SR UNIVERSITY

School Of Computer Science and Artificial Intelligence

Lab Assignment – 3.2

Name : Mohammed Sabir

Enrollment No : 2303A51506

Batch No : 22

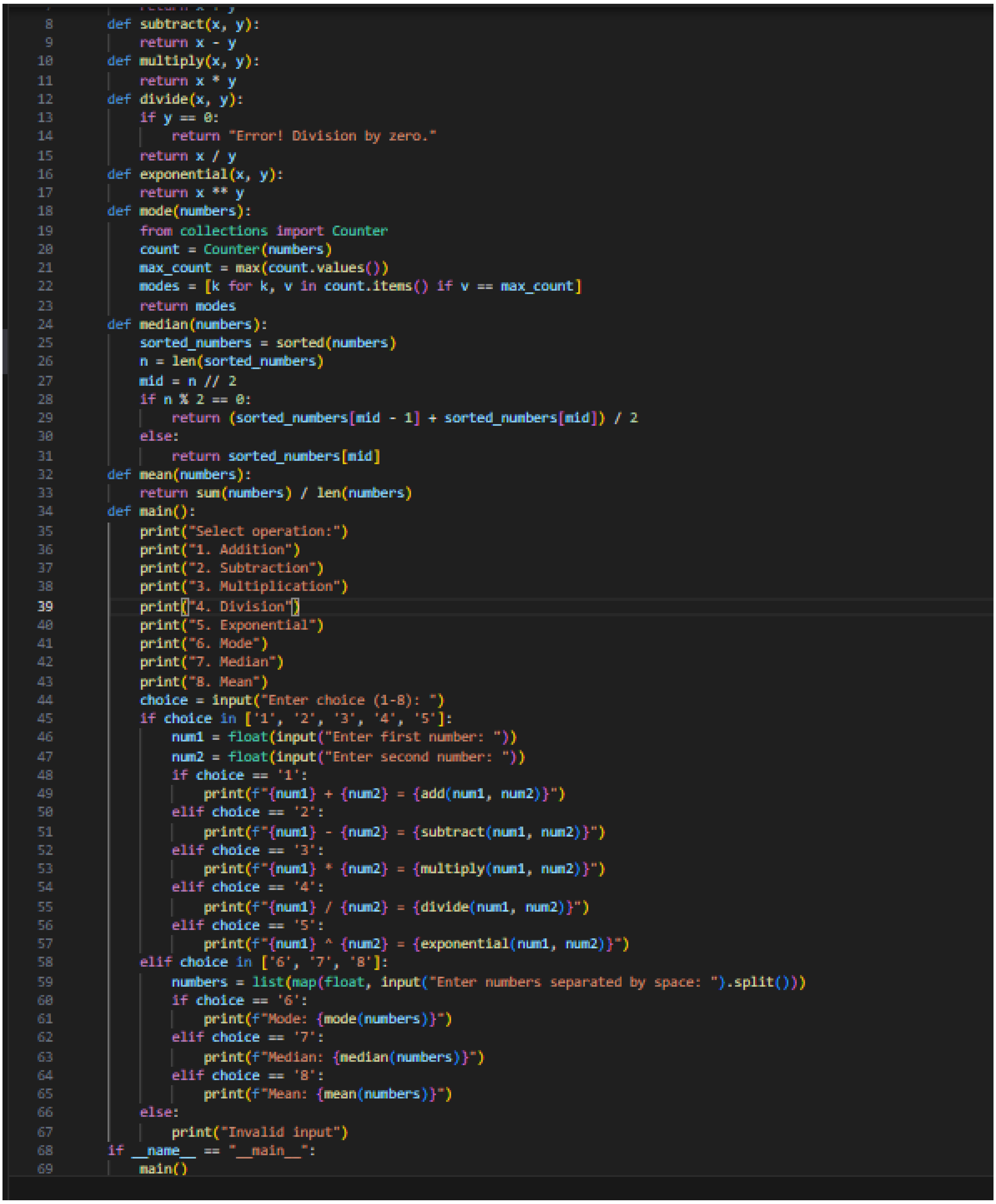
Task : 1

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.

Prompt :-

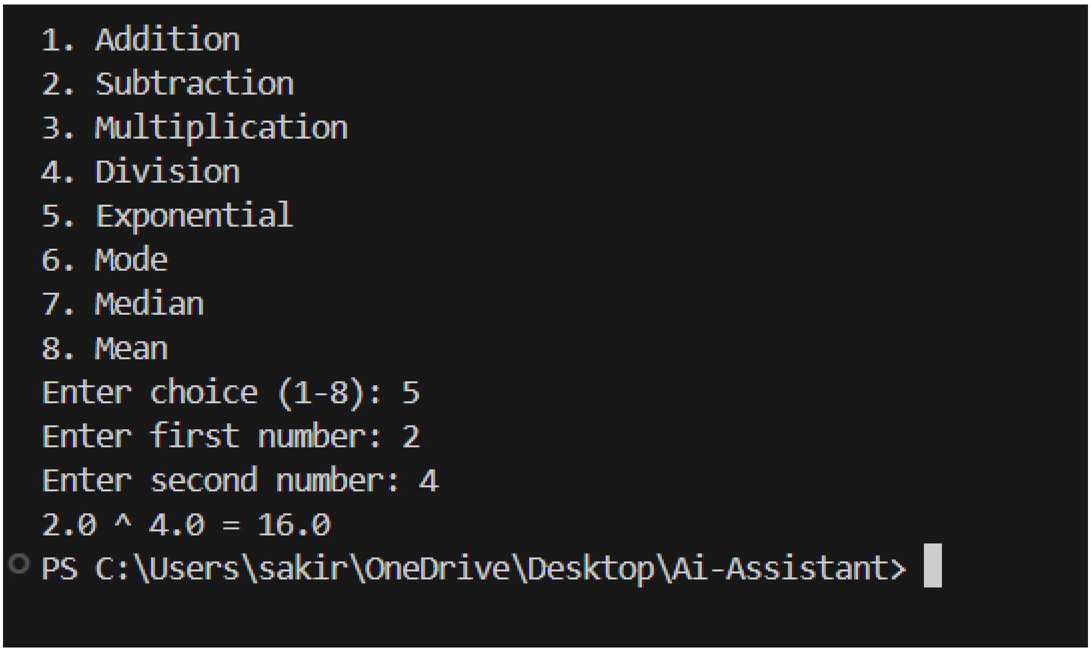
write a python code to implement a simple calculator that can perform addition, subtraction, multiplication, division , exponential , mode , median , and mean operations user should give the input if numbers is in float type it should handle that as well

Code :



```
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
def exponential(x, y):
    return x ** y
def mode(numbers):
    from collections import Counter
    count = Counter(numbers)
    max_count = max(count.values())
    modes = [k for k, v in count.items() if v == max_count]
    return modes
def median(numbers):
    sorted_numbers = sorted(numbers)
    n = len(sorted_numbers)
    mid = n // 2
    if n % 2 == 0:
        return (sorted_numbers[mid - 1] + sorted_numbers[mid]) / 2
    else:
        return sorted_numbers[mid]
def mean(numbers):
    return sum(numbers) / len(numbers)
def main():
    print("Select operation:")
    print("1. Addition")
    print("2. Subtraction")
    print("3. Multiplication")
    print("4. Division")
    print("5. Exponential")
    print("6. Mode")
    print("7. Median")
    print("8. Mean")
    choice = input("Enter choice (1-8): ")
    if choice in ['1', '2', '3', '4', '5']:
        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)}")
        elif choice == '5':
            print(f"{num1} ^ {num2} = {exponential(num1, num2)}")
    elif choice in ['6', '7', '8']:
        numbers = list(map(float, input("Enter numbers separated by space: ").split()))
        if choice == '6':
            print(f"Mode: {mode(numbers)}")
        elif choice == '7':
            print(f"Median: {median(numbers)}")
        elif choice == '8':
            print(f"Mean: {mean(numbers)}")
    else:
        print("Invalid input")
if __name__ == "__main__":
    main()
```

Output :-



```
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exponential
6. Mode
7. Median
8. Mean
Enter choice (1-8): 5
Enter first number: 2
Enter second number: 4
2.0 ^ 4.0 = 16.0
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant>
```

Explanation:-

This Python program is a menu-driven calculator that performs arithmetic operations (addition, subtraction, multiplication, division, exponential) and statistical calculations (mode, median, mean).
Each operation is written as a separate function to keep the code clean and reusable.
The main() function shows a menu, takes user input, and calls the appropriate function based on the selected option.
It also handles errors like division by zero and invalid menu choices to avoid crashes.

Task : 2

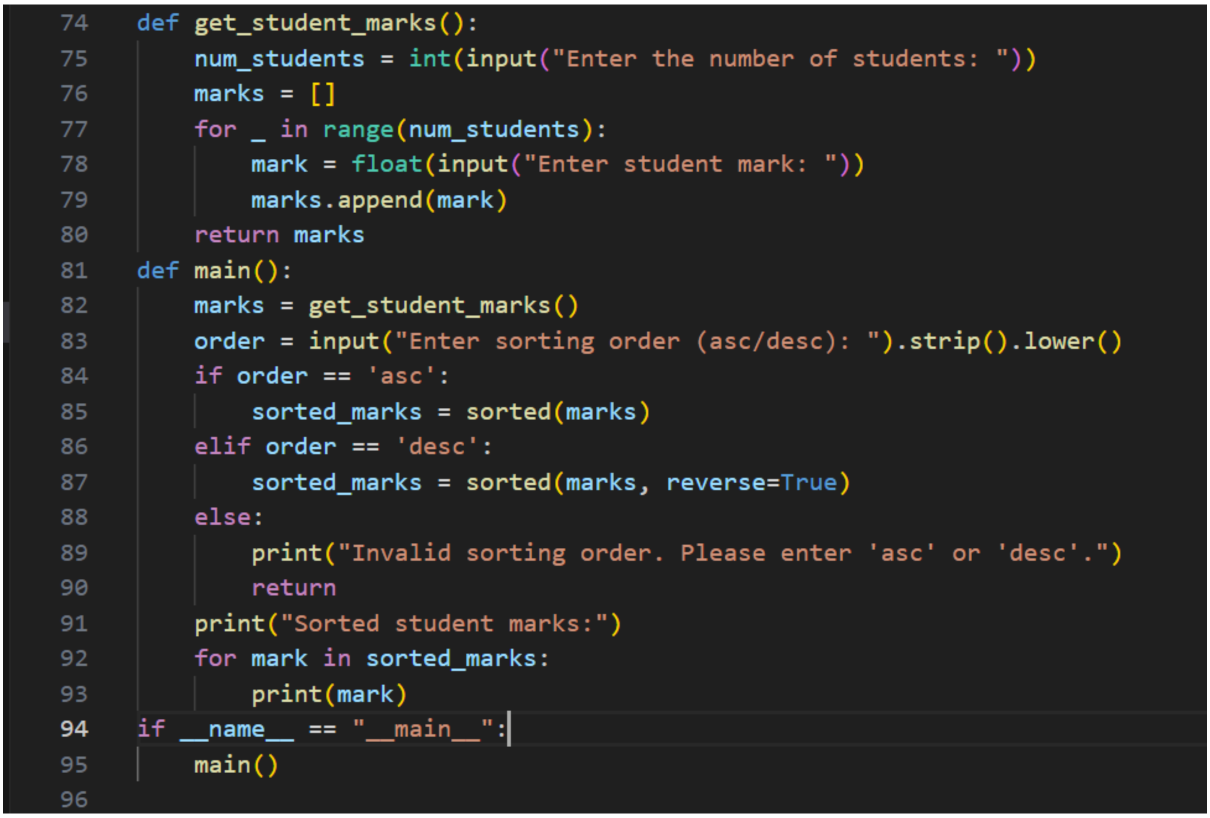
Refining Prompts for Sorting Logic: Start with a vague prompt for sorting student marks,

then refine it to clearly specify sorting order and constraints.

Prompt:-

#write a python code for student marksheet but the marks should be in sorted order then refine it clearly specify sorting order and constraints but take it as a user define input

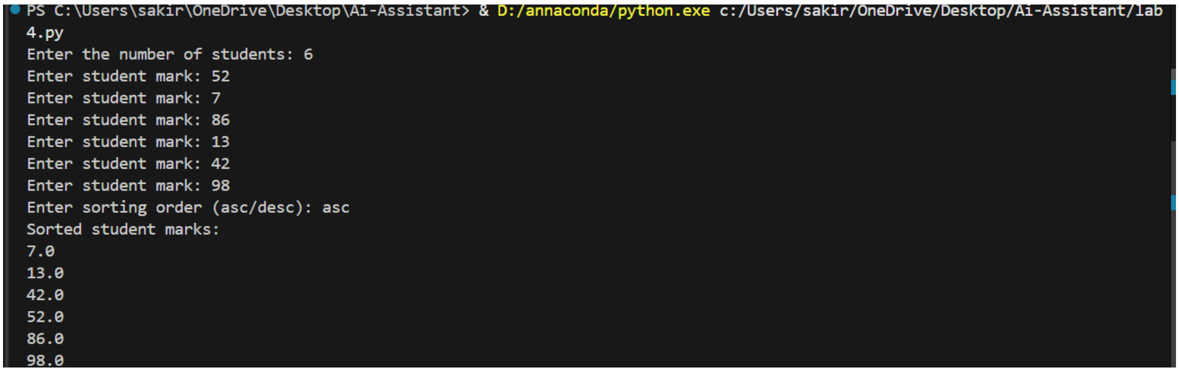
Code:-



```python
def get_student_marks():
    num_students = int(input("Enter the number of students: "))
    marks = []
    for _ in range(num_students):
        mark = float(input("Enter student mark: "))
        marks.append(mark)
    return marks
def main():
    marks = get_student_marks()
    order = input("Enter sorting order (asc/desc): ").strip().lower()
    if order == 'asc':
        sorted_marks = sorted(marks)
    elif order == 'desc':
        sorted_marks = sorted(marks, reverse=True)
    else:
        print("Invalid sorting order. Please enter 'asc' or 'desc'.")
        return
    print("Sorted student marks:")
    for mark in sorted_marks:
        print(mark)
if __name__ == "__main__":
    main()

```

Output:-



```
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant> & D:/annaconda/python.exe c:/Users/sakir/OneDrive/Desktop/Ai-Assistant/lab
4.py
Enter the number of students: 6
Enter student mark: 52
Enter student mark: 7
Enter student mark: 86
Enter student mark: 13
Enter student mark: 42
Enter student mark: 98
Enter sorting order (asc/desc): asc
Sorted student marks:
7.0
13.0
42.0
52.0
86.0
98.0
```

Explanation:-

This program takes marks for a given number of students and stores them in a list.
It then asks the user whether to sort the marks in ascending (asc) or descending (desc) order.
Based on the user's choice, it sorts the list using Python's sorted() function.
Finally, it prints the sorted student marks, and handles invalid sorting input safely.

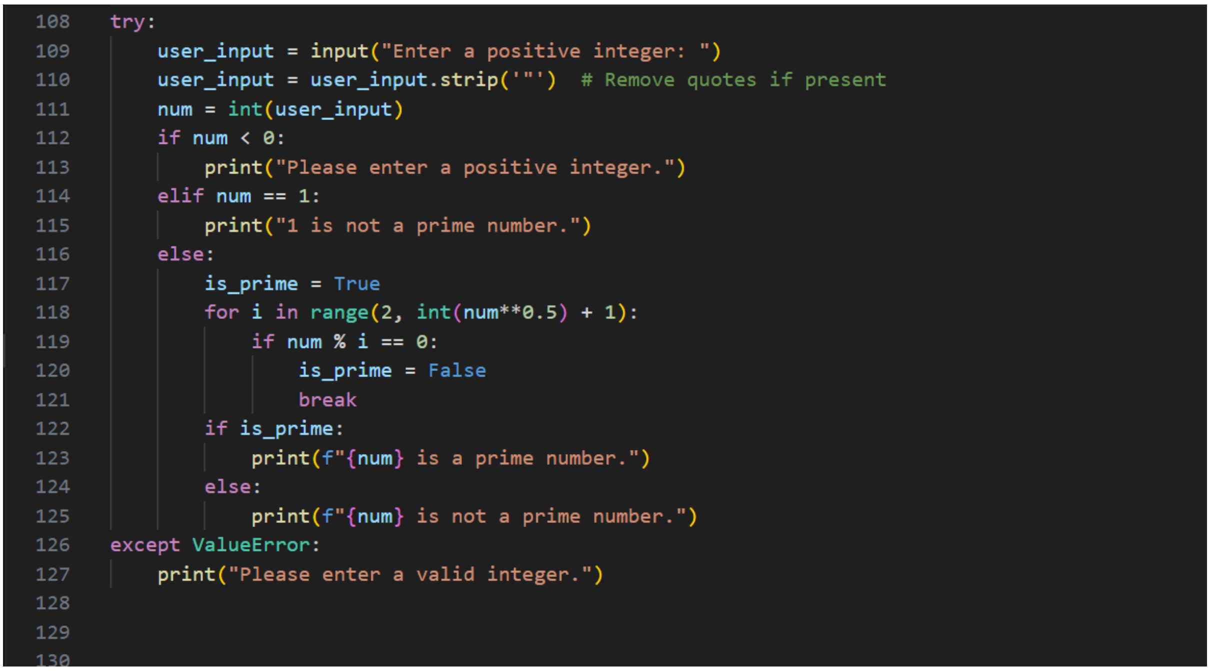
Task:3

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.

Prompt:-

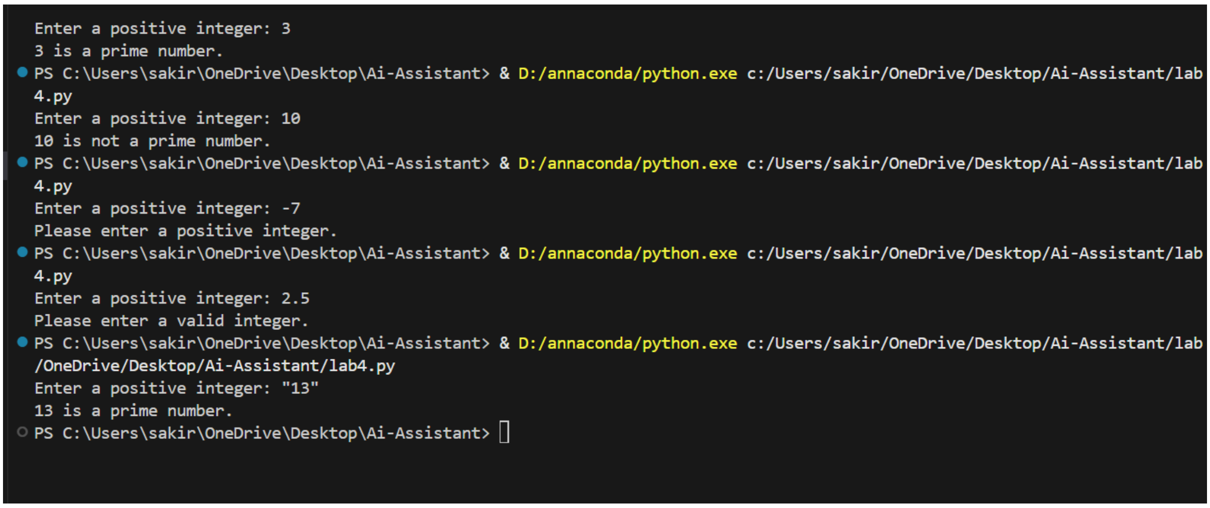
''' write a python program to check whether a given number is prime or not if i gave input as "11" it should consider as 11 and check for prime number handle the error if user give a string input or float input or negative number input example 1: input 7 output : 7 is a prime number,example 2: input 10 output : 10 is not a prime number , example 3: input -5 output : please enter a positive integer , example 4: input abc output : please enter a valid integer , example 5: input 1 output : 1 is not a prime number , example 6: input 2.34 output : please enter a valid integer , example 7: input "7" output : 7 is a prime number

Code:-



```
try:
    user_input = input("Enter a positive integer: ")
    user_input = user_input.strip('"')  # Remove quotes if present
    num = int(user_input)
    if num < 0:
        print("Please enter a positive integer.")
    elif num == 1:
        print("1 is not a prime number.")
    else:
        is_prime = True
        for i in range(2, int(num**0.5) + 1):
            if num % i == 0:
                is_prime = False
                break
        if is_prime:
            print(f"{num} is a prime number.")
        else:
            print(f"{num} is not a prime number.")
except ValueError:
    print("Please enter a valid integer.")
```

Output:-



```
Enter a positive integer: 3
3 is a prime number.
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant> & D:/annaconda/python.exe c:/Users/sakir/OneDrive/Desktop/Ai-Assistant/lab
4.py
Enter a positive integer: 10
10 is not a prime number.
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant> & D:/annaconda/python.exe c:/Users/sakir/OneDrive/Desktop/Ai-Assistant/lab
4.py
Enter a positive integer: -7
Please enter a positive integer.
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant> & D:/annaconda/python.exe c:/Users/sakir/OneDrive/Desktop/Ai-Assistant/lab
4.py
Enter a positive integer: 2.5
Please enter a valid integer.
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant> & D:/annaconda/python.exe c:/Users/sakir/OneDrive/Desktop/Ai-Assistant/lab
/OneDrive/Desktop/Ai-Assistant/lab4.py
Enter a positive integer: "13"
13 is a prime number.
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant> 
```

Explanation:-

This program takes a user input and safely converts it into an integer using try–except.
It checks special cases like negative numbers and 1, which is not a prime.
For other numbers, it tests divisibility from 2 to the square root of the number to determine primality efficiently.
Finally, it prints whether the given number is a prime number or not, and handles invalid input gracefully

---

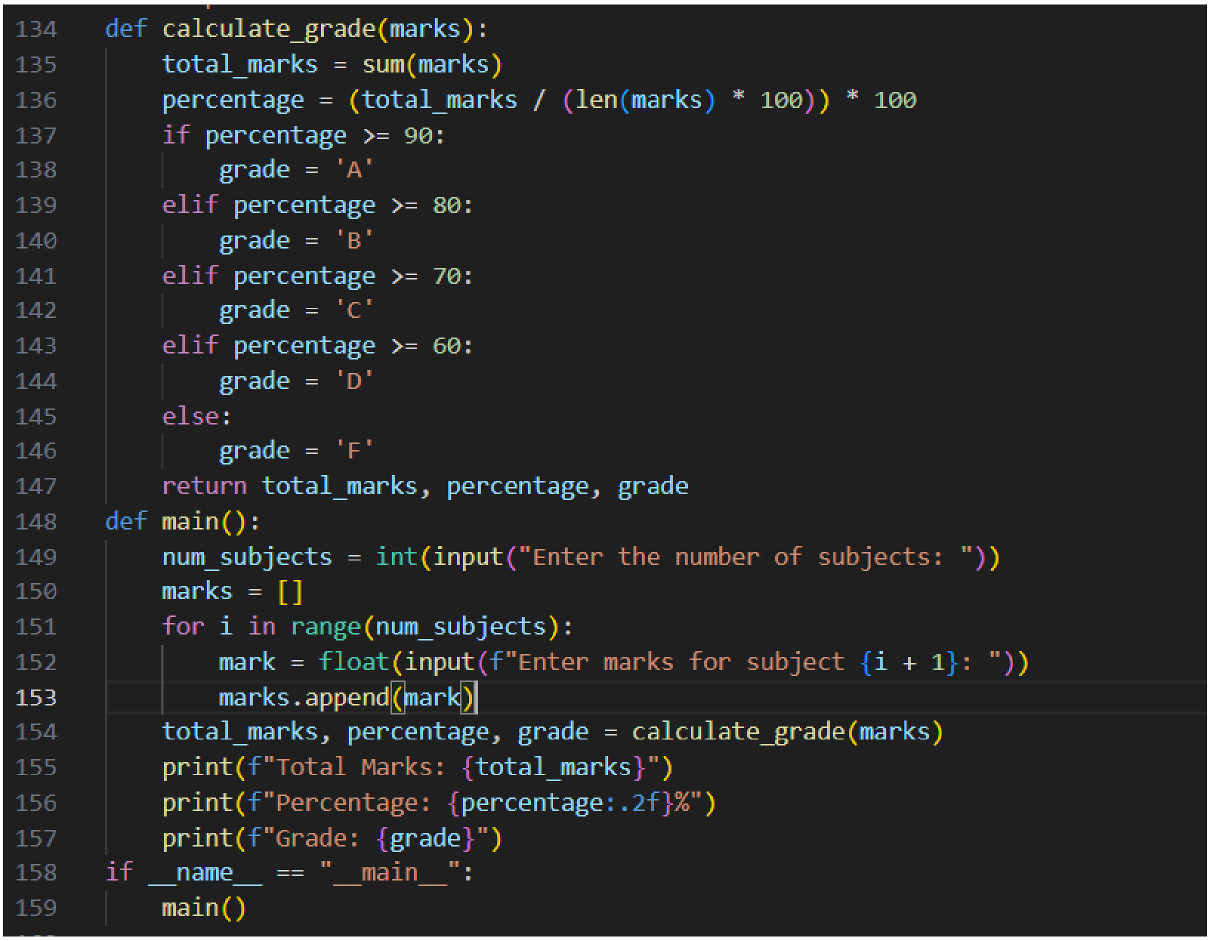
Task : 4

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:-

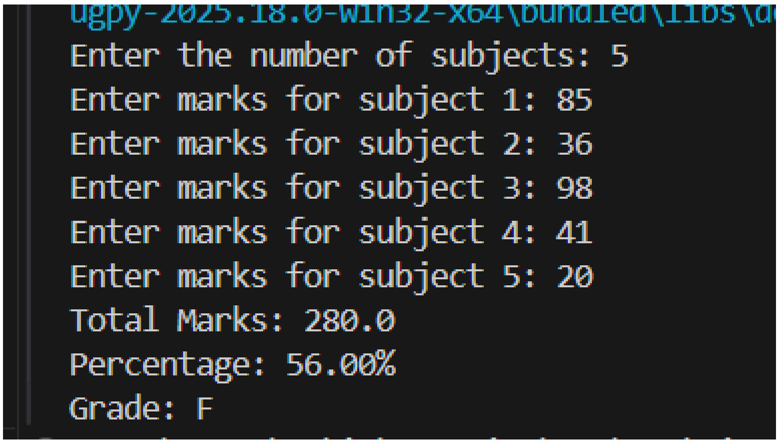
write a python program and create a user interface for student grading system that calculates total marks, percentage and grade based on user input

Code:-



```python
def calculate_grade(marks):
    total_marks = sum(marks)
    percentage = (total_marks / (len(marks) * 100)) * 100
    if percentage >= 90:
        grade = 'A'
    elif percentage >= 80:
        grade = 'B'
    elif percentage >= 70:
        grade = 'C'
    elif percentage >= 60:
        grade = 'D'
    else:
        grade = 'F'
    return total_marks, percentage, grade
def main():
    num_subjects = int(input("Enter the number of subjects: "))
    marks = []
    for i in range(num_subjects):
        mark = float(input(f"Enter marks for subject {i + 1}: "))
        marks.append(mark)
    total_marks, percentage, grade = calculate_grade(marks)
    print(f"Total Marks: {total_marks}")
    print(f"Percentage: {percentage:.2f}%")
    print(f"Grade: {grade}")
if __name__ == "__main__":
    main()
```

Output:-



```
ugpy-2025.18.0-win32-x64\bundled\libs\
Enter the number of subjects: 5
Enter marks for subject 1: 85
Enter marks for subject 2: 36
Enter marks for subject 3: 98
Enter marks for subject 4: 41
Enter marks for subject 5: 20
Total Marks: 280.0
Percentage: 56.00%
Grade: F
```

Explanation:-

This program collects marks for multiple subjects and stores them in a list.
It calculates the total marks, percentage, and assigns a grade based on the percentage range.
The grading logic uses conditional statements to map percentages to grades from A to F.
Finally, it displays the total marks, formatted percentage, and the final grade

Task : 5

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

Prompt:-

Write a Python function that converts kilometers to miles and miles to kilometers.

The function should take two parameters: the value to be converted and the unit of the value (either "km" for kilometers or "mi" for miles). The function should return the converted value based on the specified unit. Additionally, include error handling to manage invalid inputs, such as non-numeric values or unsupported units.

Example usage:

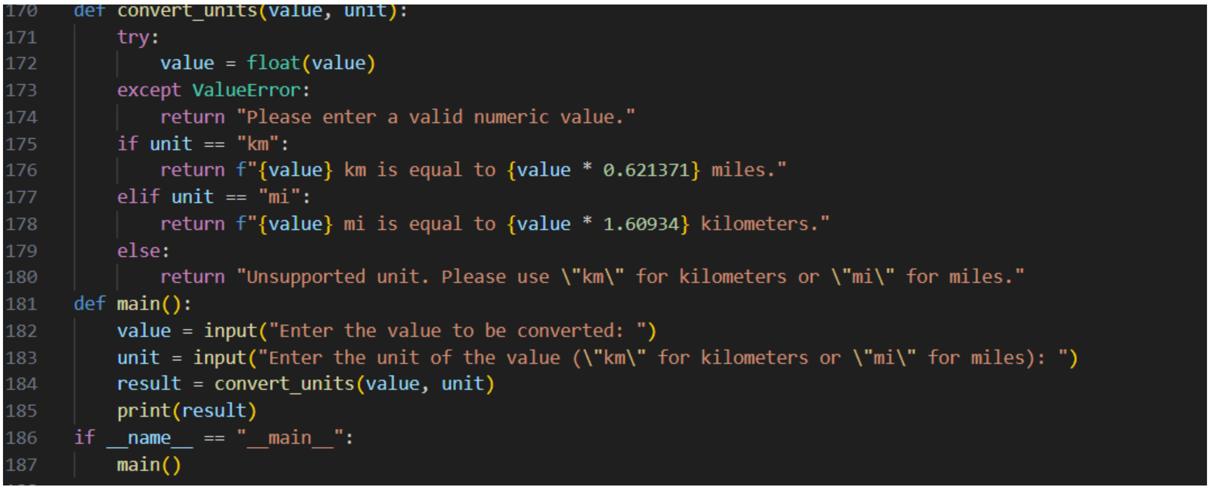
convert_units(5, "km") # Output: 3.10686 miles

convert_units(3, "mi") # Output: 4.82802 kilometers

convert_units("abc", "km") # Output: Please enter a valid numeric value.

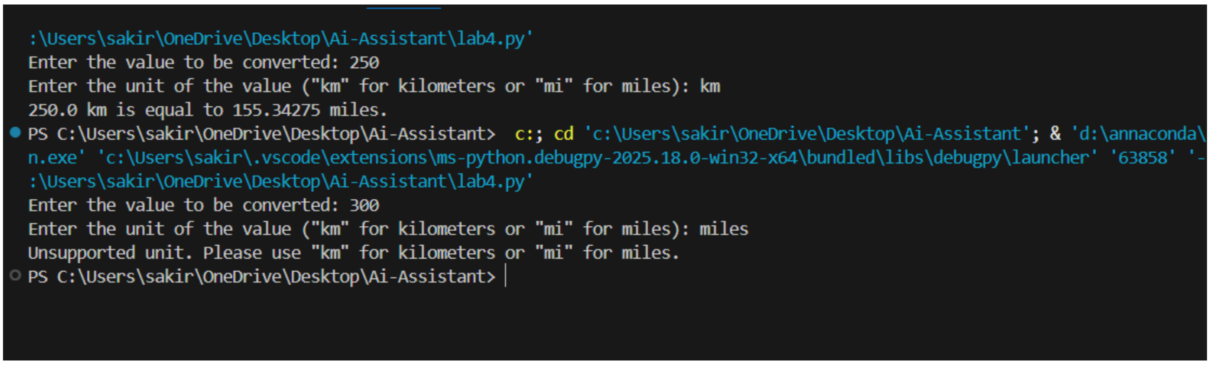
convert_units(5, "miles") # Output: Unsupported unit. Please use "km" for kilometers or "mi" for miles

Code:-



```
def convert_units(value, unit):
    try:
        value = float(value)
    except ValueError:
        return "Please enter a valid numeric value."
    if unit == "km":
        return f"{value} km is equal to {value * 0.621371} miles."
    elif unit == "mi":
        return f"{value} mi is equal to {value * 1.60934} kilometers."
    else:
        return "Unsupported unit. Please use \"km\" for kilometers or \"mi\" for miles."
def main():
    value = input("Enter the value to be converted: ")
    unit = input("Enter the unit of the value (\"km\" for kilometers or \"mi\" for miles): ")
    result = convert_units(value, unit)
    print(result)
if __name__ == "__main__":
    main()
```

Output:-



```
:\Users\sakir\OneDrive\Desktop\Ai-Assistant\lab4.py'
Enter the value to be converted: 250
Enter the unit of the value ("km" for kilometers or "mi" for miles): km
250.0 km is equal to 155.34275 miles.
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant>  c:; cd 'c:\Users\sakir\OneDrive\Desktop\Ai-Assistant'; & 'd:\annaconda\
n.exe' 'c:\Users\sakir\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '63858' '-
:\Users\sakir\OneDrive\Desktop\Ai-Assistant\lab4.py'
Enter the value to be converted: 300
Enter the unit of the value ("km" for kilometers or "mi" for miles): miles
Unsupported unit. Please use "km" for kilometers or "mi" for miles.
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant> 
```

Explanation:-

This program converts distance values between kilometers and miles.
It safely converts the input value to a number using error handling.
Based on the unit entered (km or mi), it performs the appropriate conversion using standard formulas.
If the unit or value is invalid, it displays a clear error message.