Assignment-13

Task-1:

Prompt:

Refactor the following legacy code to use a more Pythonic approach, such as list comprehensions or helper functions. Ensure the output remains the same.

**numbers = [1, 2, 3, 4, 5]**

**squares = []**

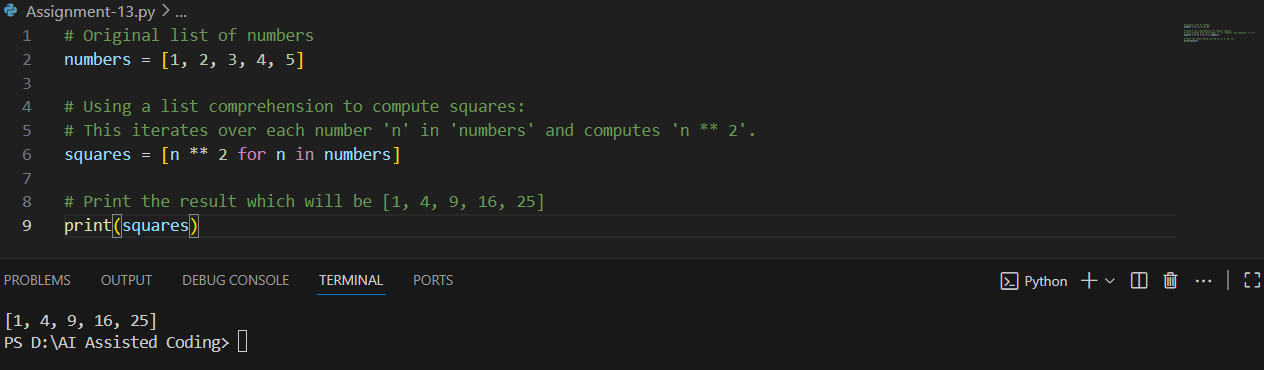
**for n in numbers:**

**squares.append(n \*\* 2)**

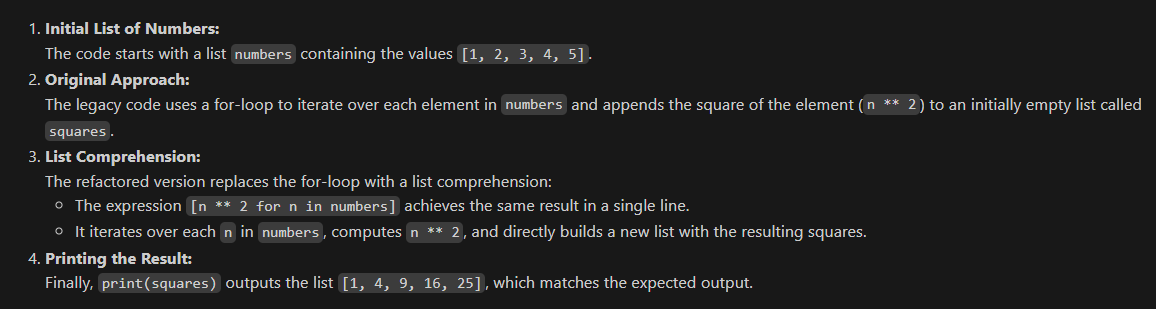
**print(squares)**

**Expected Output:  
[1, 4, 9, 16, 25]**

Code and Output:



Explanation:



Task 2:

Prompt:

Refactor the following legacy code to simplify string concatenation using " ".join() for better efficiency and readability. Keep the output unchanged

**words = ["AI", "helps", "in", "refactoring", "code"]**

**sentence = ""**

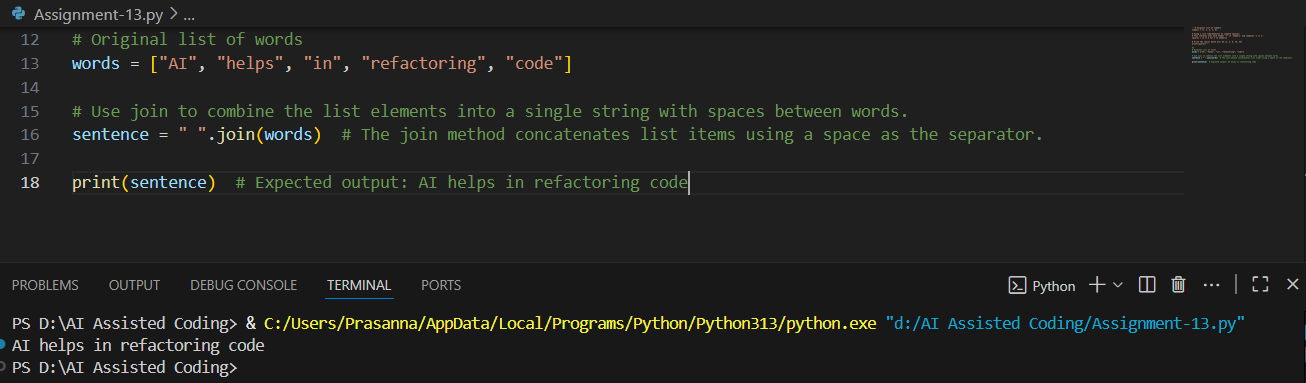
**for word in words:**

**sentence += word + " "**

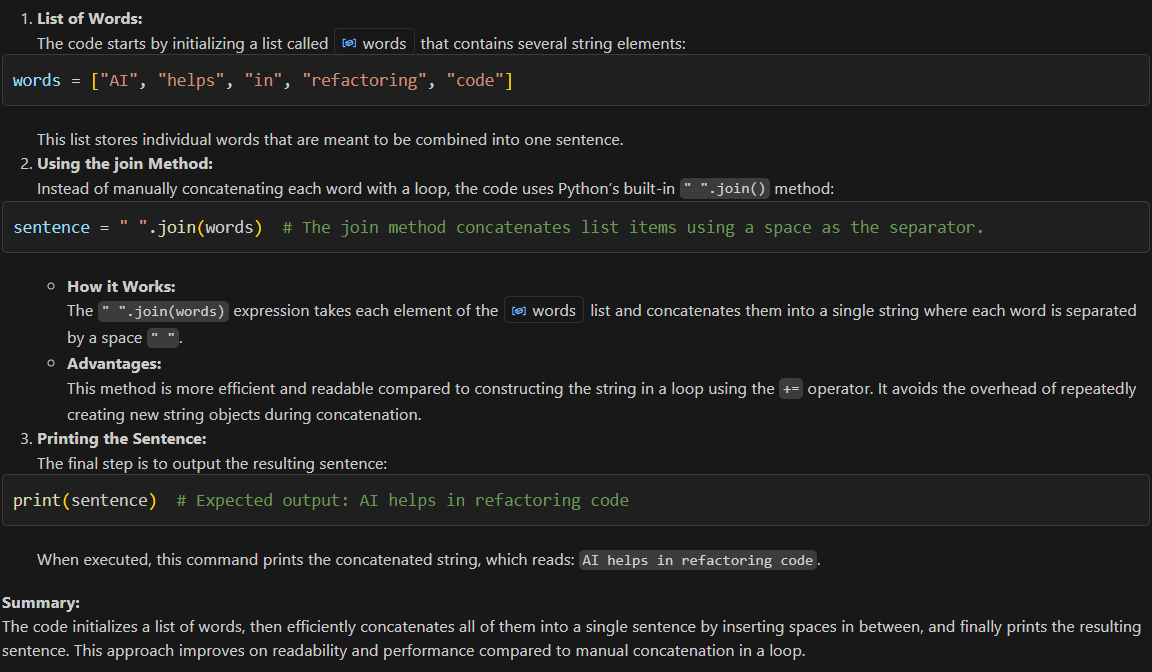
**print(sentence.strip())**

**ExpectedOutput:  
AI helps in refactoring code**

Code and Output:



Explanation:



**Task 3**:

Prompt:

Refactor the following legacy code to use a safer and more Pythonic dictionary access method like .get() to handle missing keys gracefully. Ensure the output remains the same.

**student\_scores = {"Alice": 85, "Bob": 90}**

**if "Charlie" in student\_scores:**

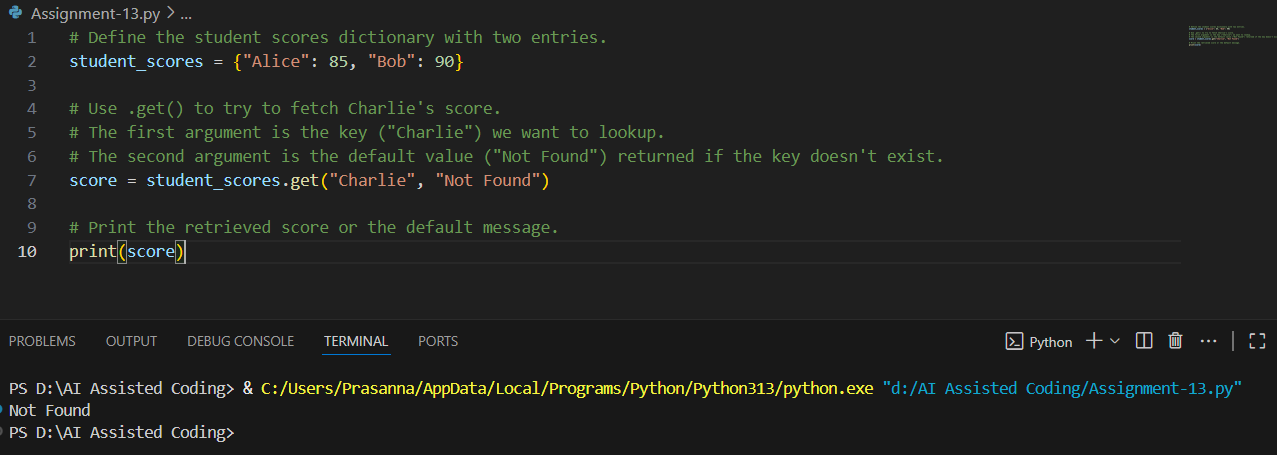
**print(student\_scores["Charlie"])**

**else:**

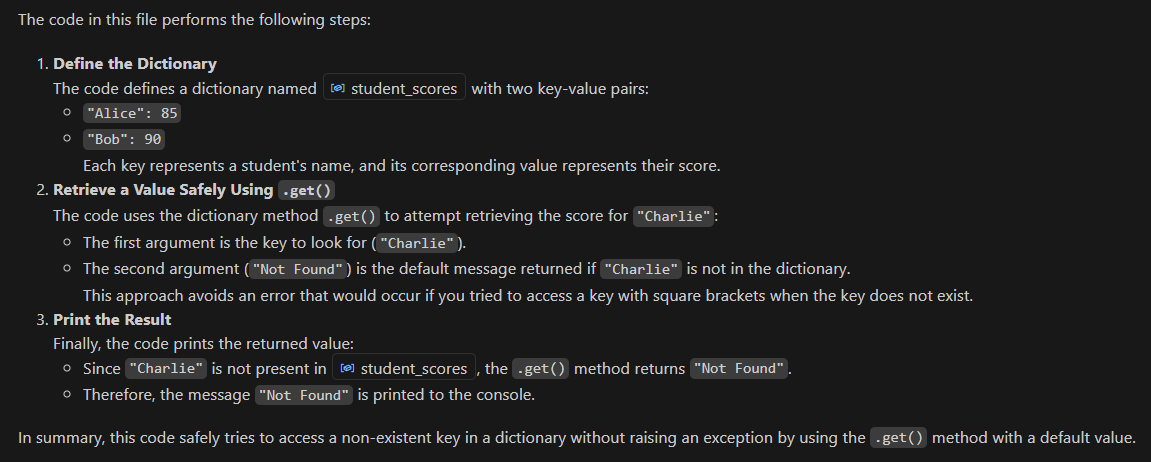
**print("Not Found")**

**Expected Output:  
Not Found**

Code and Output:



Explanation:



Task 4:

Prompt:

Refactor the following legacy code to replace repetitive if-elif blocks with a dictionary-based approach for cleaner and more scalable logic. Ensure the output remains the same.

**operation = "multiply"**

**a, b = 5, 3**

**if operation == "add":**

**result = a + b**

**elif operation == "subtract":**

**result = a - b**

**elif operation == "multiply":**

**result = a \* b**

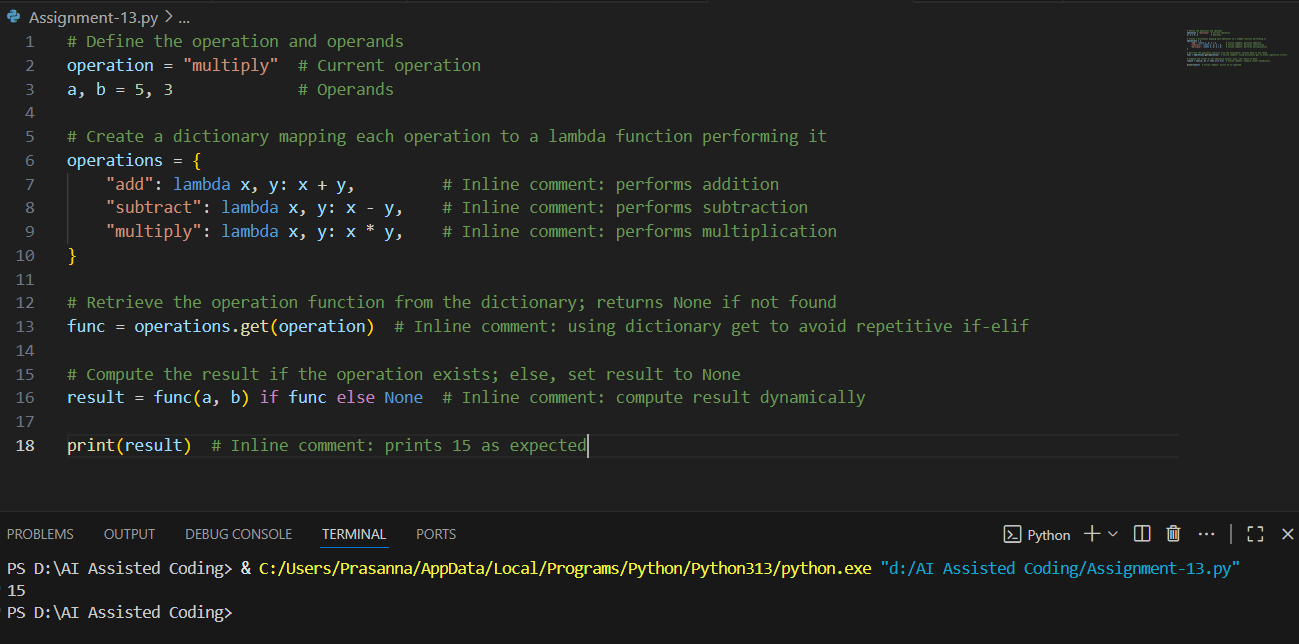
**else:**

**result = None**

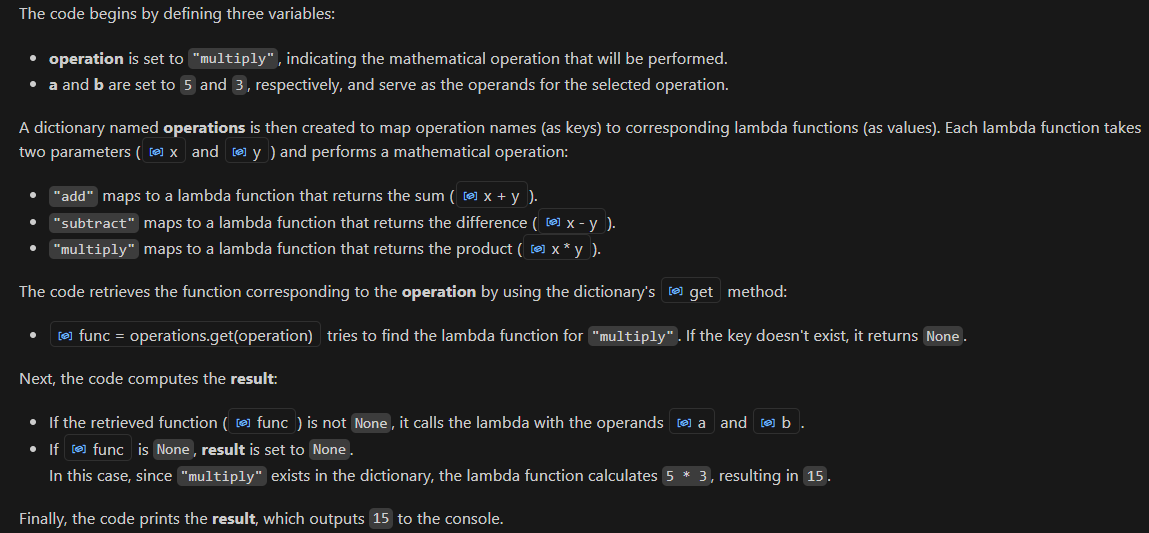
**print(result)**

**Expected Output:  
15**

Code & Output:



Explanation:



Task 5:

Prompt:

Refactor the following legacy code to simplify the search logic using Python’s in keyword or other efficient techniques. Ensure the output remains the same

**items = [10, 20, 30, 40, 50]**

**found = False**

**for i in items:**

**if i == 30:**

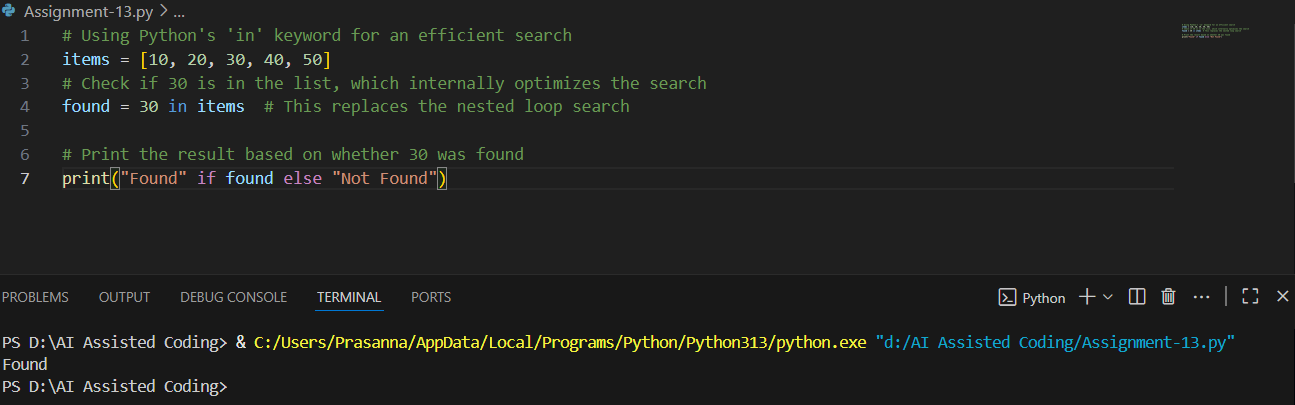
**found = True**

**break**

**print("Found" if found else "Not Found")**

**Expected Output:  
Found**

Code & Output:



Explanation:

