



TECHNIK NEST
INNOVATIVE MINDS, NESTING SUCCESS

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

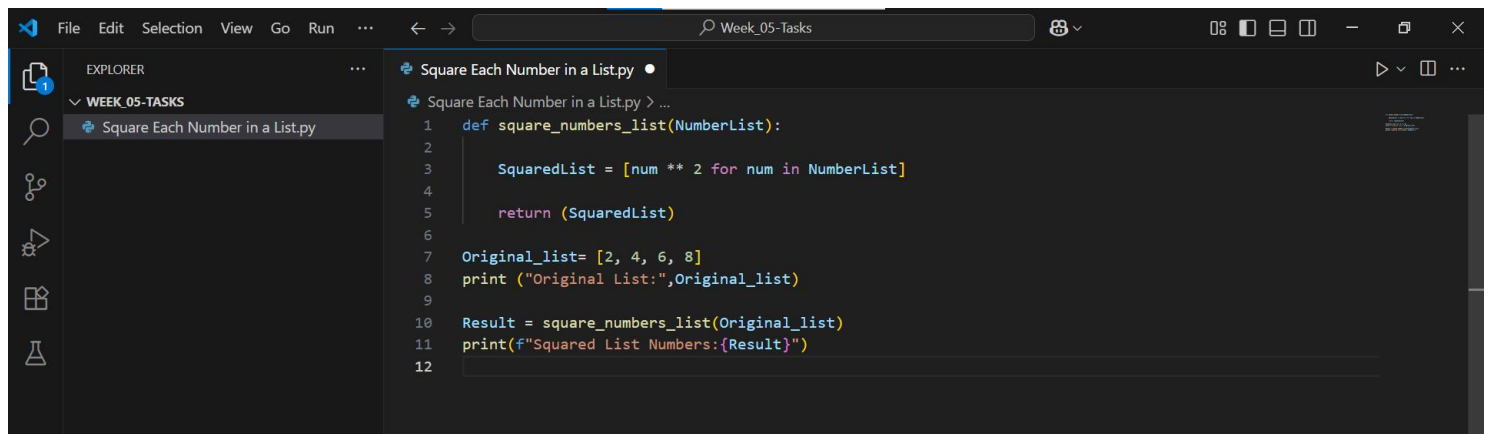
Task 1 :

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

Steps to Perform this Task:

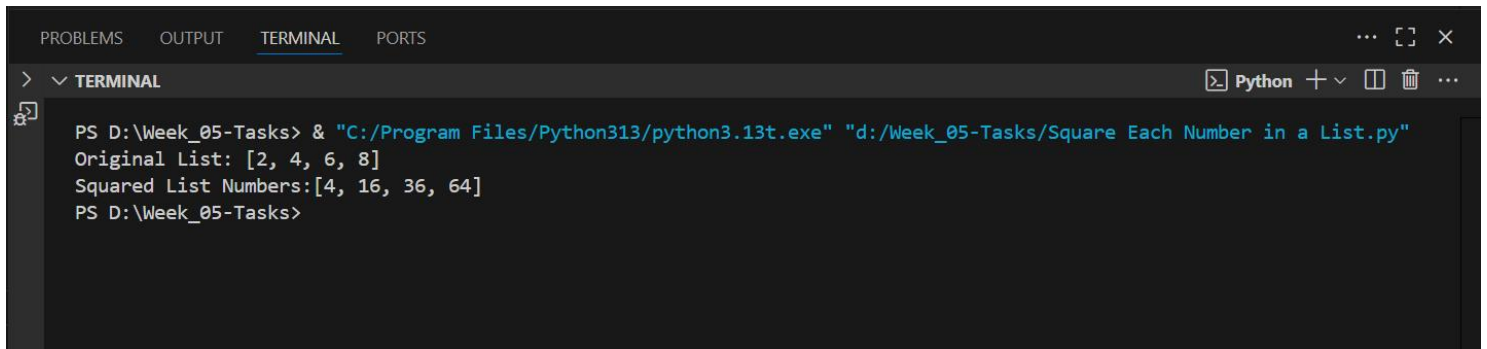
- Defined a function called `square_numbers(numbers)`.
- Took space-separated input from the user and converted it to a list.
- Called the function with the user's list as input.
- Stored the result in a variable `squared_list`.
- Printed the squared numbers clearly.

Code Snippet:

A screenshot of a code editor window. The Explorer pane on the left shows a project named 'WEEK_05-TASKS' with a file 'Square Each Number in a List.py'. The main editor area shows the following Python code:

```
1 def square_numbers_list(NumberList):
2
3     SquaredList = [num ** 2 for num in NumberList]
4
5     return (SquaredList)
6
7 Original_list= [2, 4, 6, 8]
8 print ("Original List:",Original_list)
9
10 Result = square_numbers_list(Original_list)
11 print(f"Squared List Numbers:{Result}")
12
```

Output Snippet:



```
PROBLEMS OUTPUT TERMINAL PORTS
> TERMINAL
Python + - [ ] [ ] [ ]
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:

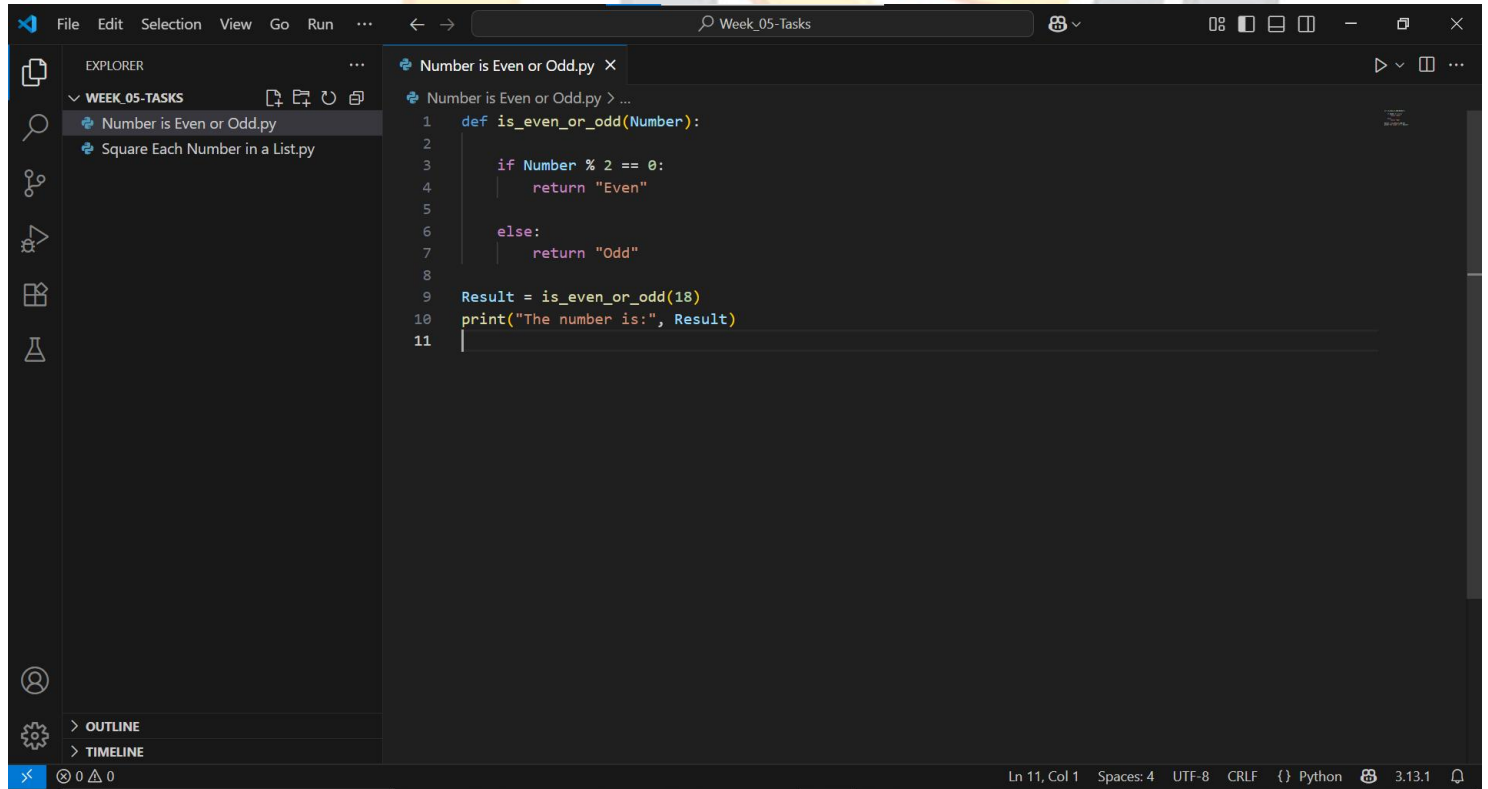
Create a function `'is_even_or_odd'` that takes a number and returns whether it is even or odd.

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:

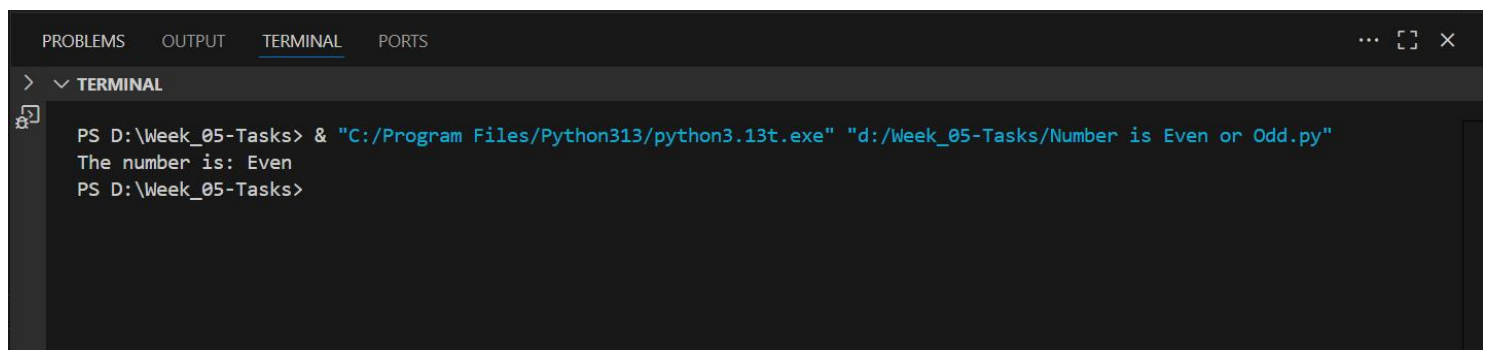


The screenshot shows a code editor with a dark theme. The Explorer panel on the left shows a project named 'WEEK_05-TASKS' with two files: 'Number is Even or Odd.py' and 'Square Each Number in a List.py'. The main editor area displays the code for 'Number is Even or Odd.py'. The code defines a function 'is_even_or_odd' that takes a 'Number' as input and returns 'Even' if the number is divisible by 2, and 'Odd' otherwise. It then calls this function with the value 18 and prints the result.

```
1 def is_even_or_odd(Number):  
2  
3     if Number % 2 == 0:  
4         return "Even"  
5  
6     else:  
7         return "Odd"  
8  
9 Result = is_even_or_odd(18)  
10 print("The number is:", Result)  
11
```

The status bar at the bottom indicates the cursor is at line 11, column 1, with 4 spaces, UTF-8 encoding, CRLF line endings, and Python 3.13.1.

Output Snippet:



The screenshot shows a terminal window with the following output:

```
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.

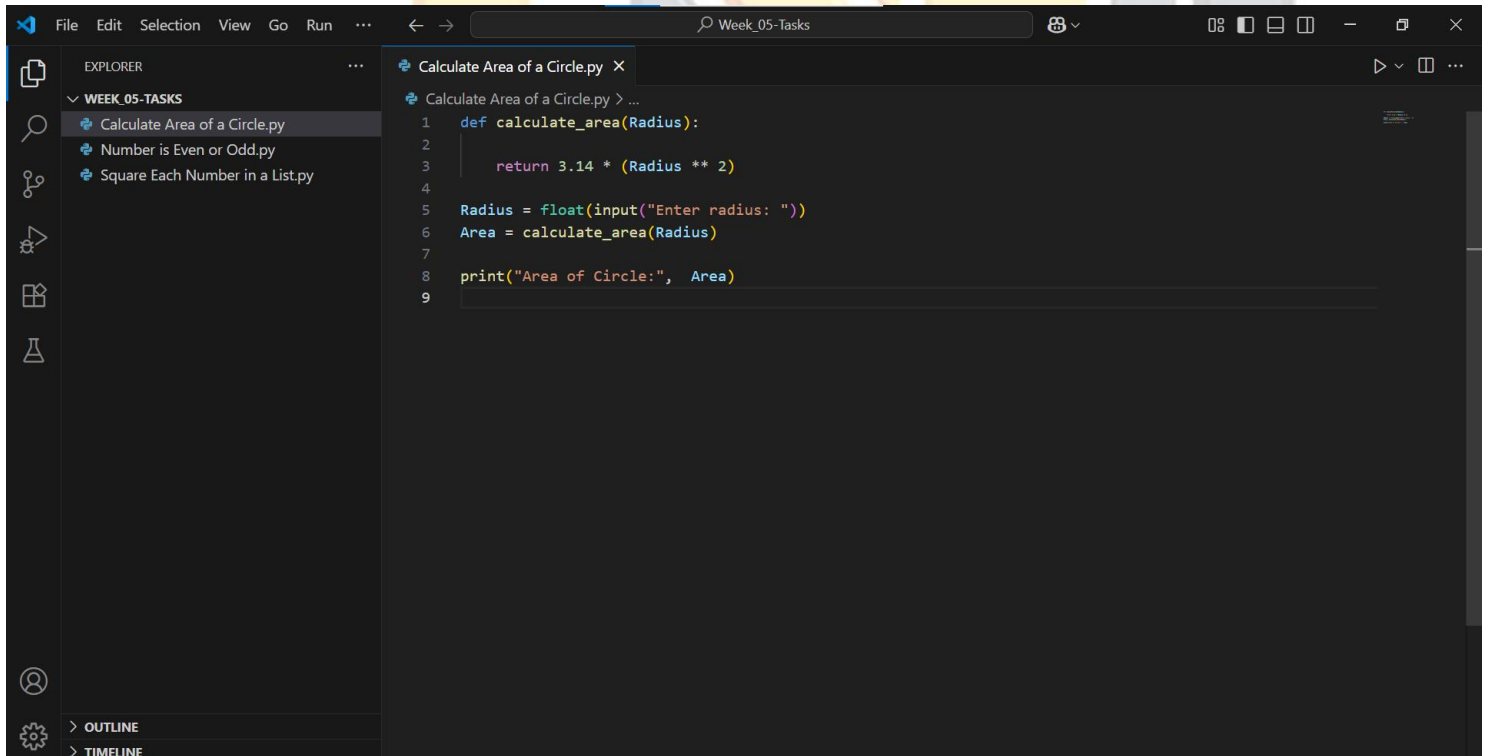
Task 03:

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 $3.14 \times \text{radius}^2$ to calculate area.
- Took radius input from the user.
- Called the function and printed the result.

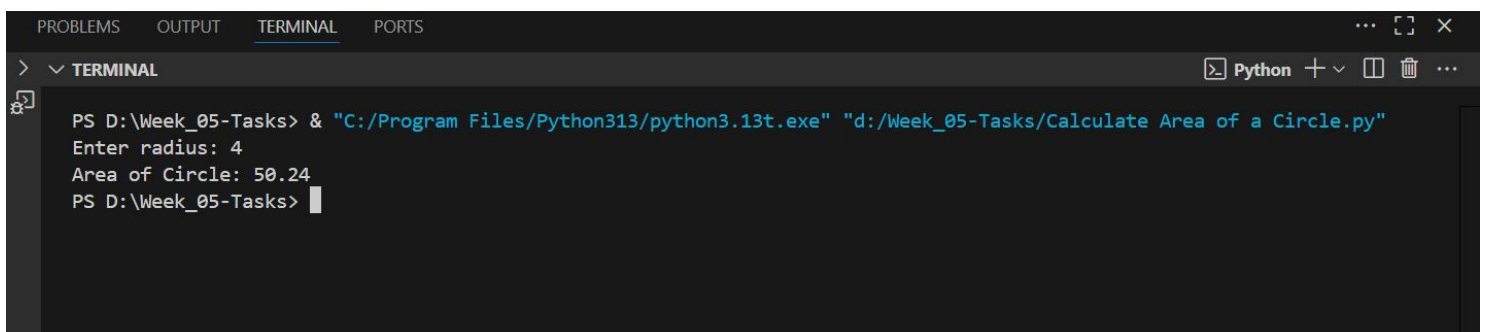
Code Snippet:



The screenshot shows the Visual Studio Code interface with a file explorer on the left and a code editor on the right. The file explorer shows a folder named 'WEEK_05-TASKS' containing three files: 'Calculate Area of a Circle.py', 'Number is Even or Odd.py', and 'Square Each Number in a List.py'. The code editor displays the content of 'Calculate Area of a Circle.py'.

```
1 def calculate_area(Radius):  
2     return 3.14 * (Radius ** 2)  
3  
4  
5 Radius = float(input("Enter radius: "))  
6 Area = calculate_area(Radius)  
7  
8 print("Area of Circle:", Area)  
9
```

Output Snippet:



The screenshot shows a terminal window with the following output:

```
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.

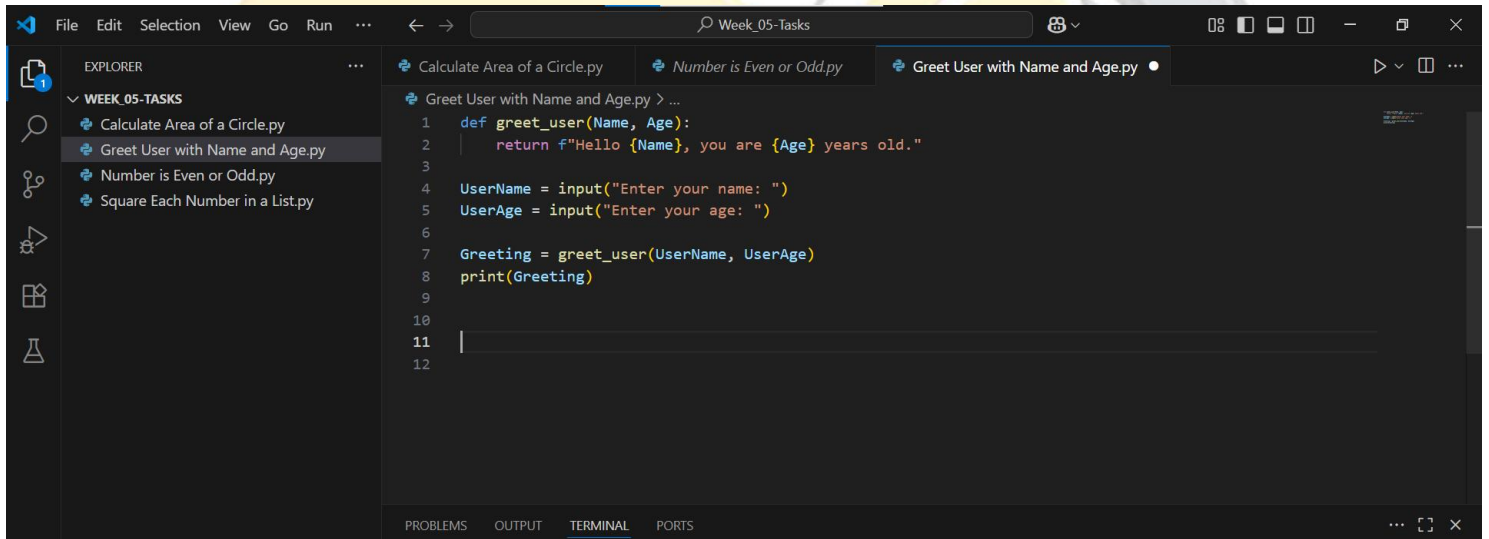
Task 04:

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.

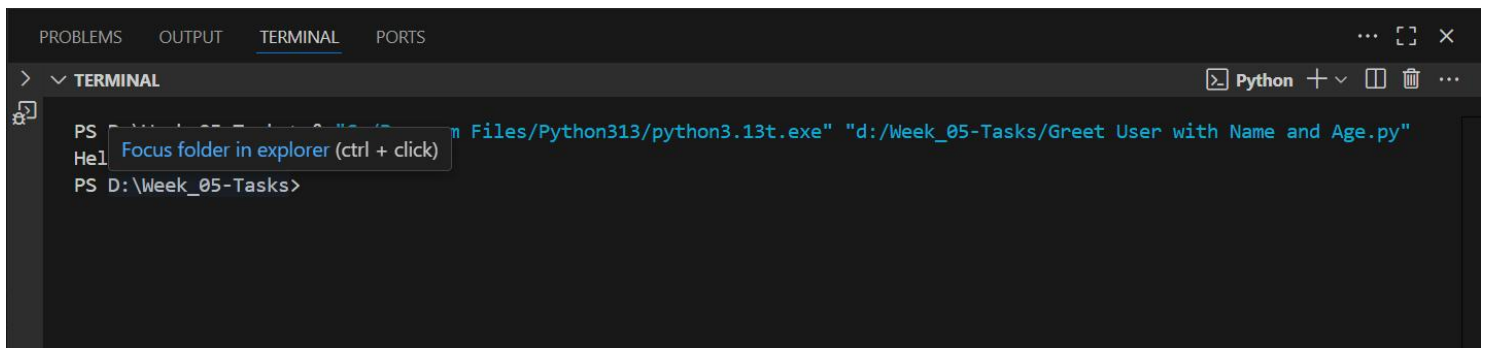
Code Snippet



The screenshot shows the Visual Studio Code editor interface. The Explorer panel on the left displays a folder named 'WEEK_05-TASKS' containing four Python files: 'Calculate Area of a Circle.py', 'Greet User with Name and Age.py', 'Number is Even or Odd.py', and 'Square Each Number in a List.py'. The 'Greet User with Name and Age.py' file is selected and open in the editor. The code in the editor is as follows:

```
1 def greet_user(Name, Age):
2     return f"Hello {Name}, you are {Age} years old."
3
4 UserName = input("Enter your name: ")
5 UserAge = input("Enter your age: ")
6
7 Greeting = greet_user(UserName, UserAge)
8 print(Greeting)
9
10
11
12
```

Output Snippet



The screenshot shows the terminal window in VS Code. The terminal is titled 'TERMINAL' and shows the following output:

```
PS D:\Week_05-Tasks> python Files/Python313/python3.13t.exe "d:/Week_05-Tasks/Greet User with Name and Age.py"
Hello
18
```


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.

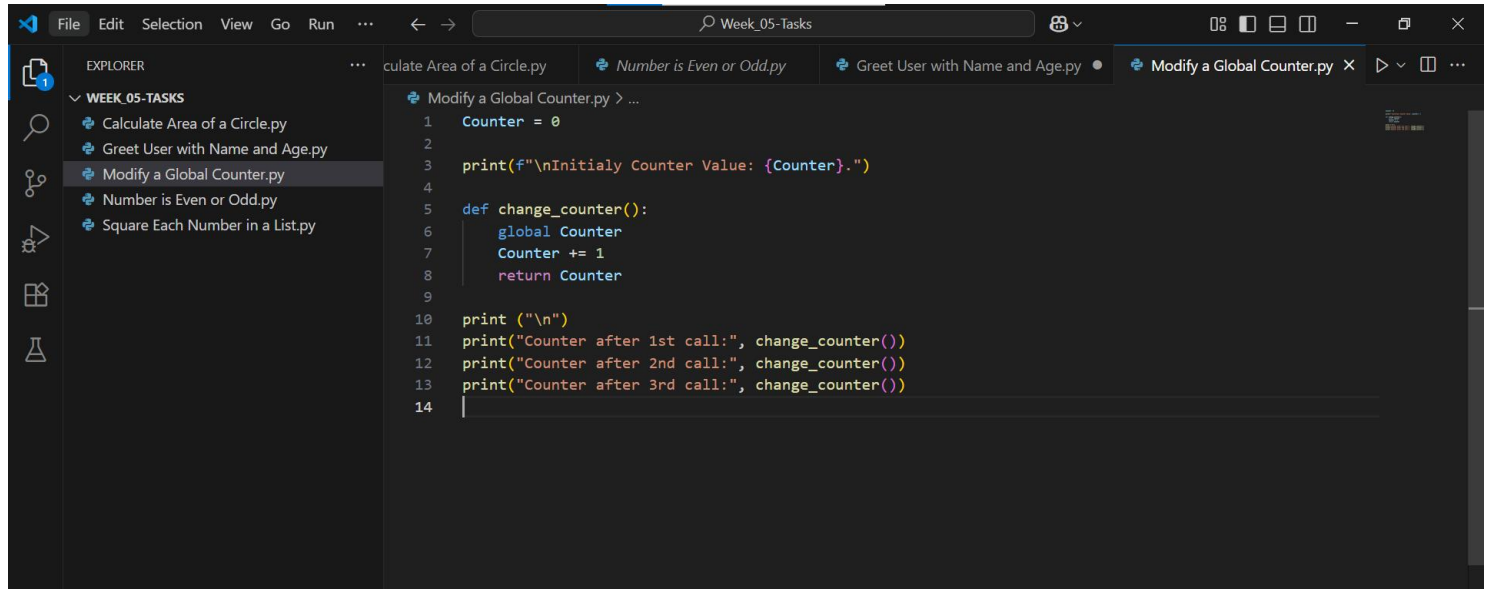
Task 05:

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.

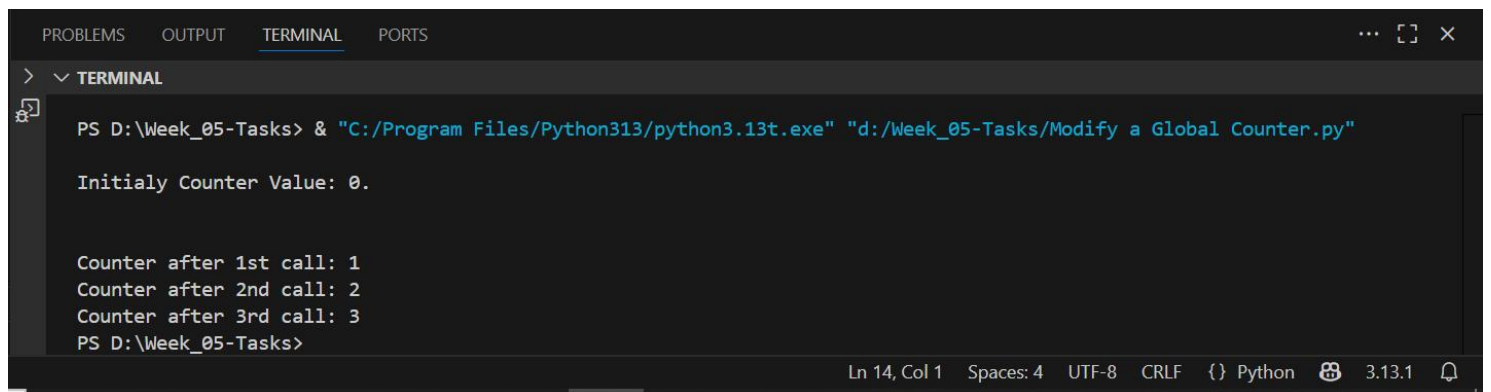
Code Snippet:



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a folder named 'WEEK_05-TASKS' containing several Python files. The code editor shows the file 'Modify a Global Counter.py' with the following code:

```
1 Counter = 0
2
3 print(f"\nInitially Counter Value: {Counter}.")
4
5 def change_counter():
6     global Counter
7     Counter += 1
8     return Counter
9
10 print("\n")
11 print("Counter after 1st call:", change_counter())
12 print("Counter after 2nd call:", change_counter())
13 print("Counter after 3rd call:", change_counter())
14
```

Output Snippet:



The screenshot shows a terminal window with the following output:

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

Initially 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.

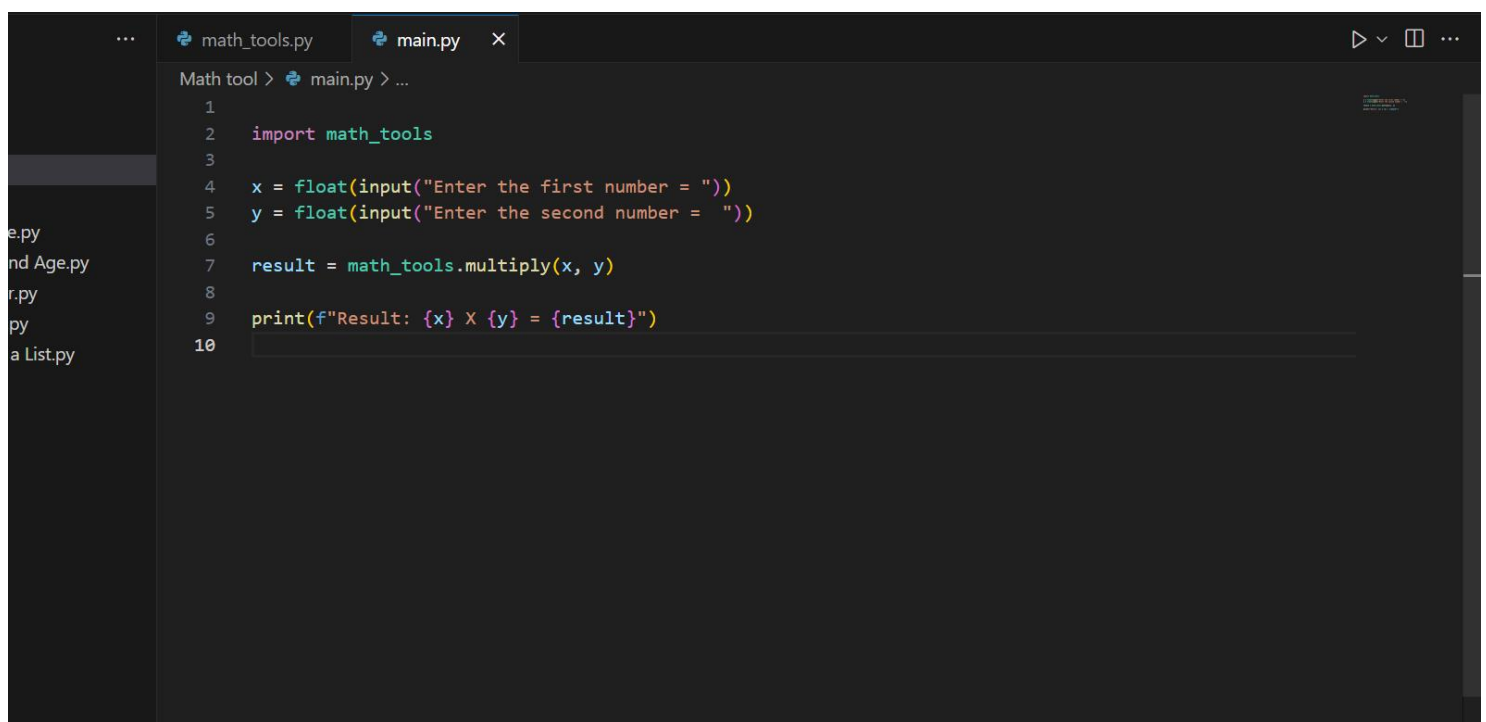
Task 06:

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

Steps to Perform this Task:

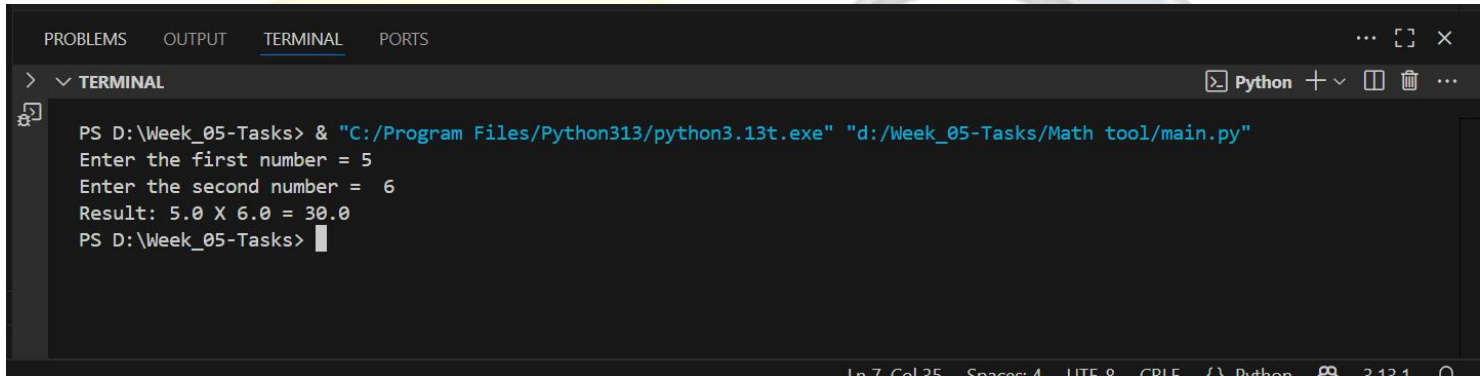
- Created a module file `math_tools.py` with a function `multiply(x, y)`.
- Wrote a second file `main.py` to import and use that function.
- Took two numbers from user input using `input()`.
- Called the function and stored the result.
- Printed the final result clearly.

Code Snippet:



```
Math tool > main.py > ...
1
2  import math_tools
3
4  x = float(input("Enter the first number = "))
5  y = float(input("Enter the second number = "))
6
7  result = math_tools.multiply(x, y)
8
9  print(f"Result: {x} X {y} = {result}")
10
```

Output Snippet:

A screenshot of a terminal window with a dark background. The terminal shows the execution of a Python script. The prompt is 'PS D:\Week_05-Tasks>'. The command entered is '& "C:/Program Files/Python313/python3.13t.exe" "d:/Week_05-Tasks/Math tool/main.py"'. The output shows 'Enter the first number = 5', 'Enter the second number = 6', and 'Result: 5.0 X 6.0 = 30.0'. The prompt returns to 'PS D:\Week_05-Tasks>'. The terminal window has tabs for 'PROBLEMS', 'OUTPUT', 'TERMINAL', and 'PORTS'. The 'TERMINAL' tab is active. The status bar at the bottom shows 'Ln 7, Col 35', 'Spaces: 4', 'UTF-8', 'CRLF', and 'Python 3.13.1'.

```
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.

