AI Assisted coding Assignment-3.2

Name: Kavati Chaitanya

Ht.no: 2303A51677

Batch. No: 22

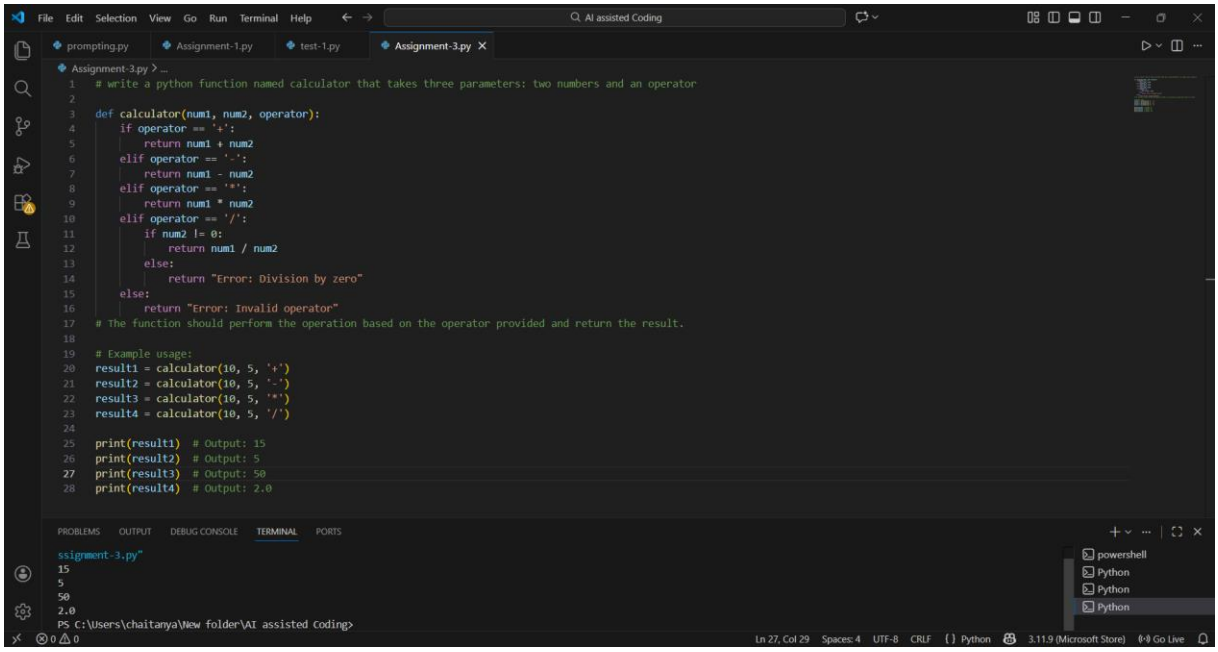
Task Description

• 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.

Prompt:

write a python function named calculator that takes three parameters: two numbers and an operator

Code:



Output-1:

15

5

50

2.0

Prompt used-2:

write a python function named calculator that takes three parameters: two numbers and an operator (as a string: '+', '-', '*', '/').

requirements:

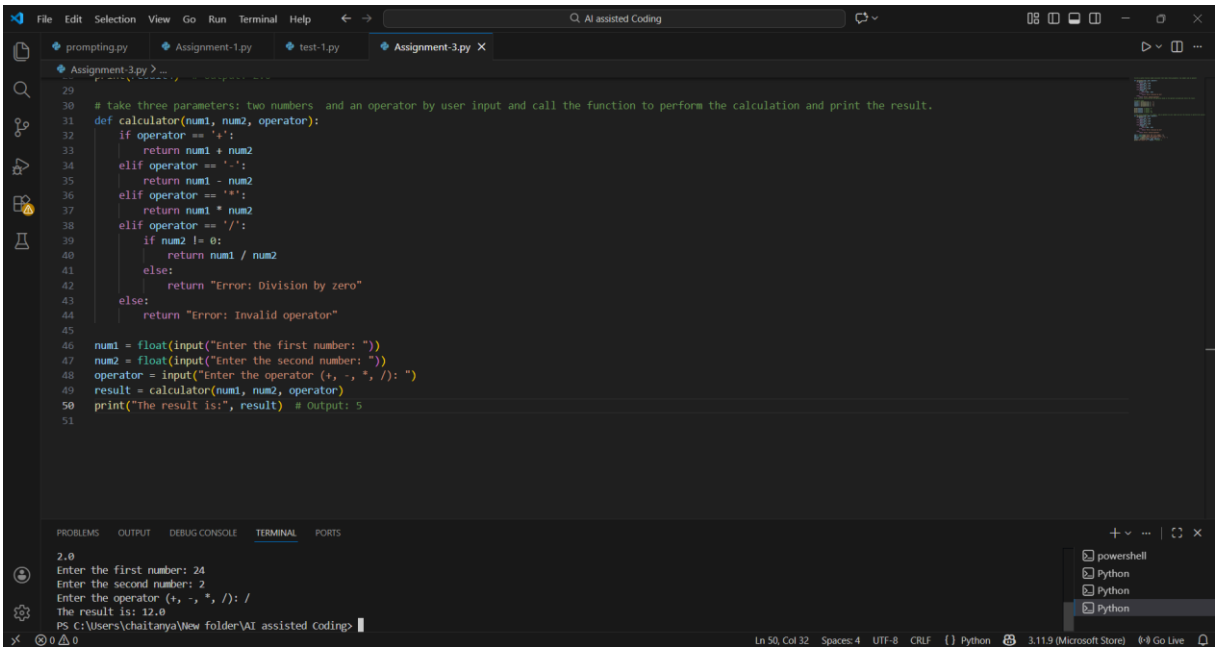
Take 2 numbers and operator as input

Handle division by zero

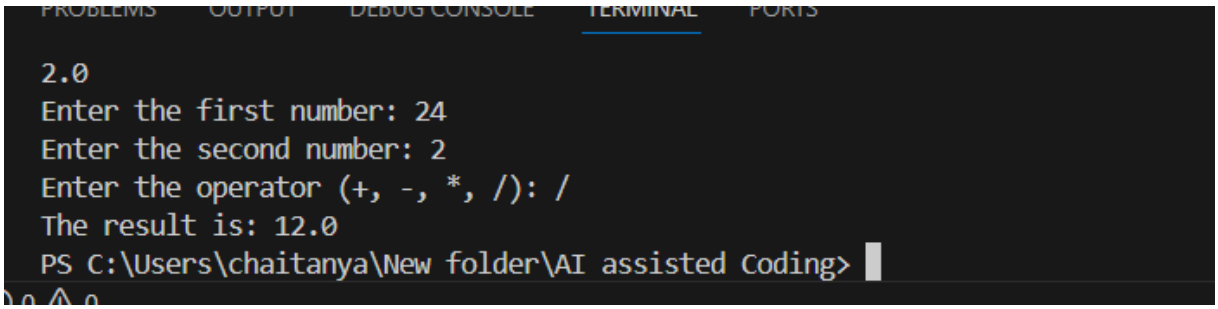
Handle invalid operator inputs

example input: 10, 5, "+" output:15

Code-2:



Output:



Aspect	Initial Calculator	Improvised Calculator
Suitability	Works fine for simple calculations	Better suited for real-world, user-facing use
Readability	Easy to understand but minimal clarity	Cleaner and easier to read with comments
Input Handling	Assumes inputs are already numbers	Safely converts and validates user input
User Feedback	Very limited error messages	Clear and helpful error messages
Code Structure	Basic logic inside the function	Well-structured with proper validation
Robustness	Can break with invalid input	More stable and user-friendly
Error Handling	Handles only basic errors	Covers invalid input, operators, and division by zero

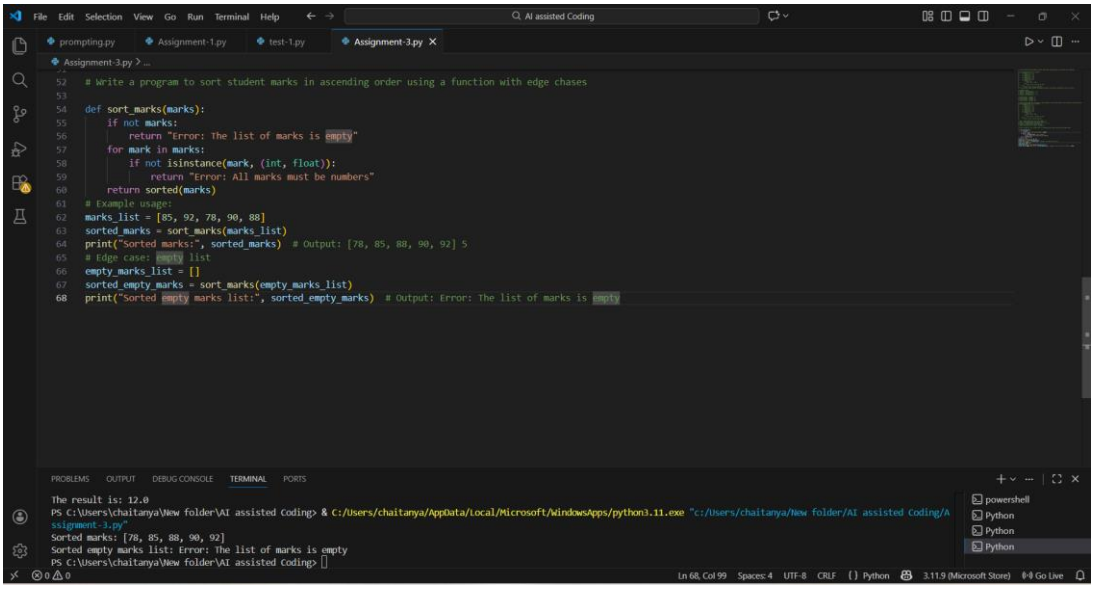
Task Description-2

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

Prompt:

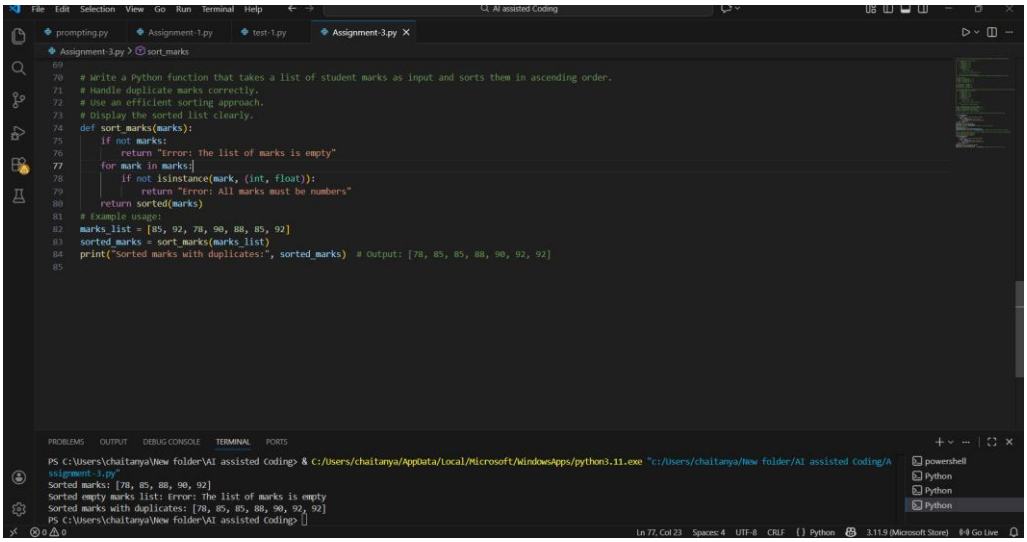
Write a program to sort student marks

Code-1:



Output:

```
The result is: 12.0
PS C:\Users\chaitanya\New folder\AI assisted Coding> & C:/Users/chaitar
ssignment-3.py"
Sorted marks: [78, 85, 88, 90, 92]
Sorted empty marks list: Error: The list of marks is empty
PS C:\Users\chaitanya\New folder\AI assisted Coding>
```



Output-2:

```
ssignment-3.py"
Sorted marks: [78, 85, 88, 90, 92]
Sorted empty marks list: Error: The list of marks is empty
Sorted marks with duplicates: [78, 85, 85, 88, 90, 92, 92]
Sorted invalid marks list: Error: All marks must be numbers
PS C:\Users\chaitanya\New folder\AI assisted Coding>
```

Justification:

By clearly specifying the sorting order and input constraints, the refined prompt improved the correctness and organization of the AI-generated logic.

It guided the AI to better handle numeric values, duplicates, and edge cases such as empty inputs.

This demonstrates that clearer prompts result in more reliable and efficient AI-generated solutions.

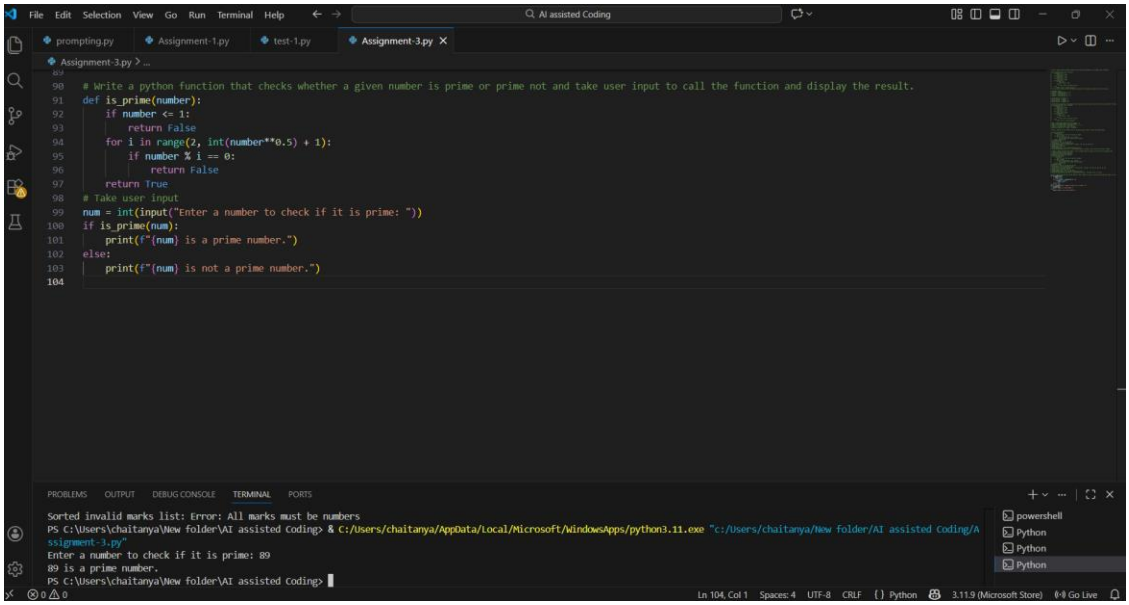
Task Description-3

• 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.

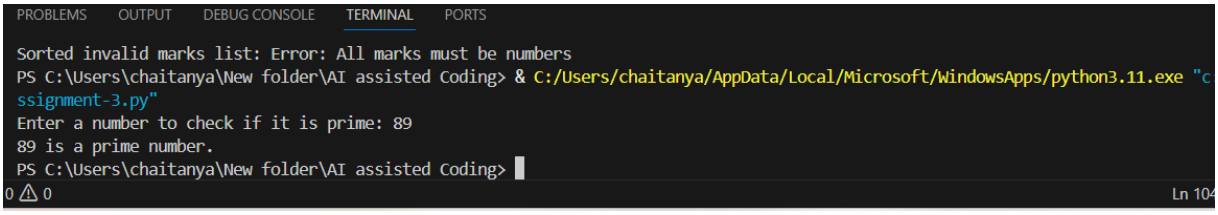
Prompt:

Write a python function that checks whether a given number is prime or prime.

Code:



OUTPUT:



Justification:

By using few-shot prompting, the AI learns prime-checking behavior from example inputs rather than assumptions.
Seeing both valid primes and non-primes helps it avoid common mistakes with values like 0, 1, and negative numbers.
This leads to a prime-checking function that is more correct, consistent, and dependable overall.

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.

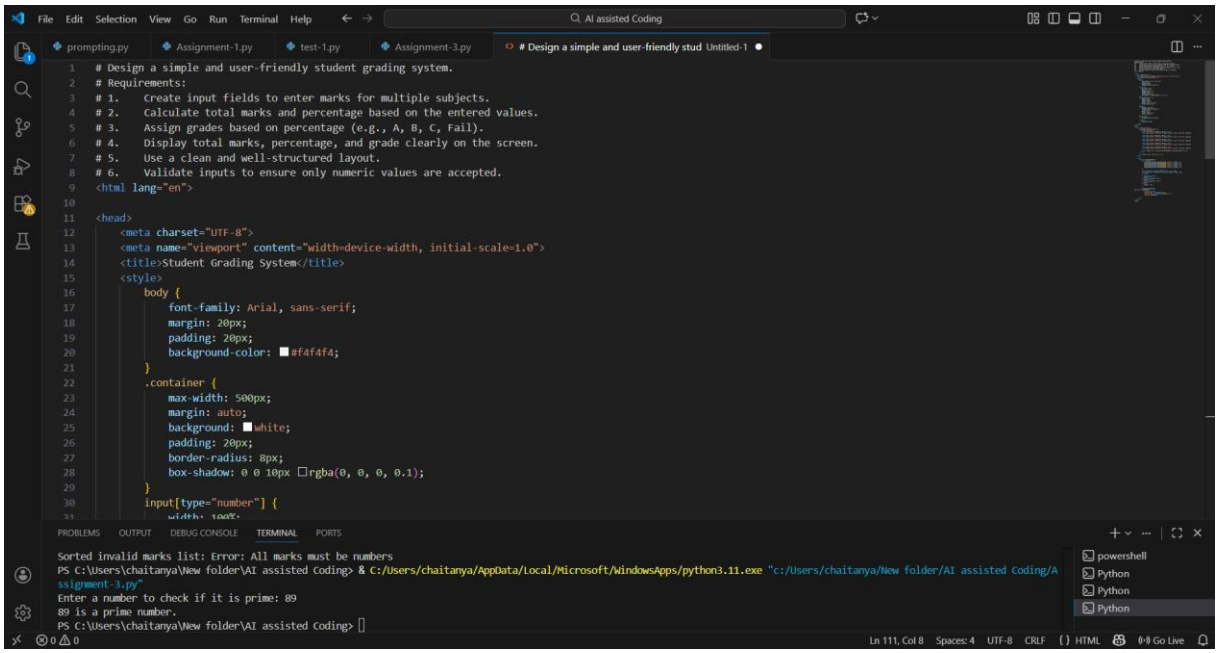
Prompt Used:

Design a simple and user-friendly student grading system UI

Requirements:

1. Create input fields to enter marks for multiple subjects.
2. Calculate total marks and percentage based on the entered values.
3. Assign grades based on percentage (e.g., A, B, C, Fail).
4. Display total marks, percentage, and grade clearly on the screen.
5. Use a clean and well-structured layout.
6. Validate inputs to ensure only numeric values are accepted.
7. Use HTML and CSS for the user interface and JavaScript for calculations. create all files in one file

Code:



prompting.py | Assignment-1.py | test-1.py | Assignment-3.py | # Design a simple and user-friendly stud Untitled-1

```
<html lang="en">
<head>
    <style>
        input[type="number"] {
            width: 100%;
            padding: 10px;
            margin: 10px 0;
            border: 1px solid #ccc;
            border-radius: 4px;
        }
        button {
            padding: 10px 15px;
            background-color: #28a745;
            color: white;
            border: none;
            border-radius: 4px;
            cursor: pointer;
        }
        button:hover {
            background-color: #218838;
        }
        .result {
            margin-top: 20px;
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Student Grading System</h2>
        <form id="gradingForm">
            <label for="subject1">Subject 1 Marks:</label>
```

Student Grading System

Enter Marks for 5 Subjects:

95

85

90

80

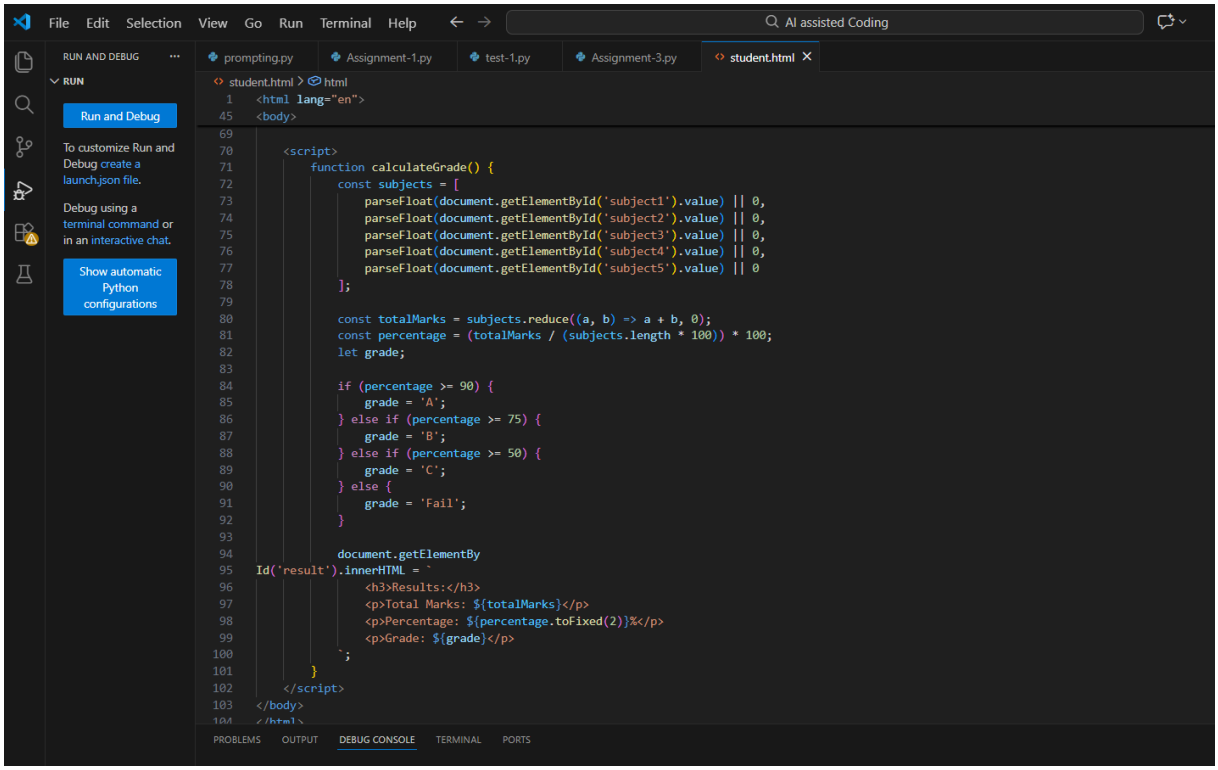
70

Calculate Grade

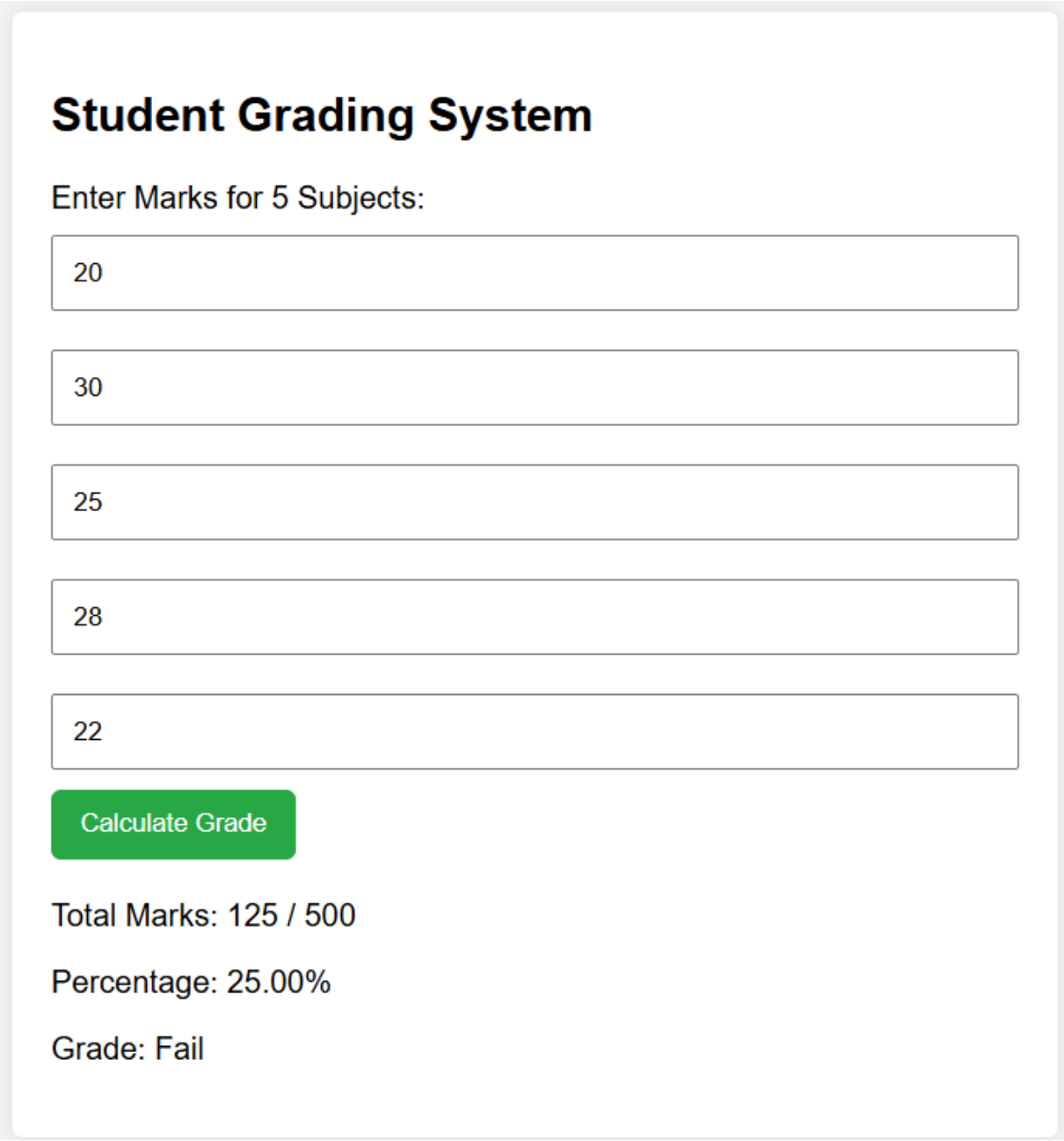
Total Marks: 420 / 500

Percentage: 84.00%

Grade: B



Output:



Justification:

A well-defined prompt helps the UI accurately calculate total marks, percentage, and grades from user input.

Proper input validation increases reliability by effectively managing missing or invalid values.

A structured and organized UI layout improves readability and delivers clear results for both students and evaluators.

Task Description-5

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

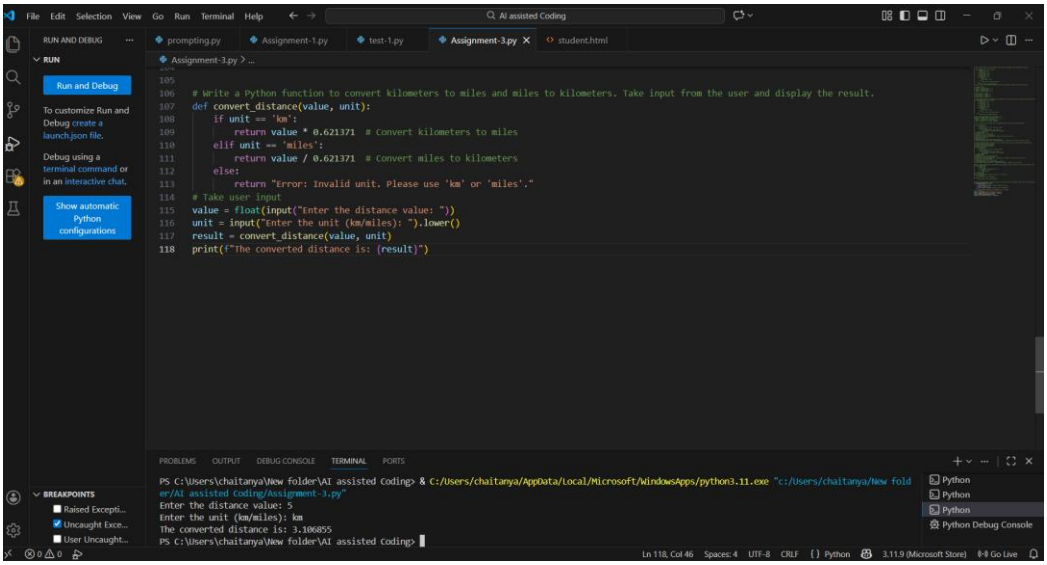
Instructions.

Prompt used-1:

Write a Python function to convert kilometers to miles and miles to kilometers.

Take input from the user and display the result.

Code-1:



Output-1:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\chaitanya\New folder\AI assisted Coding> & C:/Users/chaitanya/AppData/Local/Microsoft/W
er/AI assisted Coding/Assignment-3.py"
Enter the distance value: 5
Enter the unit (km/miles): km
The converted distance is: 3.106855
PS C:\Users\chaitanya\New folder\AI assisted Coding>
```

Prompt used-2:

Write a Python program using functions to convert:

- Kilometers to miles

- Miles to kilometers

Requirements:

- Take numeric input from the user.

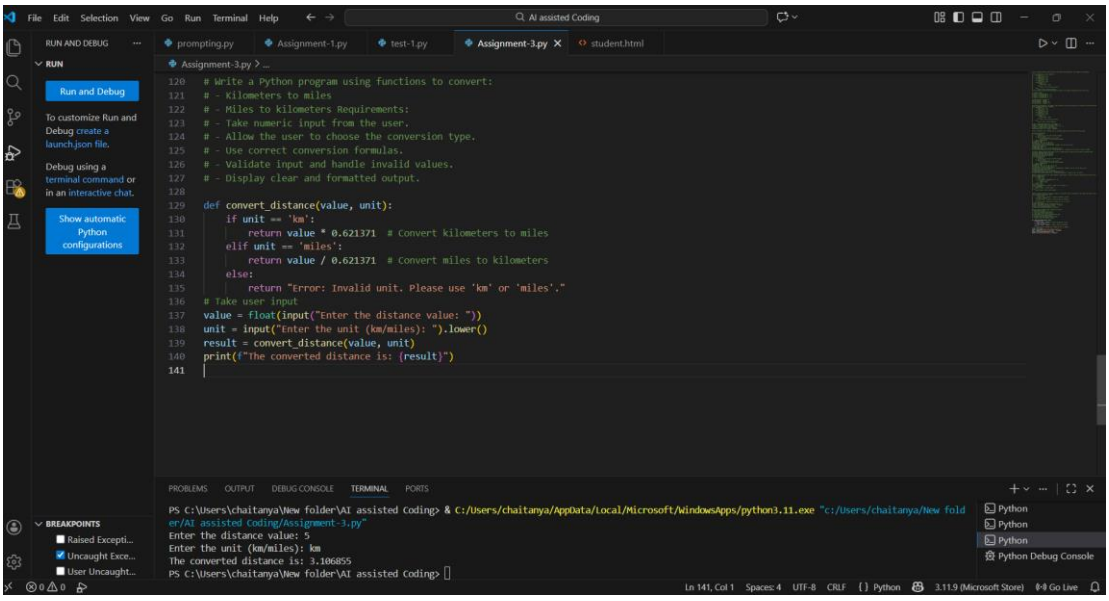
- Allow the user to choose the conversion type.

- Use correct conversion formulas.

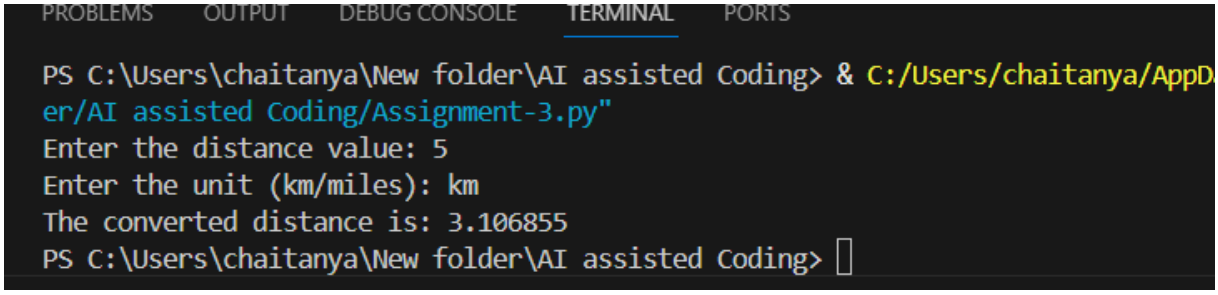
- Validate input and handle invalid values.

- Display clear and formatted output.

Code-2:



Output-2:



Justification:

In the initial prompt, the unit conversion logic gives correct results but allows negative distance values due to weak validation.
With a more detailed prompt, the program improves by enforcing input checks and presenting clearer, well-formatted output.
Overall, greater prompt specificity results in more accurate, robust, and user-friendly unit conversion code