Name: Hamza Badshah

Intern ID: TN/IN01/PY/001

Email ID : hamzabadshah2592@gmail.com

Internship Domain : Python Development

Task Week : 05

Instructor Name : Mr.Hassan Ali

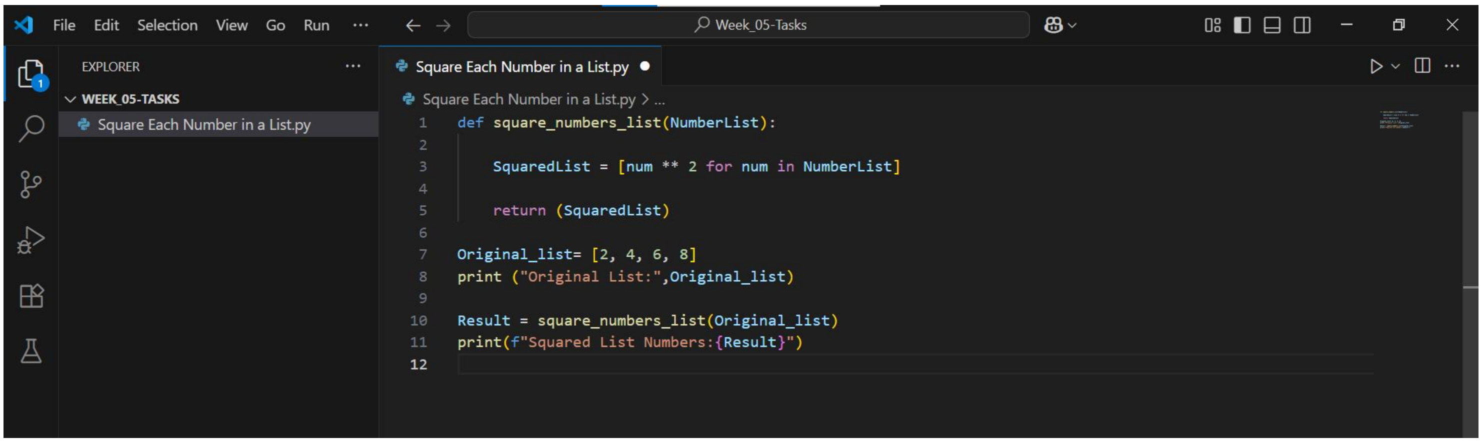
Task 1 :

Create a function `square_numbers` that takes a list of numbers and returns a list of their squares.

Steps to Perform this Task:

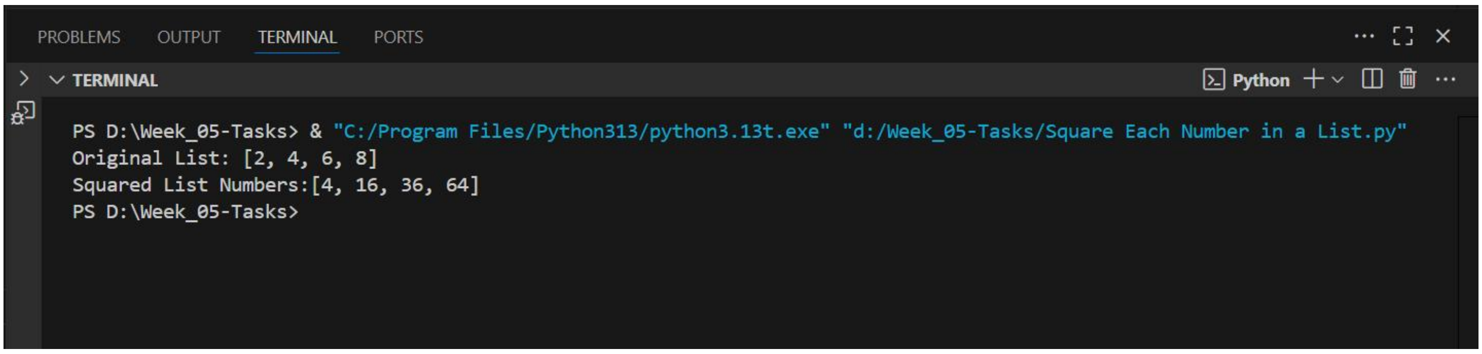
- Defined a function called square_numbers(numbers).
- Took space-separated input from the user and converted it to a list.
- Called the function with the user's list as input.
- Stored the result in a variable squared_list.
- Printed the squared numbers clearly.

Code Snippet:


File Edit Selection View Go Run ···
Week_05-Tasks
EXPLORER
WEEK_05-TASKS
Square Each Number in a List.py
Square Each Number in a List.py > ...

```python
def square_numbers_list(NumberList):

    SquaredList = [num ** 2 for num in NumberList]

    return (SquaredList)

Original_list= [2, 4, 6, 8]
print ("Original List:",Original_list)

Result = square_numbers_list(Original_list)
print(f"Squared List Numbers:{Result}")
```

Output Snippet:



```
PS D:\Week_05-Tasks> & "C:/Program Files/Python313/python3.13t.exe" "d:/Week_05-Tasks/Square Each Number in a List.py"
Original List: [2, 4, 6, 8]
Squared List Numbers:[4, 16, 36, 64]
PS D:\Week_05-Tasks>
```

Learning and Challenges:

1. Learned how to write reusable functions in Python.
2. Practiced converting string input into a list of integers.
3. Faced a small challenge with split() and int() conversion.
4. Fixed it using list comprehension to handle all numbers easily.
5. Improved understanding of how to work with lists and functions together.

Task 02:

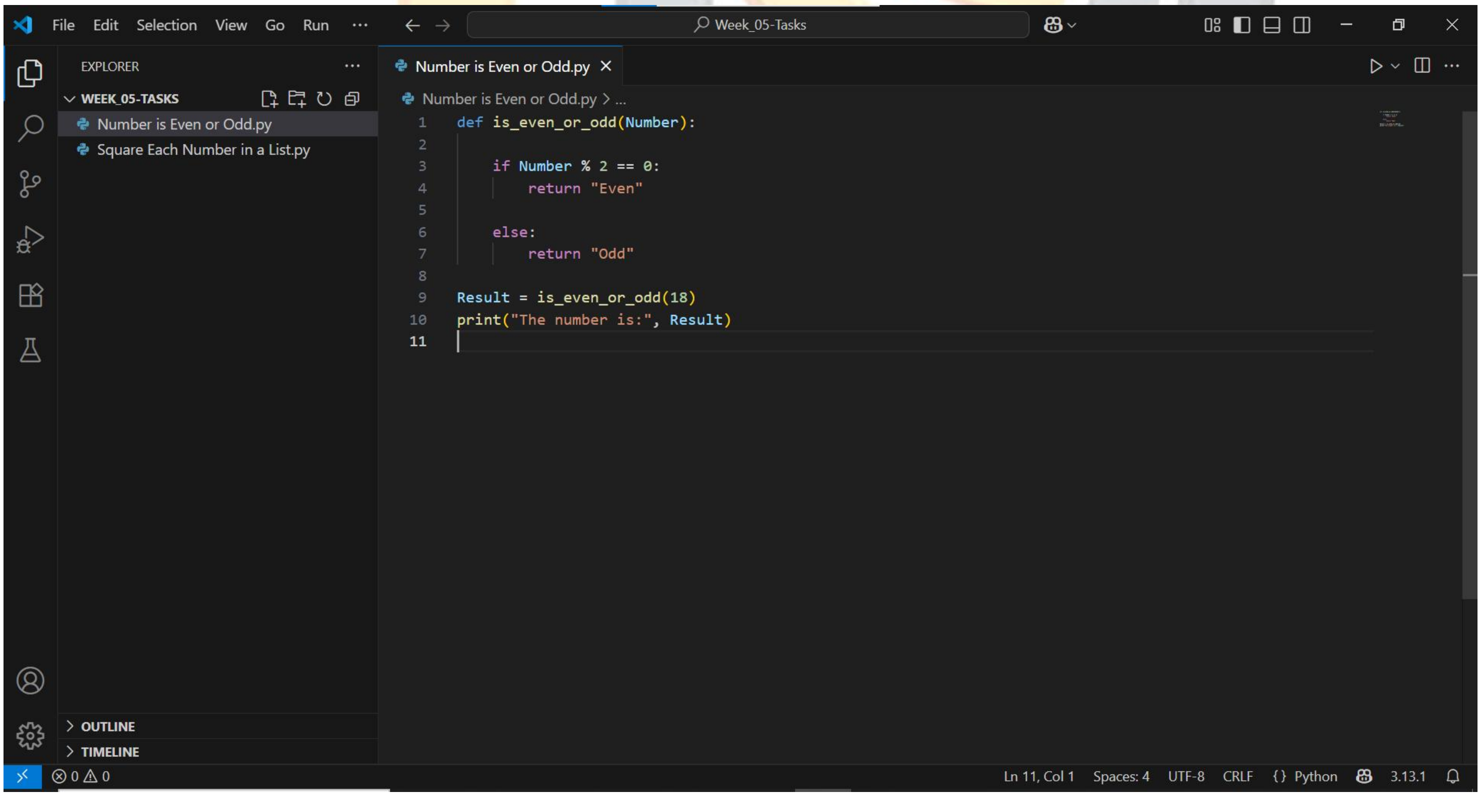
Create a function `is_even_or_odd` that takes a number and returns whether it is even or odd.

Steps to Perform this Task:

- Created a function called is_even_or_odd(number).
- Used the modulus % operator to check if the number is divisible by 2.
- Returned "Even" or "Odd" based on the result.
- Took input from the user using input().

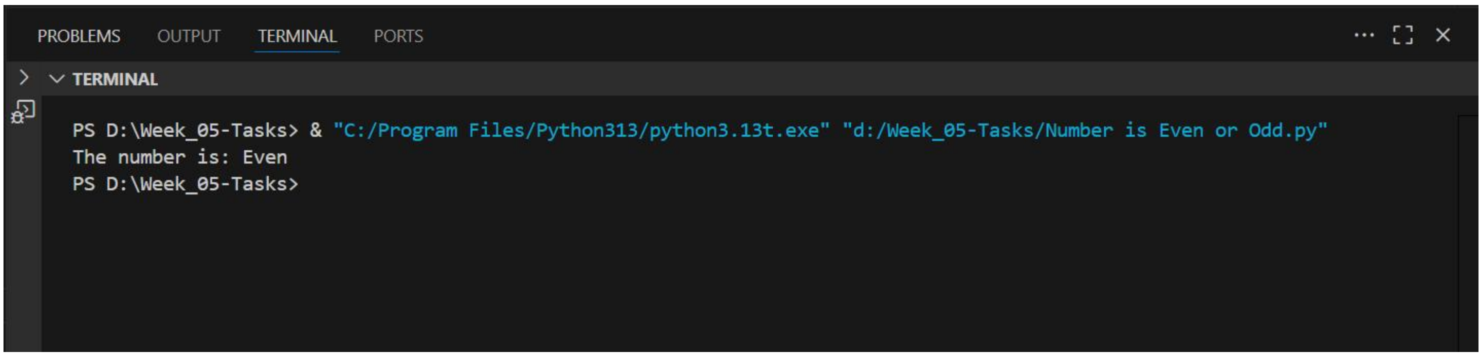
➢ Printed whether the number is even or odd.

Code Snippet:


File Edit Selection View Go Run
Week_05-Tasks
EXPLORER
WEEK_05-TASKS
Number is Even or Odd.py
Square Each Number in a List.py
OUTLINE
TIMELINE
Ln 11, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.13.1

```python
def is_even_or_odd(Number):

    if Number % 2 == 0:
        return "Even"

    else:
        return "Odd"

Result = is_even_or_odd(18)
print("The number is:", Result)

```

Output Snippet:



```
PS D:\Week_05-Tasks> & "C:/Program Files/Python313/python3.13t.exe" "d:/Week_05-Tasks/Number is Even or Odd.py"
The number is: Even
PS D:\Week_05-Tasks>
```

Learning and Challenges:

1) Learned how to check even/odd using %.
2) Practiced writing and calling a function with one input.
3) Faced no issues, logic was easy and clear.
4) Understood how return values work inside a function.
5) Improved my conditional thinking and function writing skills.

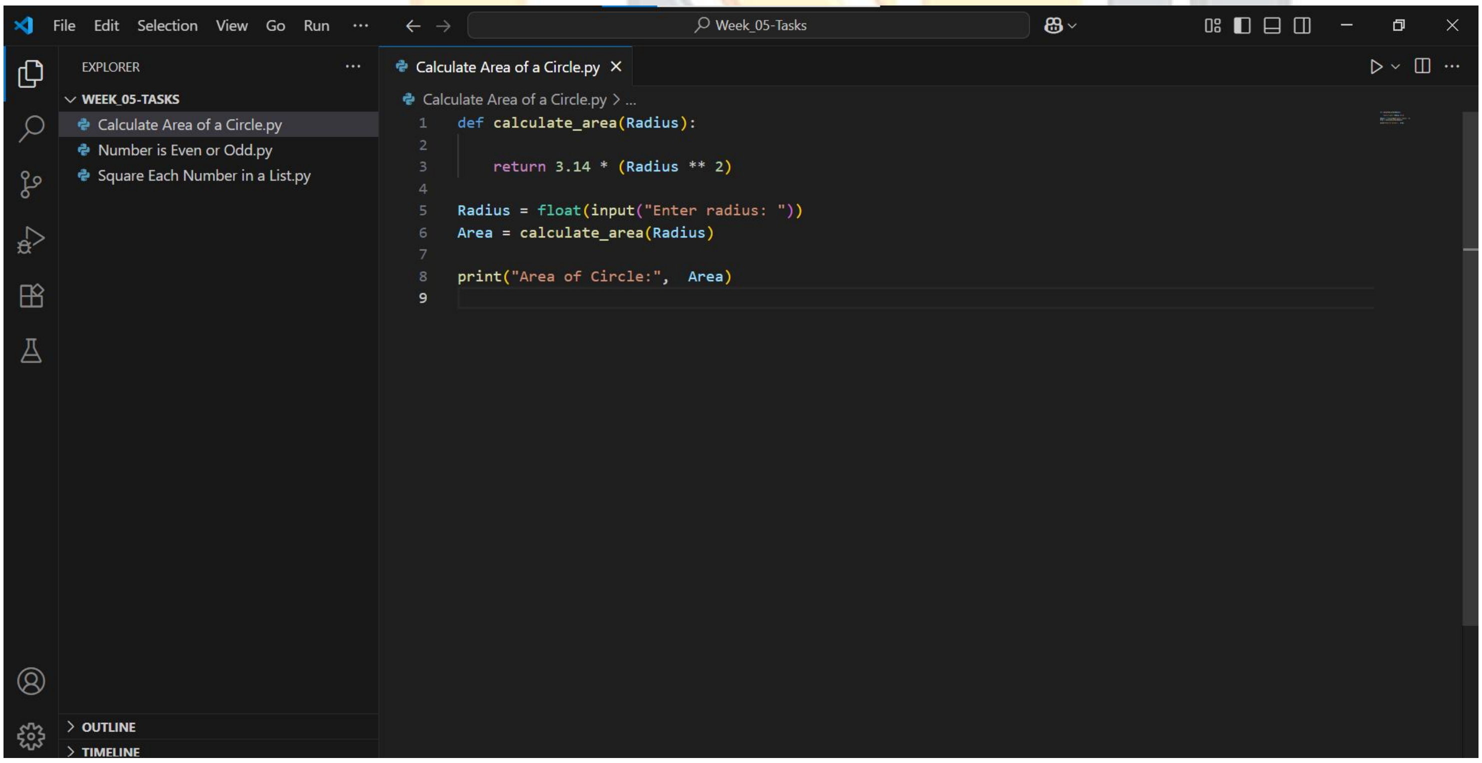
Task 03:

Write a function `calculate_area` that takes radius and returns area of a circle.

Steps to Perform this Task:

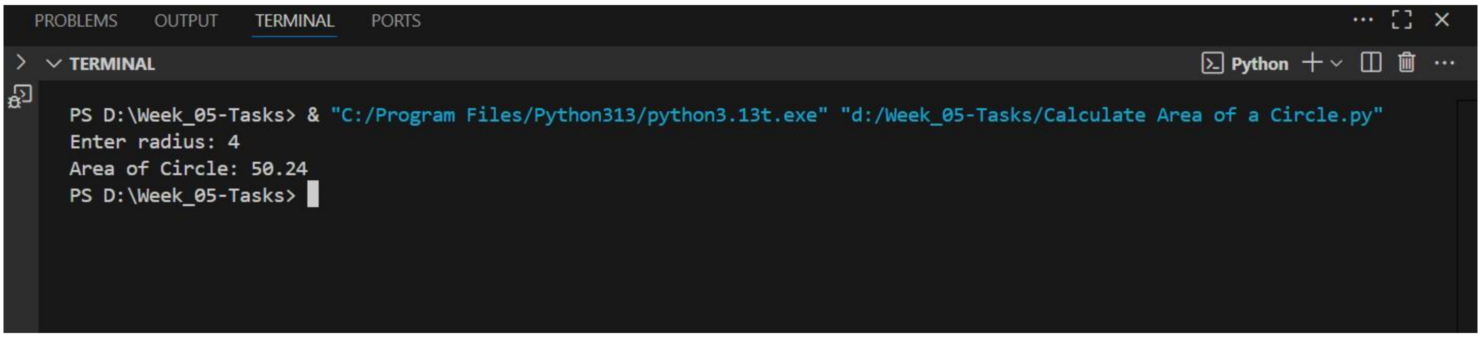
- Imported the math module to use math.pi.
- Created a function called calculate_area(radius).
- Used the formula $3.14 \times radius^2$ to calculate area.
- Took radius input from the user.
- Called the function and printed the result.

Code Snippet:



```python
def calculate_area(Radius):

    return 3.14 * (Radius ** 2)

Radius = float(input("Enter radius: "))
Area = calculate_area(Radius)

print("Area of Circle:",  Area)
```

Output Snippet:



```
PS D:\Week_05-Tasks> & "C:/Program Files/Python313/python3.13t.exe" "d:/Week_05-Tasks/Calculate Area of a Circle.py"
Enter radius: 4
Area of Circle: 50.24
PS D:\Week_05-Tasks> 
```

Learnings and Challenges:

1) Learned how to use the math module in Python.
2) Practiced writing functions with mathematical formulas.
3) Faced no issues, just used correct input and float type.
4) Understood how return values work in real-world formulas.
5) Improved confidence in writing math-based functions.

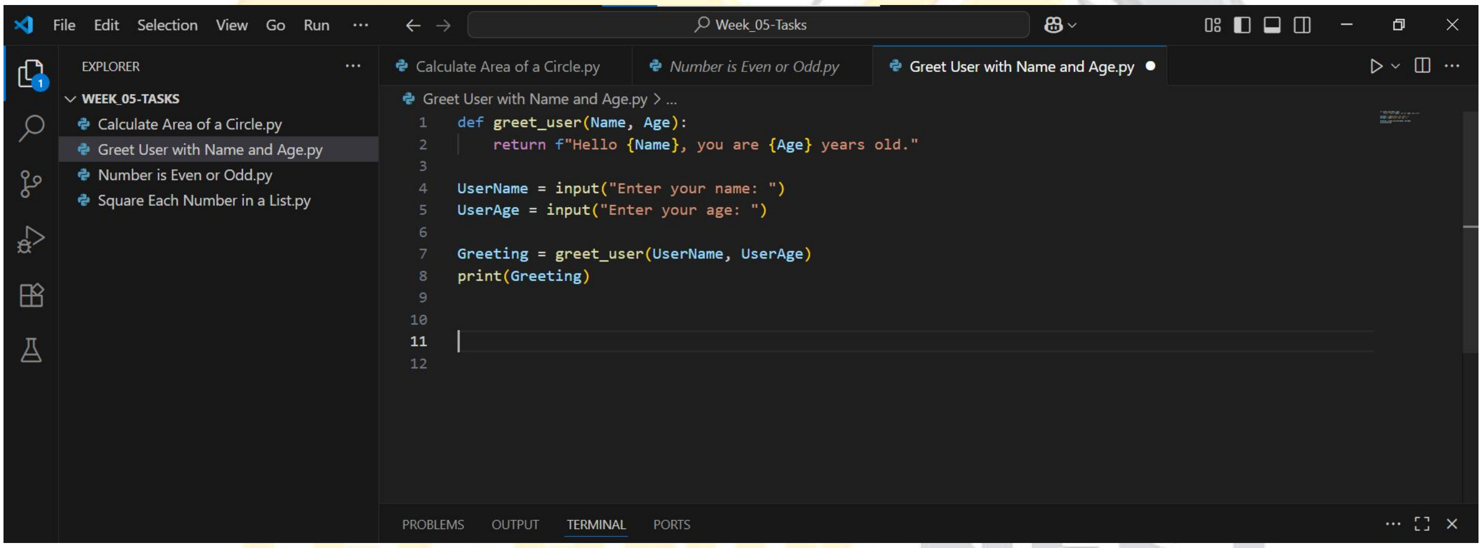
Task 04:

Write a function `greet_user(name, age)` that returns a greeting like: 'Hello Ali, you are 20 years old.'

Steps to Perform this Task:

- Defined a function greet_user(name, age) with two parameters.
- Used an f-string to format the greeting message.
- Took name and age as input from the user.
- Called the function with those inputs.
- Printed the personalized message.

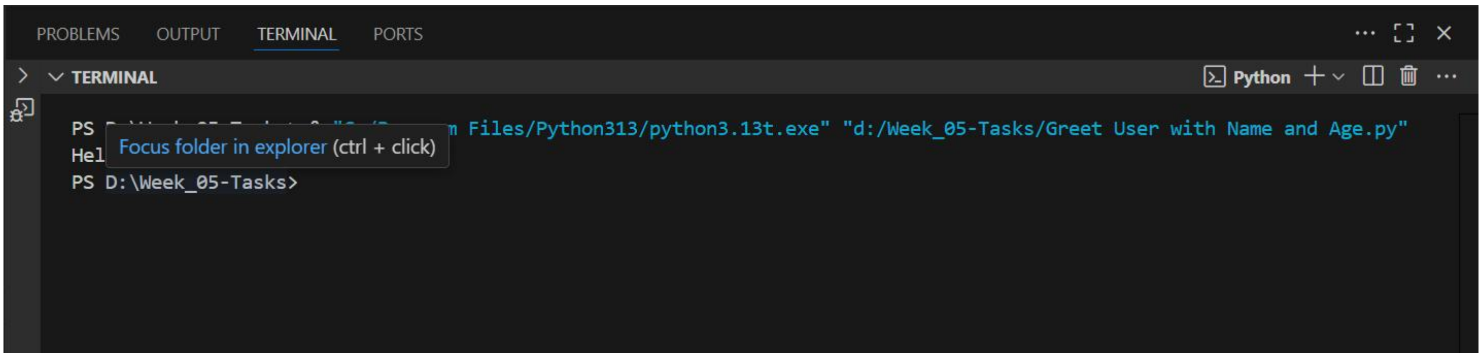
Code Snippet



File Edit Selection View Go Run
Week_05-Tasks
EXPLORER
WEEK_05-TASKS
Calculate Area of a Circle.py
Greet User with Name and Age.py
Number is Even or Odd.py
Square Each Number in a List.py
PROBLEMS OUTPUT TERMINAL PORTS

```python
def greet_user(Name, Age):
    return f"Hello {Name}, you are {Age} years old."

UserName = input("Enter your name: ")
UserAge = input("Enter your age: ")

Greeting = greet_user(UserName, UserAge)
print(Greeting)
```

Output Snippet



PROBLEMS OUTPUT TERMINAL PORTS
TERMINAL
Python
Focus folder in explorer (ctrl + click)

```
PS                  n Files/Python313/python3.13t.exe" "d:/Week_05-Tasks/Greet User with Name and Age.py"
Hel
PS D:\Week_05-Tasks>
```

Learning and Challenges

1. Practiced using f-strings for clean formatting.
2. Faced no issues logic was clear and direct.
3. Understood how to return and use strings from a function.
4. Improved basic input/output handling and formatting.
5. Learned how to use multiple arguments in a function.

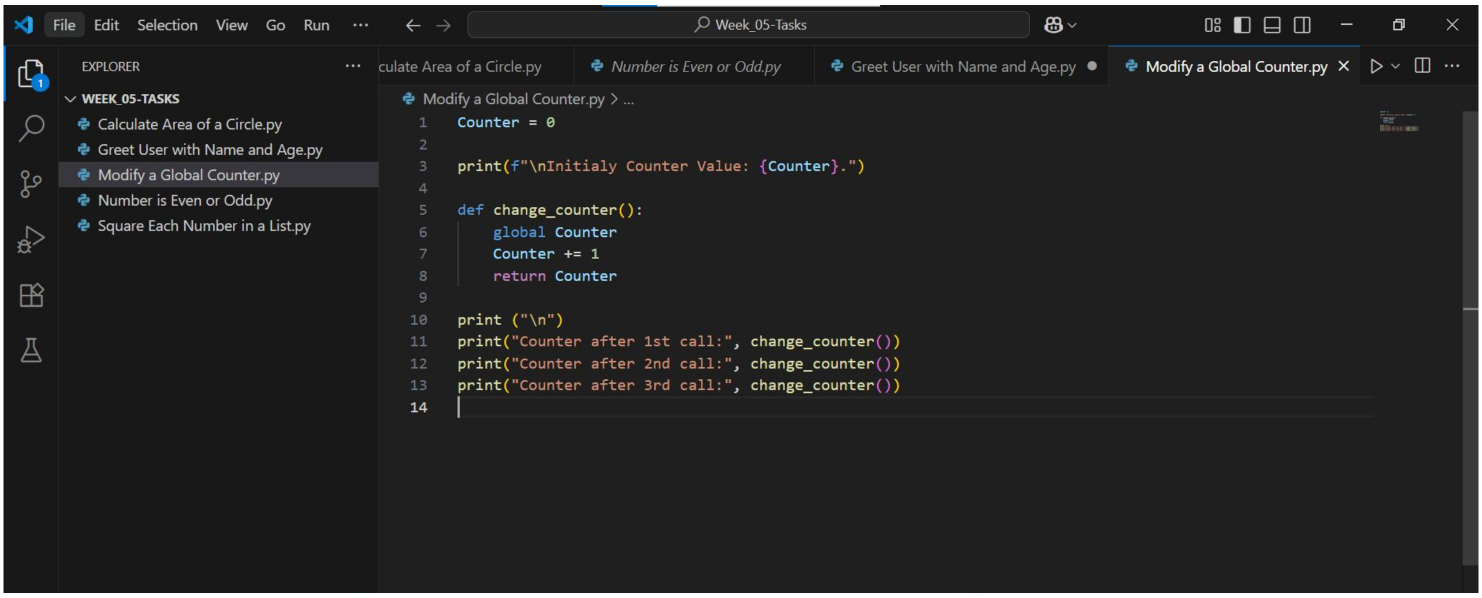
Task 05:

Create a function `change_counter()` that modifies a global counter variable.

Steps to Perform this Task:

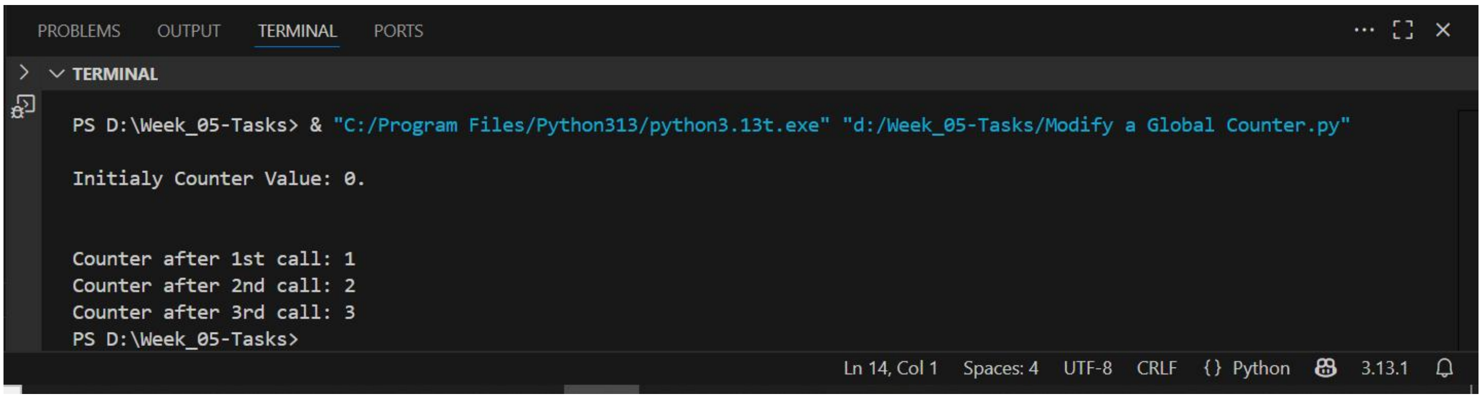
- Declared a global variable counter with initial value 0.
- Created a function change_counter() that uses global keyword.
- Increased the value of counter by 1 inside the function.
- Called the function multiple times to show the change.
- Printed the updated counter value after each call.

Code Snippet:


File Edit Selection View Go Run
Week_05-Tasks
EXPLORER
WEEK_05-TASKS
Calculate Area of a Circle.py
Greet User with Name and Age.py
Modify a Global Counter.py
Number is Even or Odd.py
Square Each Number in a List.py

```python
Counter = 0

print(f"\nInitialy Counter Value: {Counter}.")

def change_counter():
    global Counter
    Counter += 1
    return Counter

print ("\n")
print("Counter after 1st call:", change_counter())
print("Counter after 2nd call:", change_counter())
print("Counter after 3rd call:", change_counter())
```

Output Snippet:


PROBLEMS OUTPUT TERMINAL PORTS
TERMINAL
Ln 14, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.13.1

```
PS D:\Week_05-Tasks> & "C:/Program Files/Python313/python3.13t.exe" "d:/Week_05-Tasks/Modify a Global Counter.py"

Initialy Counter Value: 0.


Counter after 1st call: 1
Counter after 2nd call: 2
Counter after 3rd call: 3
PS D:\Week_05-Tasks>
```

Learning and Challenges

1. Learned how to use and modify global variables in a function.
2. Practiced using the global keyword correctly.
3. Faced no issues syntax is simple and clear.
4. Understood the difference between local and global scope.
5. Gained better control over how variables behave across code.

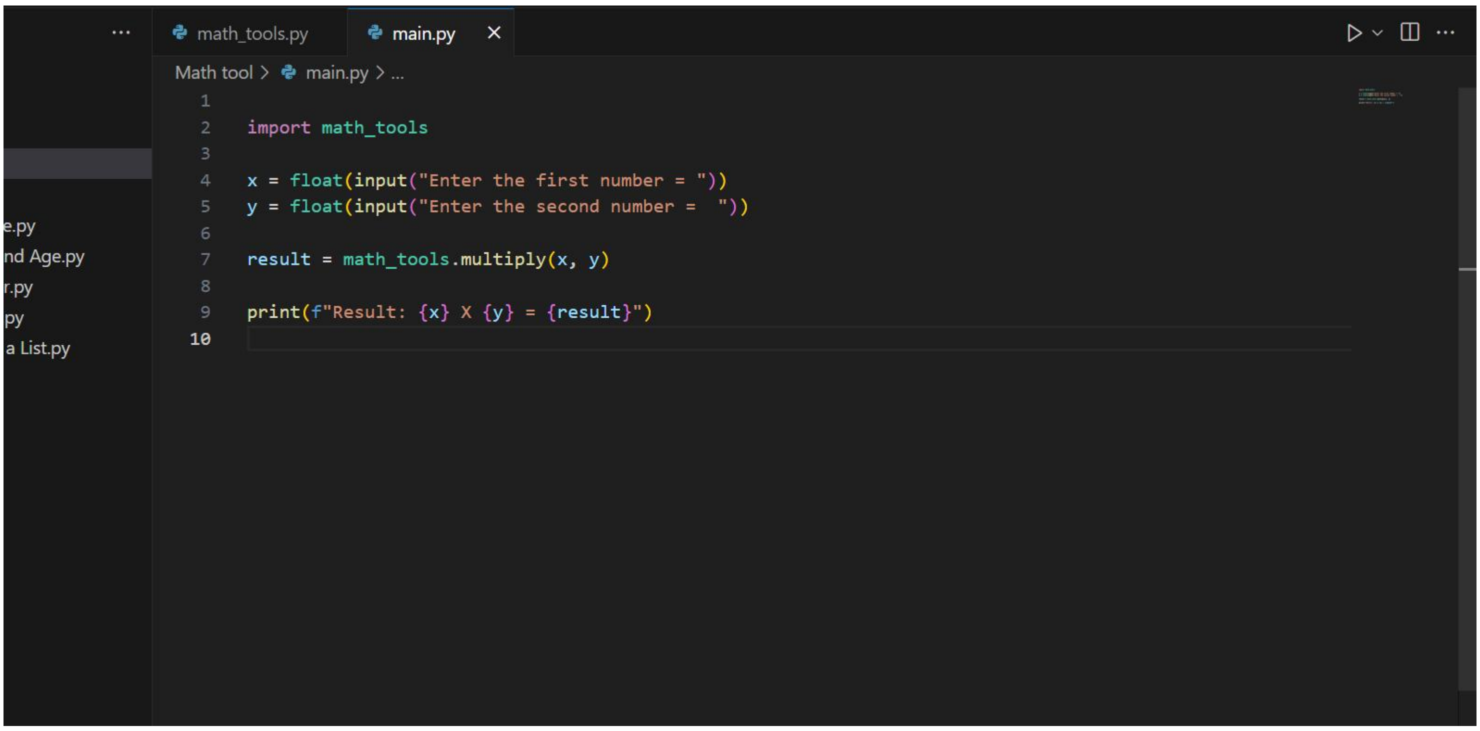
Task 06:

Create a module named `math_tools.py` with a function `multiply(x, y)` and use it in another script.

Steps to Perform this Task:

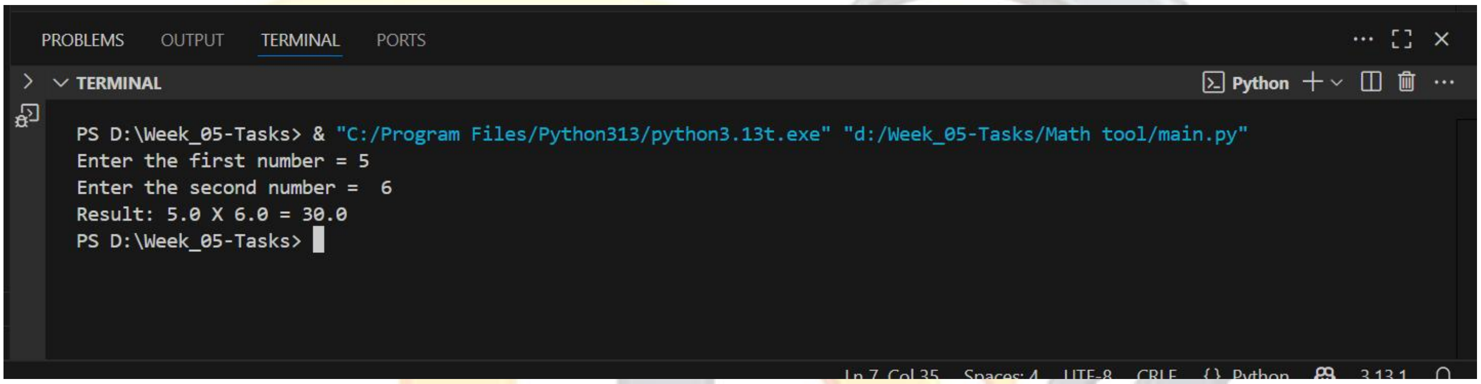
- Created a module file math_tools.py with a function multiply(x, y).
- Wrote a second file main.py to import and use that function.
- Took two numbers from user input using input().
- Called the function and stored the result.
- Printed the final result clearly.

Code Snippet:



```python

import math_tools

x = float(input("Enter the first number = "))
y = float(input("Enter the second number =  "))

result = math_tools.multiply(x, y)

print(f"Result: {x} X {y} = {result}")

```

Output Snippet:



```
PS D:\Week_05-Tasks> & "C:/Program Files/Python313/python3.13t.exe" "d:/Week_05-Tasks/Math tool/main.py"
Enter the first number = 5
Enter the second number =  6
Result: 5.0 X 6.0 = 30.0
PS D:\Week_05-Tasks> 
```

Learning and Challenges

1. Learned how to build and import custom modules in Python.
2. Practiced separating code into reusable files.
3. Faced an import error when files weren't in the same folder.
4. Solved it by keeping both files together.
5. Understood the use of modular code for cleaner projects.