python3.13.e
e' 'c:\Users\hp\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundle\libs\debugpy\launcher' '63375' '--' 'C:\Users\hp\OneDrive\Des
top\ai\assignment.py'
10 KM = 6.2137 Miles
PS C:\Users\hp\OneDrive\Desktop\ai>
```