Assignment => 2

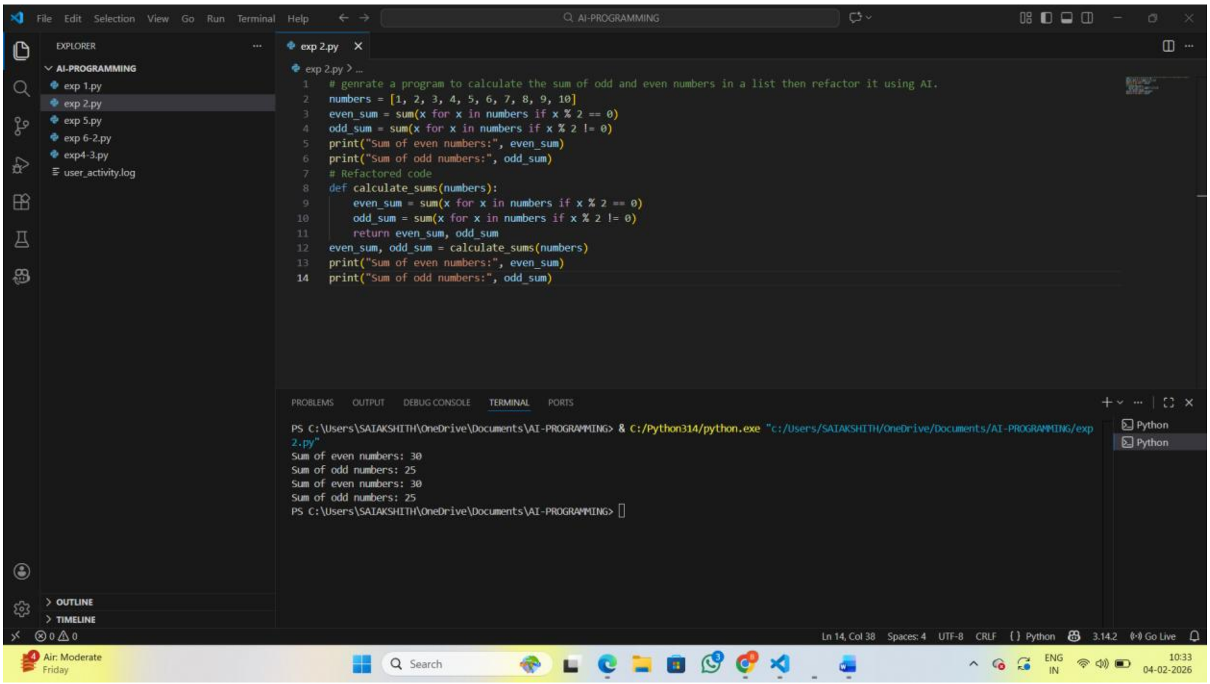
Name =>Sai akshith

Batch=>45

Task 1: Refactoring Odd/Even Logic (List Version)

Code

```
# genrate a program to calculate the sum of odd and even numbers in a list then refactor it using AI.

numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

even_sum = sum(x for x in numbers if x % 2 == 0)

odd_sum = sum(x for x in numbers if x % 2 != 0)

print("Sum of even numbers:", even_sum)

print("Sum of odd numbers:", odd_sum)

# Refactored code

def calculate_sums(numbers):

    even_sum = sum(x for x in numbers if x % 2 == 0)

    odd_sum = sum(x for x in numbers if x % 2 != 0)

    return even_sum, odd_sum

even_sum, odd_sum = calculate_sums(numbers)

print("Sum of even numbers:", even_sum)

print("Sum of odd numbers:", odd_sum)
```

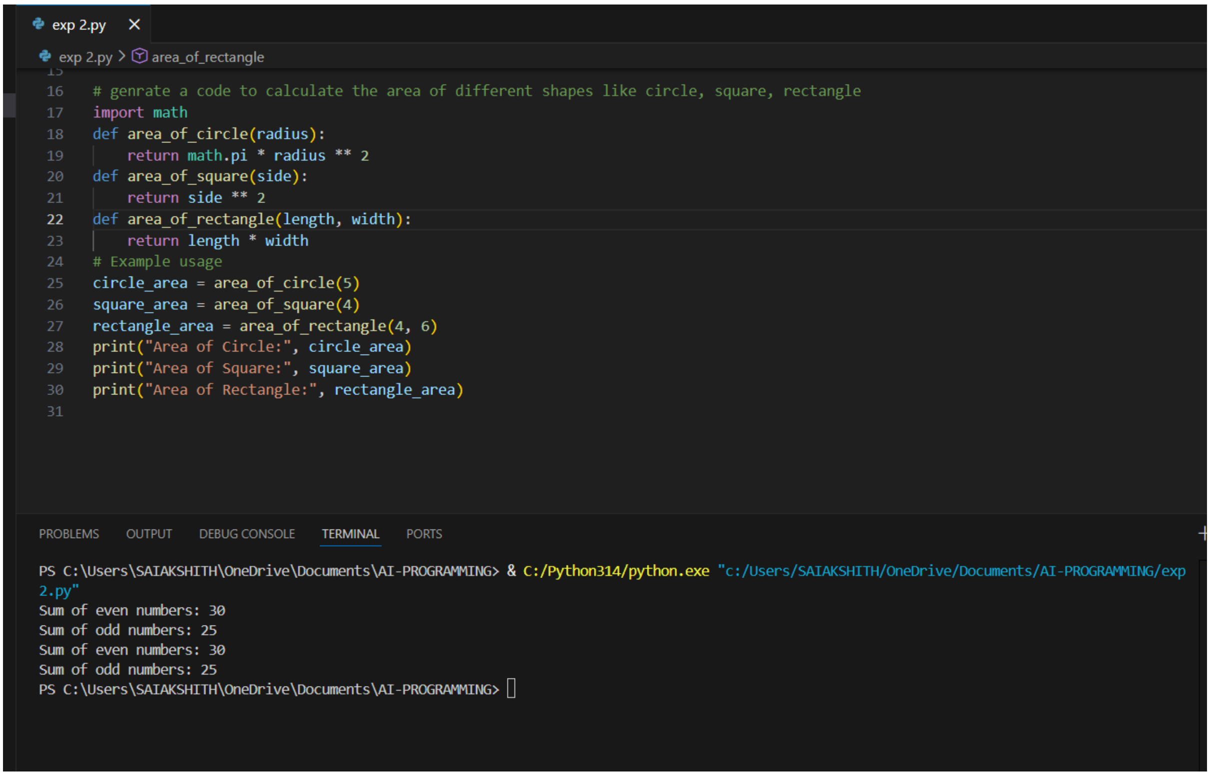


```python
# genrate a program to calculate the sum of odd and even numbers in a list then refactor it using AI.
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
even_sum = sum(x for x in numbers if x % 2 == 0)
odd_sum = sum(x for x in numbers if x % 2 != 0)
print("Sum of even numbers:", even_sum)
print("Sum of odd numbers:", odd_sum)
# Refactored code
def calculate_sums(numbers):
    even_sum = sum(x for x in numbers if x % 2 == 0)
    odd_sum = sum(x for x in numbers if x % 2 != 0)
    return even_sum, odd_sum
even_sum, odd_sum = calculate_sums(numbers)
print("Sum of even numbers:", even_sum)
print("Sum of odd numbers:", odd_sum)
```

```
PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING> & C:/Python314/python.exe "c:/Users/SAIAKSHITH/OneDrive/Documents/AI-PROGRAMMING/exp
2.py"
Sum of even numbers: 30
Sum of odd numbers: 25
Sum of even numbers: 30
Sum of odd numbers: 25
PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING> 
```

Task 2: Area Calculation Explanation

```python
# genrate a code to calculate the area of different shapes like circle, square, rectangle

import math

def area_of_circle(radius):

    return math.pi * radius ** 2

def area_of_square(side):

    return side ** 2

def area_of_rectangle(length, width):

    return length * width

# Example usage

circle_area = area_of_circle(5)

square_area = area_of_square(4)

rectangle_area = area_of_rectangle(4, 6)

print("Area of Circle:", circle_area)

print("Area of Square:", square_area)

print("Area of Rectangle:", rectangle_area)
```



```python
# genrate a code to calculate the area of different shapes like circle, square, rectangle
import math
def area_of_circle(radius):
    return math.pi * radius ** 2
def area_of_square(side):
    return side ** 2
def area_of_rectangle(length, width):
    return length * width
# Example usage
circle_area = area_of_circle(5)
square_area = area_of_square(4)
rectangle_area = area_of_rectangle(4, 6)
print("Area of Circle:", circle_area)
print("Area of Square:", square_area)
print("Area of Rectangle:", rectangle_area)
```

```
PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING> & C:/Python314/python.exe "c:/Users/SAIAKSHITH/OneDrive/Documents/AI-PROGRAMMING/exp 2.py"
Sum of even numbers: 30
Sum of odd numbers: 25
Sum of even numbers: 30
Sum of odd numbers: 25
PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING> 
```

Task 4: Tool Comparison Reflection

❖ Scenario:

You must recommend an AI coding tool.

❖ Task:

Based on your work in this topic, compare Gemini, Copilot, and Cursor AI for usability and code quality.

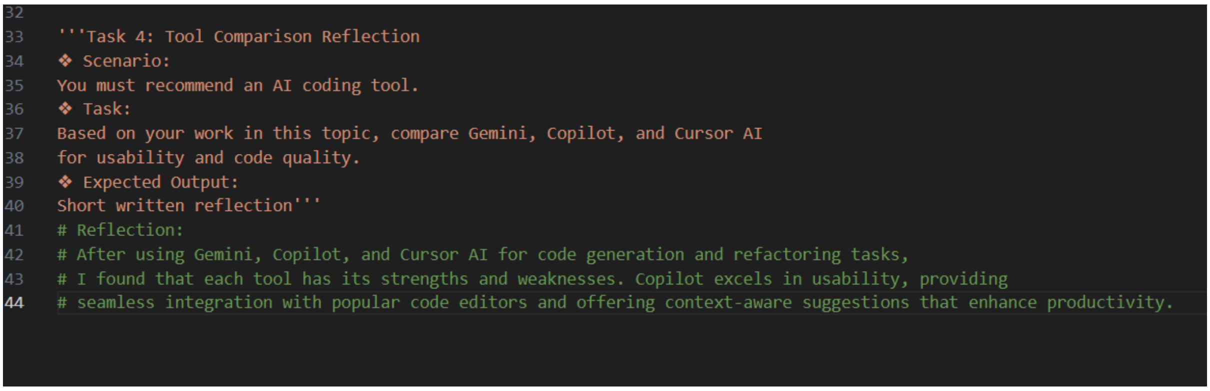
❖ Expected Output:

Short written reflection

PROMT :

compare Gemini, Copilot, and Cursor AI for usability and code quality.

CODE:



```

'''Task 4: Tool Comparison Reflection
❖ Scenario:
You must recommend an AI coding tool.
❖ Task:
Based on your work in this topic, compare Gemini, Copilot, and Cursor AI
for usability and code quality.
❖ Expected Output:
Short written reflection'''
# Reflection:
# After using Gemini, Copilot, and Cursor AI for code generation and refactoring tasks,
# I found that each tool has its strengths and weaknesses. Copilot excels in usability, providing
# seamless integration with popular code editors and offering context-aware suggestions that enhance productivity.
```