AI ASSISTED CODING

Assignment-3.2

P Sree Varshini

2303A52336

Batch-40

Task Description-1

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

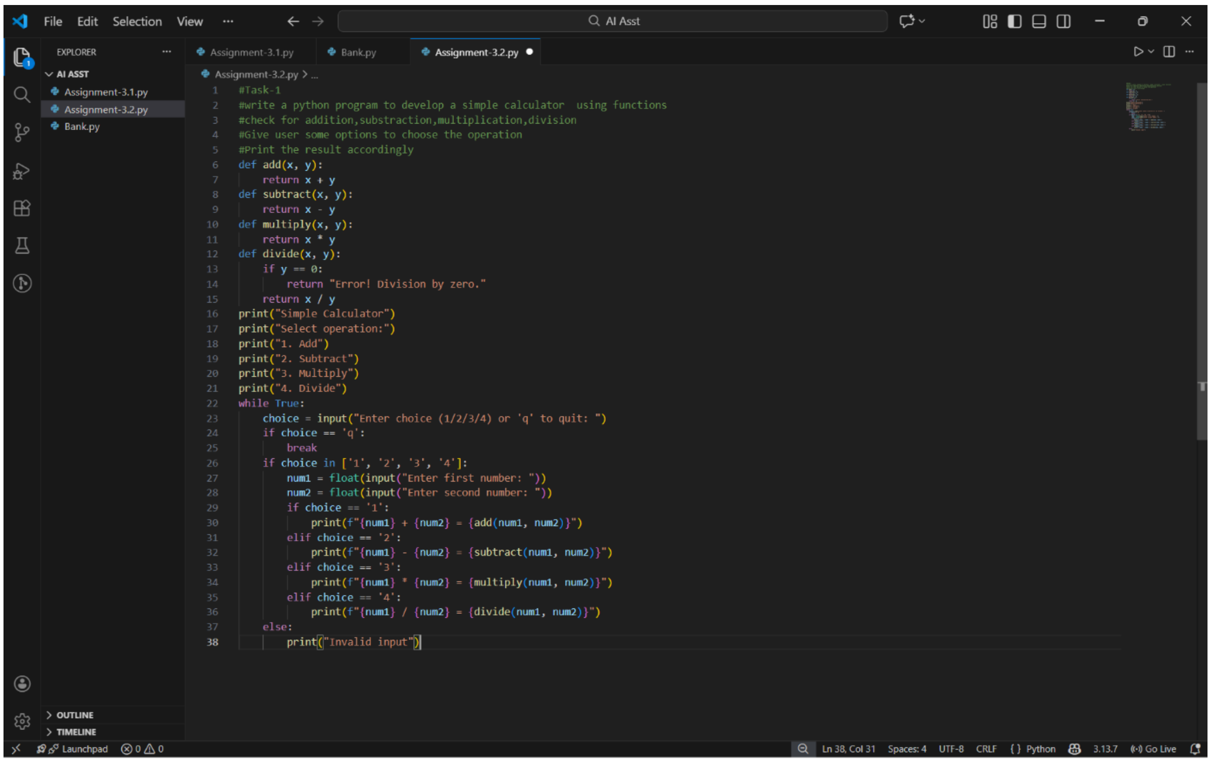
program by initially providing only the function name. Gradually enhance the prompt by

adding comments and usage examples.

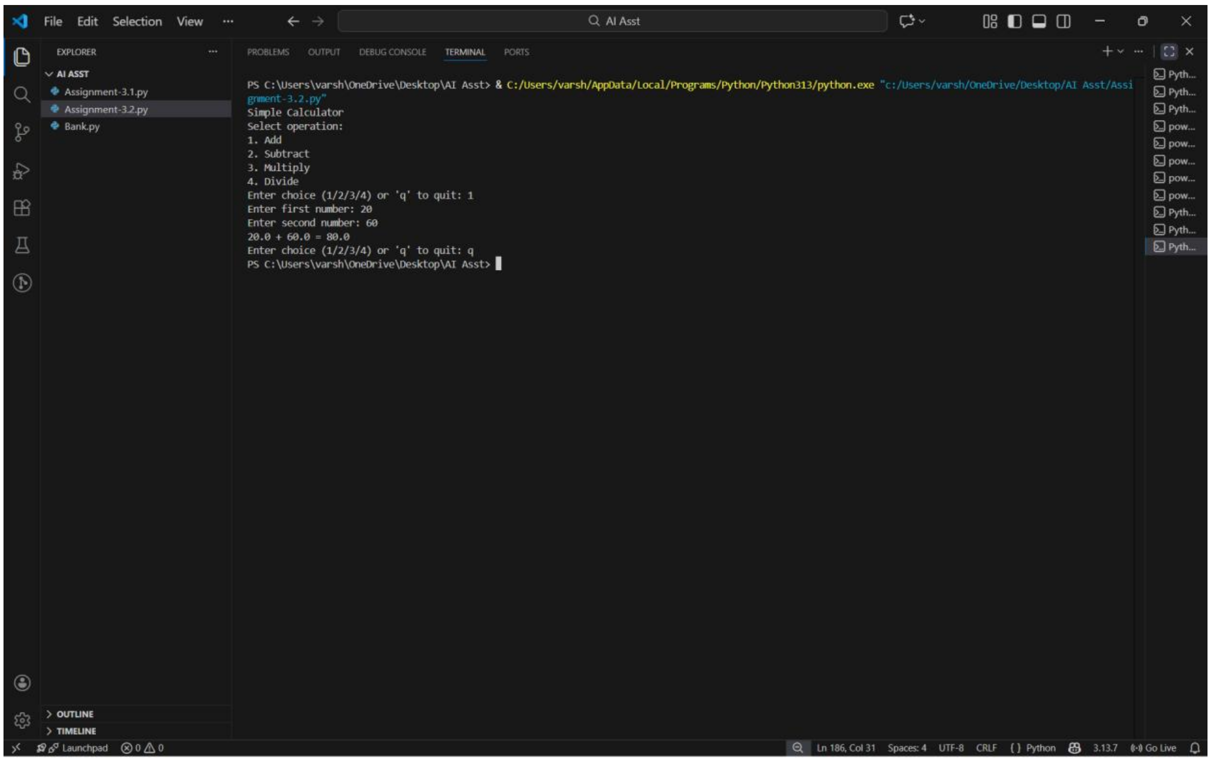
Expected Output-1

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

Code and Output:



```python
#Task-1
#write a python program to develop a simple calculator  using functions
#check for addition,substraction,multiplication,division
#Give user some options to choose the operation
#Print the result accordingly
def add(x, y):
    return x + y
def subtract(x, y):
    return x - y
def multiply(x, y):
    return x * y
def divide(x, y):
    if y == 0:
        return "Error! Division by zero."
    return x / y
print("Simple Calculator")
print("Select operation:")
print("1. Add")
print("2. Subtract")
print("3. Multiply")
print("4. Divide")
while True:
    choice = input("Enter choice (1/2/3/4) or 'q' to quit: ")
    if choice == 'q':
        break
    if choice in ['1', '2', '3', '4']:
        num1 = float(input("Enter first number: "))
        num2 = float(input("Enter second number: "))
        if choice == '1':
            print(f"{num1} + {num2} = {add(num1, num2)}")
        elif choice == '2':
            print(f"{num1} - {num2} = {subtract(num1, num2)}")
        elif choice == '3':
            print(f"{num1} * {num2} = {multiply(num1, num2)}")
        elif choice == '4':
            print(f"{num1} / {num2} = {divide(num1, num2)}")
    else:
        print("Invalid input")
```



```
PS C:\Users\varsh\OneDrive\Desktop\AI Asst> & C:/Users/varsh/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/varsh/OneDrive/Desktop/AI Asst/Assi
gnment-3.2.py"
Simple Calculator
Select operation:
1. Add
2. Subtract
3. Multiply
4. Divide
Enter choice (1/2/3/4) or 'q' to quit: 1
Enter first number: 20
Enter second number: 60
20.0 + 60.0 = 80.0
Enter choice (1/2/3/4) or 'q' to quit: q
PS C:\Users\varsh\OneDrive\Desktop\AI Asst>
```

Analysis:

Analysis: Progressive Prompting for Calculator Design

Progressive prompting means giving instructions step by step to the AI.

First, only the function name is given, so the output is basic.

Next, comments are added to explain what the calculator should do.

Then, usage examples are included to show how the program works.

Each step improves clarity and accuracy of the AI response.

This method helps in getting a complete and correct calculator program.

Task Description-2

• Refining Prompts for Sorting Logic: Start with a vague prompt for sorting student marks,

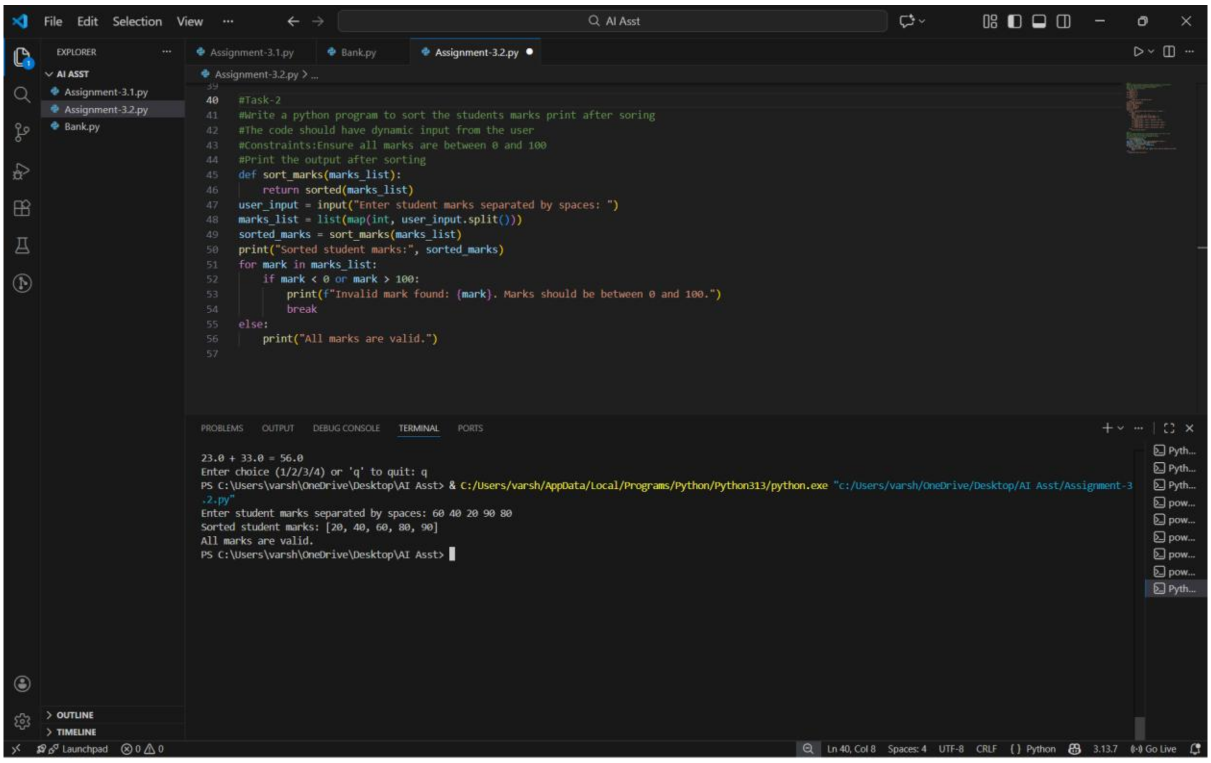
then refine it to clearly specify sorting order and constraints.

Expected Output-2

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

implementation.

Code and Output:



```python
#Task-2
#Write a python program to sort the students marks print after soring
#The code should have dynamic input from the user
#Constraints:Ensure all marks are between 0 and 100
#Print the output after sorting
def sort_marks(marks_list):
    return sorted(marks_list)
user_input = input("Enter student marks separated by spaces: ")
marks_list = list(map(int, user_input.split()))
sorted_marks = sort_marks(marks_list)
print("Sorted student marks:", sorted_marks)
for mark in marks_list:
    if mark < 0 or mark > 100:
        print(f"Invalid mark found: {mark}. Marks should be between 0 and 100.")
        break
else:
    print("All marks are valid.")
```

```
23.0 + 33.0 = 56.0
Enter choice (1/2/3/4) or 'q' to quit: q
PS C:\Users\varsh\OneDrive\Desktop\AI Asst> & C:/Users/varsh/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/varsh/OneDrive/Desktop/AI Asst/Assignment-3
.2.py"
Enter student marks separated by spaces: 60 40 20 90 80
Sorted student marks: [20, 40, 60, 80, 90]
All marks are valid.
PS C:\Users\varsh\OneDrive\Desktop\AI Asst>
```

Analysis:

Refining Prompts for Sorting Logic

Initially, a vague prompt is given to sort student marks.

Due to less clarity, the AI may give an incomplete or unclear solution.

The prompt is then refined to specify sorting order (ascending or descending).

Additional constraints (such as valid marks or number of students) are added.

Clear instructions help the AI produce accurate and expected output.

Refining prompts reduces confusion and errors in the solution.

Task Description-3

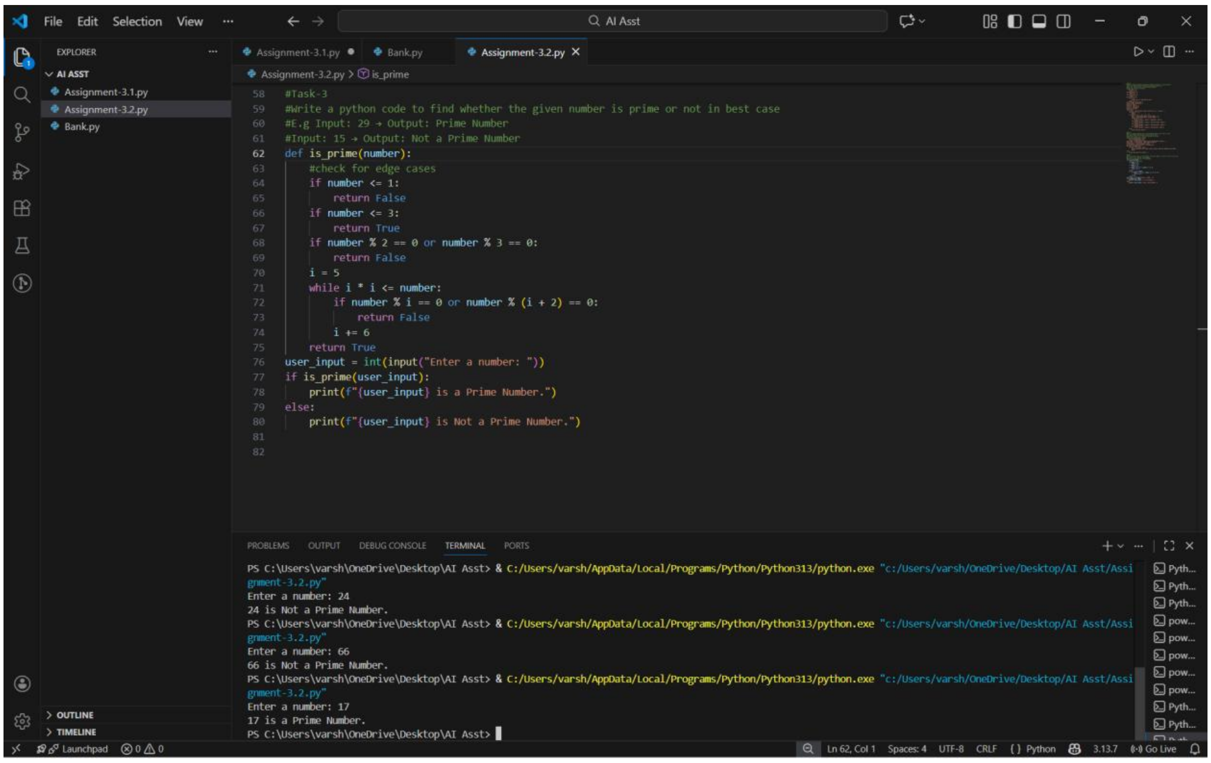
• Few-Shot Prompting for Prime Number Validation: Provide multiple input-output examples for a function that checks whether a number is prime. Observe how few-shot

prompting improves correctness.

Expected Output-3

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

Code and Output:



```python
#Task-3
#Write a python code to find whether the given number is prime or not in best case
#E.g Input: 29 → Output: Prime Number
#Input: 15 → Output: Not a Prime Number
def is_prime(number):
    #check for edge cases
    if number <= 1:
        return False
    if number <= 3:
        return True
    if number % 2 == 0 or number % 3 == 0:
        return False
    i = 5
    while i * i <= number:
        if number % i == 0 or number % (i + 2) == 0:
            return False
        i += 6
    return True
user_input = int(input("Enter a number: "))
if is_prime(user_input):
    print(f"{user_input} is a Prime Number.")
else:
    print(f"{user_input} is Not a Prime Number.")
```

```
PS C:\Users\varsh\OneDrive\Desktop\AI Asst> & C:/Users/varsh/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/varsh/OneDrive/Desktop/AI Asst/Assi
gnment-3.2.py"
Enter a number: 24
24 is Not a Prime Number.
PS C:\Users\varsh\OneDrive\Desktop\AI Asst> & C:/Users/varsh/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/varsh/OneDrive/Desktop/AI Asst/Assi
gnment-3.2.py"
Enter a number: 66
66 is Not a Prime Number.
PS C:\Users\varsh\OneDrive\Desktop\AI Asst> & C:/Users/varsh/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/varsh/OneDrive/Desktop/AI Asst/Assi
gnment-3.2.py"
Enter a number: 17
17 is a Prime Number.
PS C:\Users\varsh\OneDrive\Desktop\AI Asst>
```

Analysis:

Few-Shot Prompting for Prime Number Validation

Few-shot prompting means giving multiple input-output examples to the AI.

The examples show how prime and non-prime numbers behave.

This helps the AI understand the exact logic of prime number checking.

The AI produces more correct and reliable results.

Errors are reduced compared to a prompt with no examples.

Few-shot prompting improves accuracy and consistency of the function.

Task Description-4

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

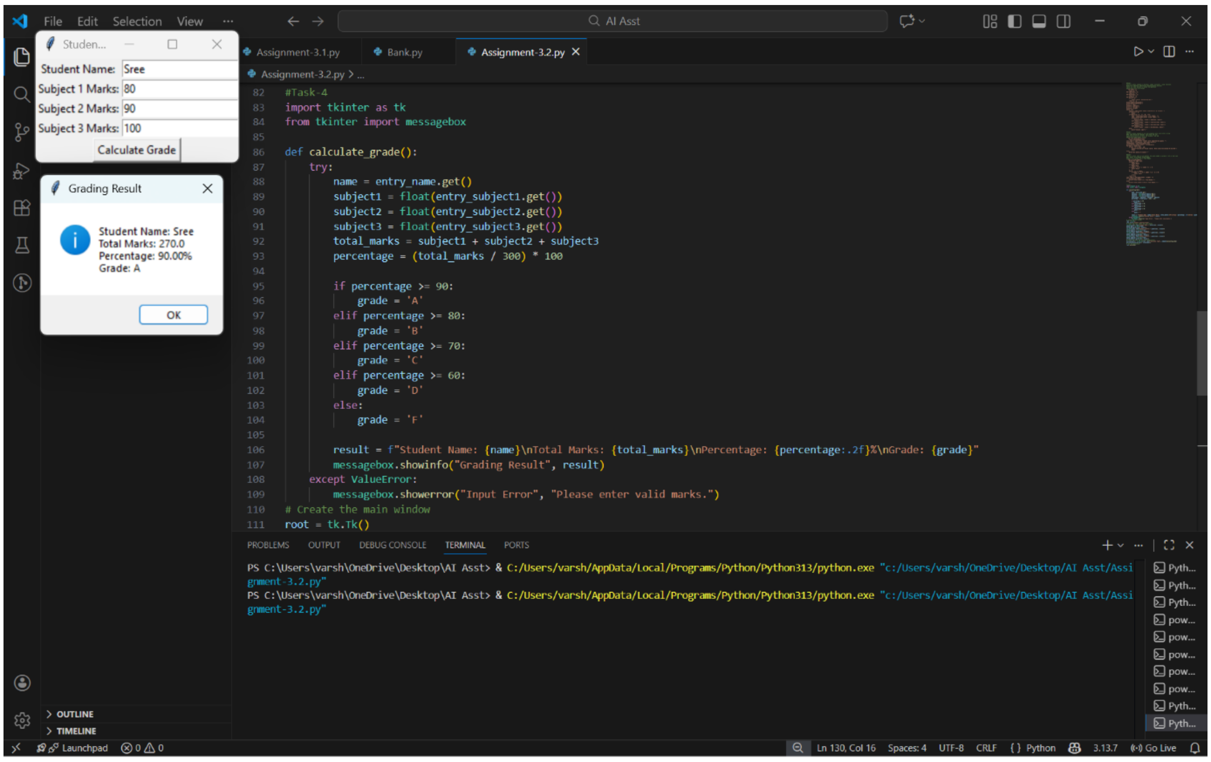
student grading system that calculates total marks, percentage, and grade based on user

input.

Expected Output-4

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

Code and Output:



```
Studen...
Student Name:    Sree
Subject 1 Marks: 80
Subject 2 Marks: 90
Subject 3 Marks: 100
Calculate Grade
```

```
Grading Result
Student Name: Sree
Total Marks: 270.0
Percentage: 90.00%
Grade: A
OK
```

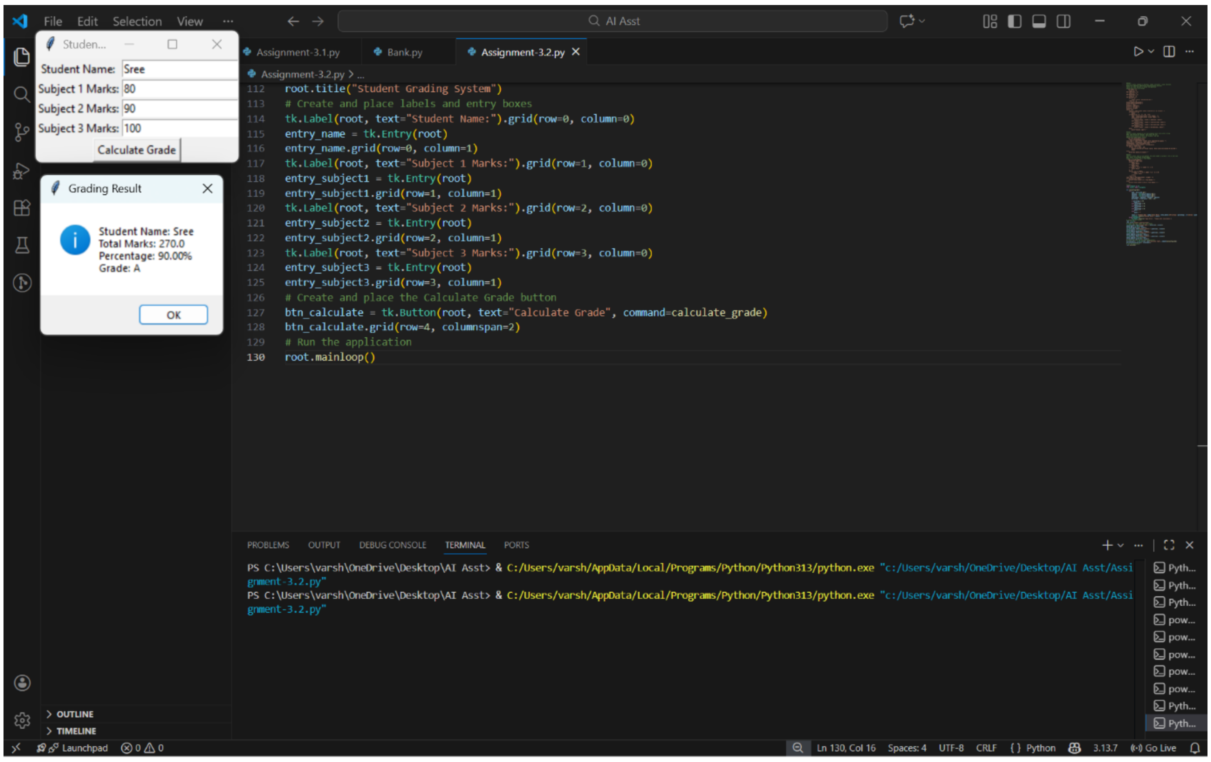
```python
#Task-4
import tkinter as tk
from tkinter import messagebox

def calculate_grade():
    try:
        name = entry_name.get()
        subject1 = float(entry_subject1.get())
        subject2 = float(entry_subject2.get())
        subject3 = float(entry_subject3.get())
        total_marks = subject1 + subject2 + subject3
        percentage = (total_marks / 300) * 100

        if percentage >= 90:
            grade = 'A'
        elif percentage >= 80:
            grade = 'B'
        elif percentage >= 70:
            grade = 'C'
        elif percentage >= 60:
            grade = 'D'
        else:
            grade = 'F'

        result = f"Student Name: {name}\nTotal Marks: {total_marks}\nPercentage: {percentage:.2f}%\nGrade: {grade}"
        messagebox.showinfo("Grading Result", result)
    except ValueError:
        messagebox.showerror("Input Error", "Please enter valid marks.")
# Create the main window
root = tk.Tk()
```

```
PS C:\Users\varsh\OneDrive\Desktop\AI Asst> & C:/Users/varsh/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/varsh/OneDrive/Desktop/AI Asst/Assi
gnment-3.2.py"
PS C:\Users\varsh\OneDrive\Desktop\AI Asst> & C:/Users/varsh/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/varsh/OneDrive/Desktop/AI Asst/Assi
gnment-3.2.py"
```



```
Studen...
Student Name: Sree
Subject 1 Marks: 80
Subject 2 Marks: 90
Subject 3 Marks: 100
Calculate Grade
```

```
Grading Result
Student Name: Sree
Total Marks: 270.0
Percentage: 90.00%
Grade: A
OK
```

```python
root.title("Student Grading System")
# Create and place labels and entry boxes
tk.Label(root, text="Student Name:").grid(row=0, column=0)
entry_name = tk.Entry(root)
entry_name.grid(row=0, column=1)
tk.Label(root, text="Subject 1 Marks:").grid(row=1, column=0)
entry_subject1 = tk.Entry(root)
entry_subject1.grid(row=1, column=1)
tk.Label(root, text="Subject 2 Marks:").grid(row=2, column=0)
entry_subject2 = tk.Entry(root)
entry_subject2.grid(row=2, column=1)
tk.Label(root, text="Subject 3 Marks:").grid(row=3, column=0)
entry_subject3 = tk.Entry(root)
entry_subject3.grid(row=3, column=1)
# Create and place the Calculate Grade button
btn_calculate = tk.Button(root, text="Calculate Grade", command=calculate_grade)
btn_calculate.grid(row=4, columnspan=2)
# Run the application
root.mainloop()
```

```
PS C:\Users\varsh\OneDrive\Desktop\AI Asst> & C:/Users/varsh/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/varsh/OneDrive/Desktop/AI Asst/Assi
gnment-3.2.py"
PS C:\Users\varsh\OneDrive\Desktop\AI Asst> & C:/Users/varsh/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/varsh/OneDrive/Desktop/AI Asst/Assi
gnment-3.2.py"
```

Analysis:

Prompt-Guided UI Design for Student Grading System

Prompt-guided design gives clear instructions for UI creation.

The prompt specifies inputs like student marks.

The system calculates total marks and percentage automatically.

It assigns grades based on rules given in the prompt.

Clear prompts help design a user-friendly interface.

Prompt guidance reduces errors and confusion in UI behavior.

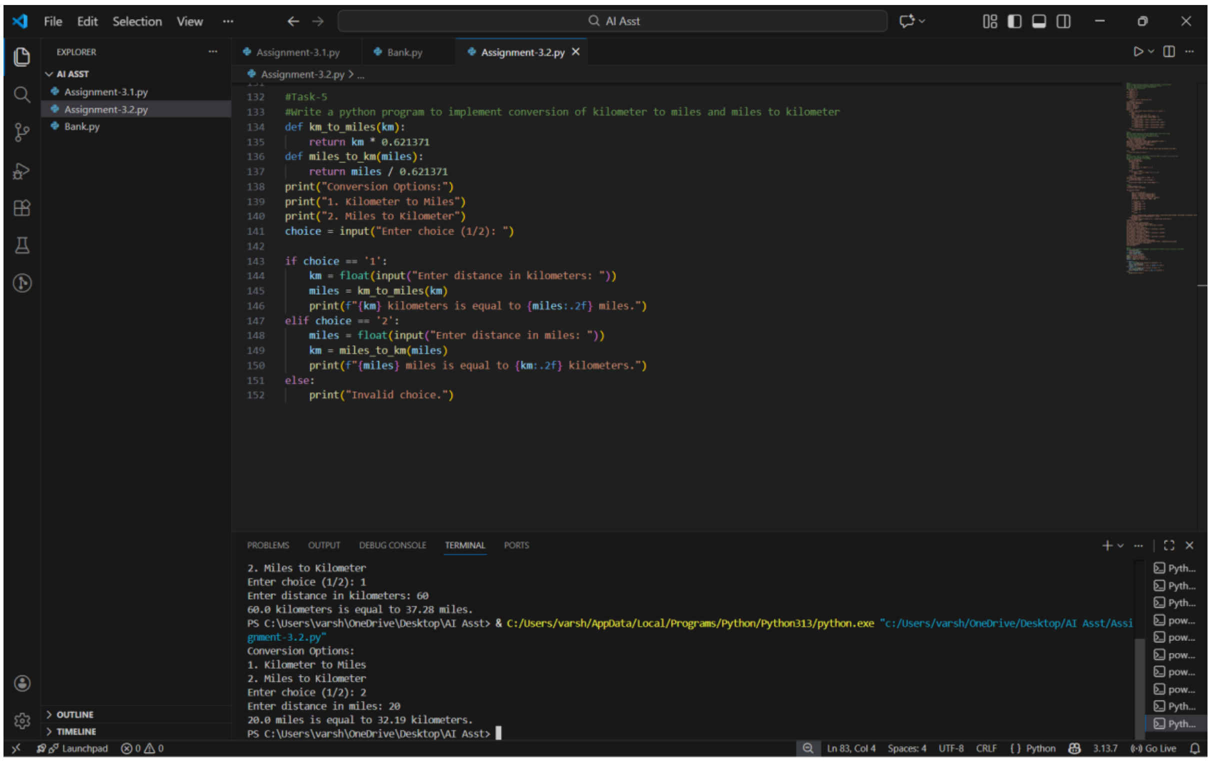
Task Description-5

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

Expected Output-5

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

Code and Output:



```python
#Task-5
#Write a python program to implement conversion of kilometer to miles and miles to kilometer
def km_to_miles(km):
    return km * 0.621371
def miles_to_km(miles):
    return miles / 0.621371
print("Conversion Options:")
print("1. Kilometer to Miles")
print("2. Miles to Kilometer")
choice = input("Enter choice (1/2): ")

if choice == '1':
    km = float(input("Enter distance in kilometers: "))
    miles = km_to_miles(km)
    print(f"{km} kilometers is equal to {miles:.2f} miles.")
elif choice == '2':
    miles = float(input("Enter distance in miles: "))
    km = miles_to_km(miles)
    print(f"{miles} miles is equal to {km:.2f} kilometers.")
else:
    print("Invalid choice.")
```

```
2. Miles to Kilometer
Enter choice (1/2): 1
Enter distance in kilometers: 60
60.0 kilometers is equal to 37.28 miles.
PS C:\Users\varsh\OneDrive\Desktop\AI Asst> & C:/Users/varsh/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/varsh/OneDrive/Desktop/AI Asst/Assi
gnment-3.2.py"
Conversion Options:
1. Kilometer to Miles
2. Miles to Kilometer
Enter choice (1/2): 2
Enter distance in miles: 20
20.0 miles is equal to 32.19 kilometers.
PS C:\Users\varsh\OneDrive\Desktop\AI Asst>
```

Analysis:

Prompt Specificity in Unit Conversion Functions

Prompt specificity means giving clear and detailed instructions.

Initially, a vague prompt may cause incorrect or incomplete conversion.

Clear instructions specify conversion type (km to miles, miles to km).

The prompt defines input and output format clearly.

Specific prompts improve accuracy and correctness of the function.

This reduces errors and misunderstanding in unit conversion.