LAB NAME : AI ASSISTED CODING

LAB NUMBER :02

ROLL NO :2503A51L16

BRANCH : CSE

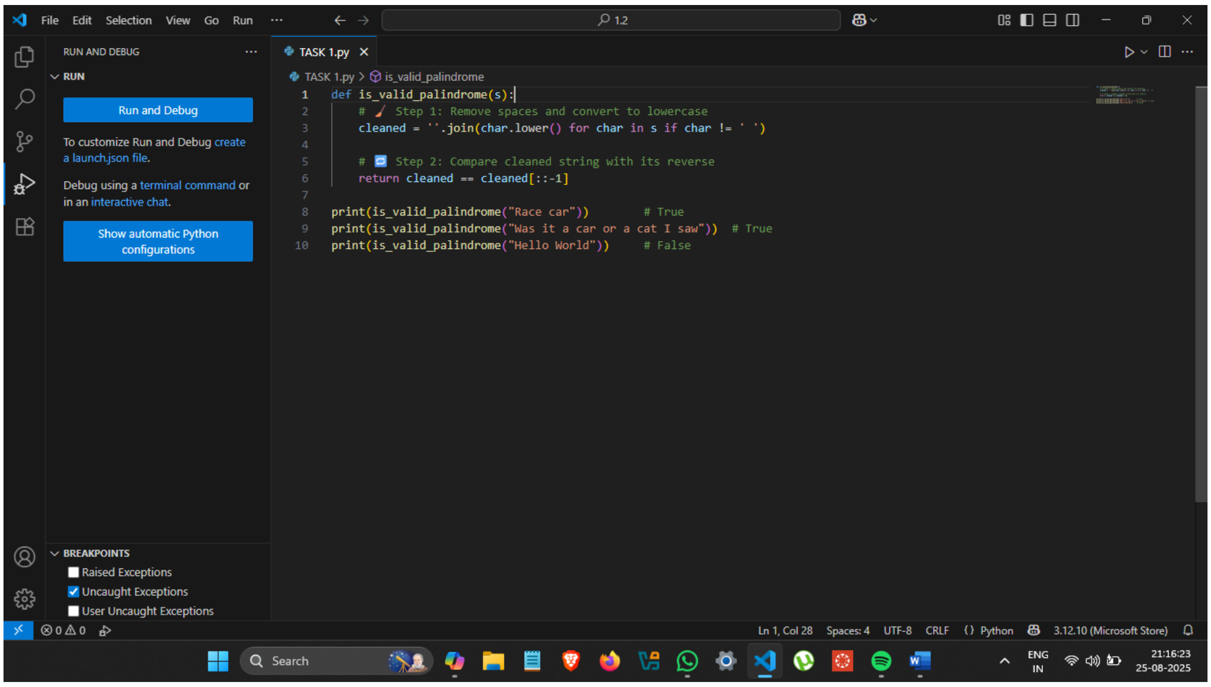
NAME : K.JASHUVA

TASK 1

- **TASK DESCRIPTION:** Write a comment: # Function to check if a string is a valid palindrome (ignoring spaces and case) and allow Copilot to complete it.

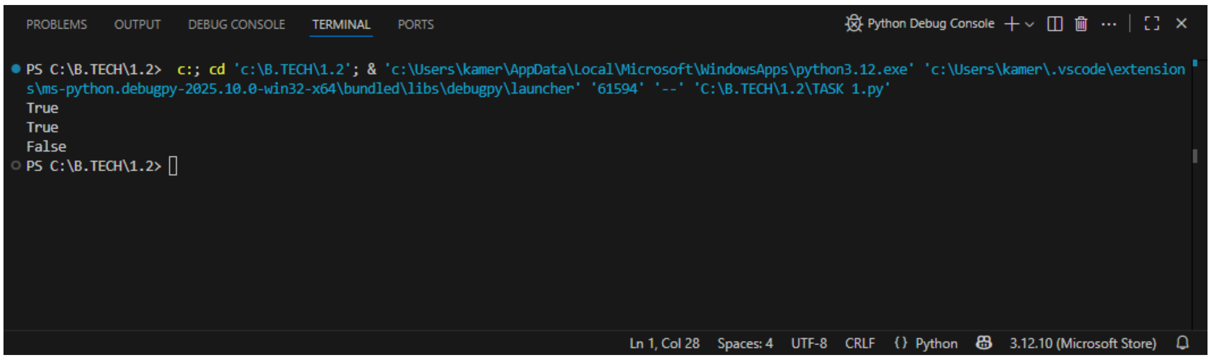
PROMPT: Generate a Python function to check if a string is a valid palindrome, ignoring spaces and case.

CODE:



```python
def is_valid_palindrome(s):
    # 🧹 Step 1: Remove spaces and convert to lowercase
    cleaned = ''.join(char.lower() for char in s if char != ' ')

    # 🔁 Step 2: Compare cleaned string with its reverse
    return cleaned == cleaned[::-1]

print(is_valid_palindrome("Race car"))        # True
print(is_valid_palindrome("Was it a car or a cat I saw"))  # True
print(is_valid_palindrome("Hello World"))     # False
```

OUTPUT:



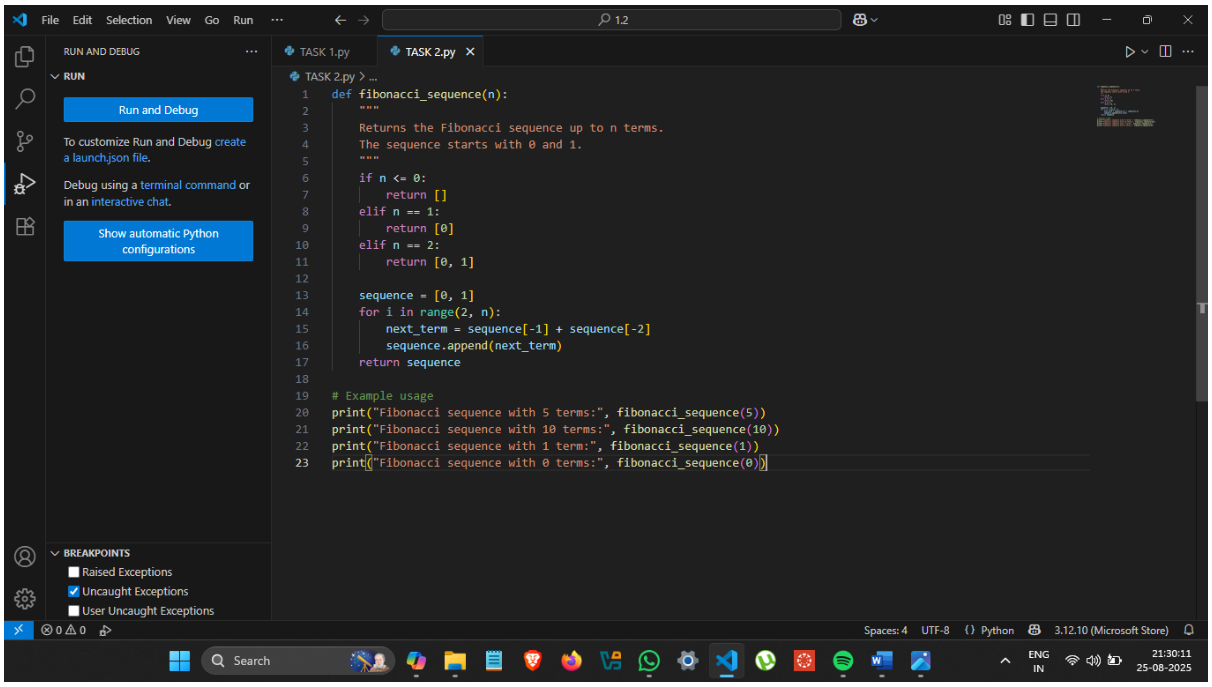
```
PS C:\B.TECH\1.2>  c:; cd 'c:\B.TECH\1.2'; & 'c:\Users\kamer\AppData\Local\Microsoft\WindowsApps\python3.12.exe' 'c:\Users\kamer\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '61594' '--' 'C:\B.TECH\1.2\TASK 1.py'
True
True
False
PS C:\B.TECH\1.2>
```

TASK 2

- **TASK DESCRIPTION:** Generate a Python function that returns the Fibonacci sequence up to n terms. Prompt with only a function header and docstring.

PROMPT: Write a Python function that returns the Fibonacci sequence up to n terms. Include only the function header and a descriptive docstring.

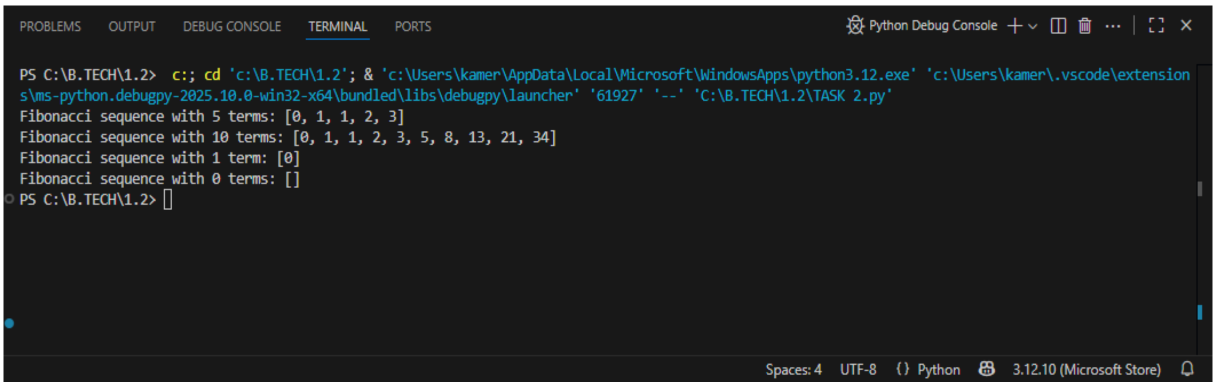
CODE:



File Edit Selection View Go Run
RUN AND DEBUG
RUN
Run and Debug
To customize Run and Debug create a launch.json file.
Debug using a terminal command or in an interactive chat.
Show automatic Python configurations
BREAKPOINTS
Raised Exceptions
Uncaught Exceptions
User Uncaught Exceptions
TASK 1.py
TASK 2.py
Spaces: 4 UTF-8 Python 3.12.10 (Microsoft Store)

```python
def fibonacci_sequence(n):
    """
    Returns the Fibonacci sequence up to n terms.
    The sequence starts with 0 and 1.
    """
    if n <= 0:
        return []
    elif n == 1:
        return [0]
    elif n == 2:
        return [0, 1]

    sequence = [0, 1]
    for i in range(2, n):
        next_term = sequence[-1] + sequence[-2]
        sequence.append(next_term)
    return sequence

# Example usage
print("Fibonacci sequence with 5 terms:", fibonacci_sequence(5))
print("Fibonacci sequence with 10 terms:", fibonacci_sequence(10))
print("Fibonacci sequence with 1 term:", fibonacci_sequence(1))
print("Fibonacci sequence with 0 terms:", fibonacci_sequence(0))
```

OUTPUT:



PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python Debug Console
Spaces: 4 UTF-8 Python 3.12.10 (Microsoft Store)

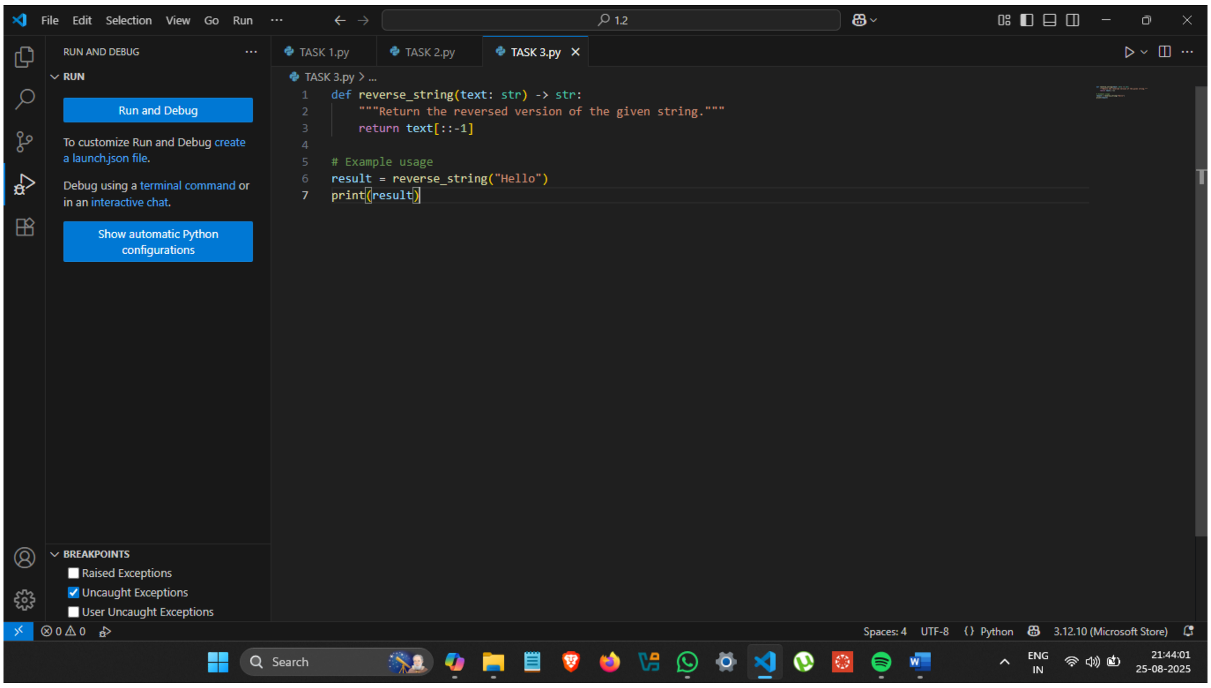
```
PS C:\B.TECH\1.2>  c:; cd 'c:\B.TECH\1.2'; & 'c:\Users\kamer\AppData\Local\Microsoft\WindowsApps\python3.12.exe' 'c:\Users\kamer\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '61927' '--' 'C:\B.TECH\1.2\TASK 2.py'
Fibonacci sequence with 5 terms: [0, 1, 1, 2, 3]
Fibonacci sequence with 10 terms: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
Fibonacci sequence with 1 term: [0]
Fibonacci sequence with 0 terms: []
PS C:\B.TECH\1.2> 
```

TASK 3

Task Description: Write a comment like # Function to reverse a string and use Copilot to generate the function.

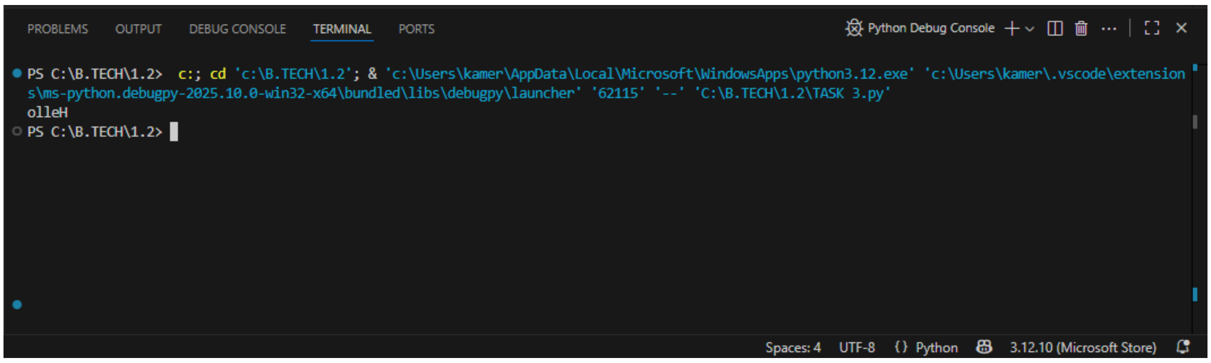
Prompt: Generate a Python function with type hints that takes a string as input and returns the reversed string.

CODE:



```python
def reverse_string(text: str) -> str:
    """Return the reversed version of the given string."""
    return text[::-1]

# Example usage
result = reverse_string("Hello")
print(result)
```

OUTPUT:



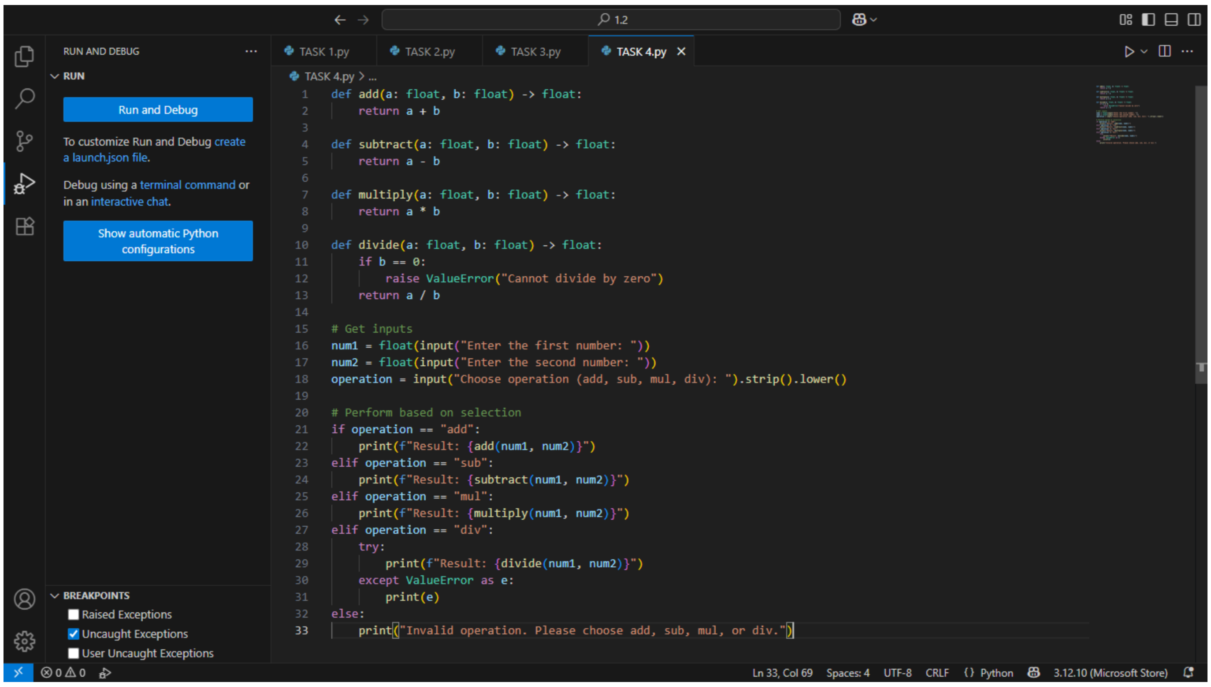
```
PS C:\B.TECH\1.2>  c:; cd 'c:\B.TECH\1.2'; & 'c:\Users\kamer\AppData\Local\Microsoft\WindowsApps\python3.12.exe' 'c:\Users\kamer\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '62115' '--' 'C:\B.TECH\1.2\TASK 3.py'
olleH
PS C:\B.TECH\1.2> 
```

TASK 4

Task Description: Generate a program that simulates a basic calculator (add, subtract, multiply, divide). Write the comment: # Simple calculator with 4 operations and let AI complete it.

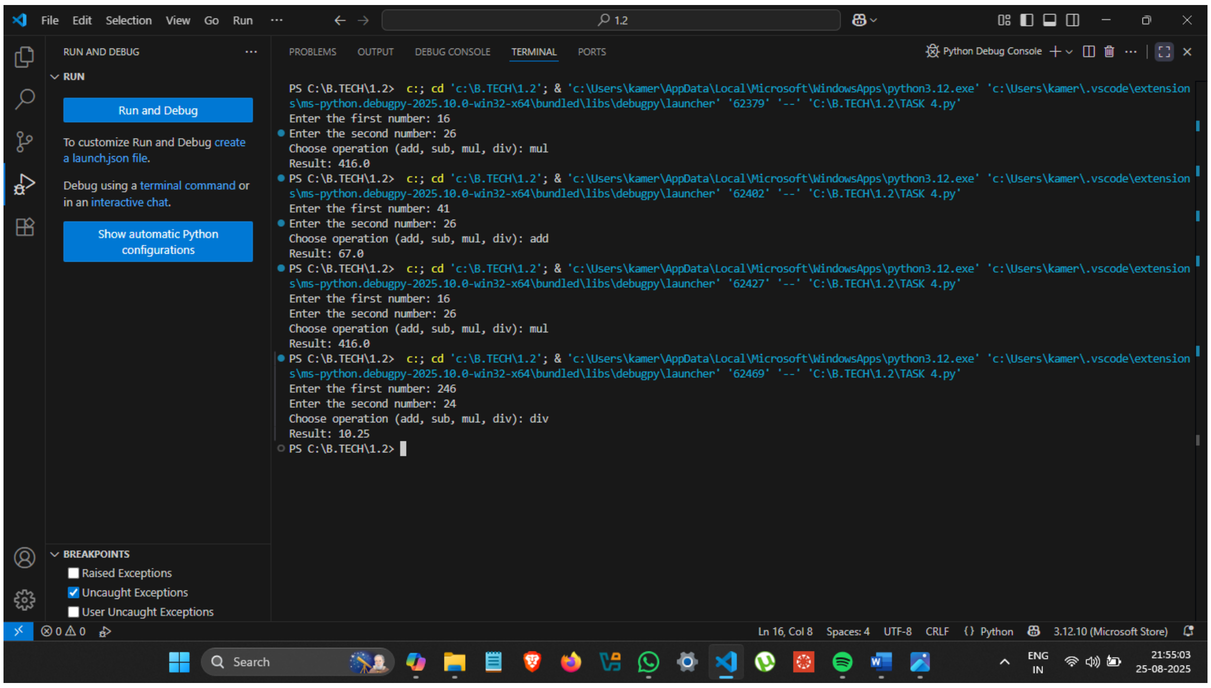
PROMPT: Generate a Python program that takes two numbers as input and performs addition, subtraction, multiplication, and division.

CODE:



```python
def add(a: float, b: float) -> float:
    return a + b

def subtract(a: float, b: float) -> float:
    return a - b

def multiply(a: float, b: float) -> float:
    return a * b

def divide(a: float, b: float) -> float:
    if b == 0:
        raise ValueError("Cannot divide by zero")
    return a / b

# Get inputs
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))
operation = input("Choose operation (add, sub, mul, div): ").strip().lower()

# Perform based on selection
if operation == "add":
    print(f"Result: {add(num1, num2)}")
elif operation == "sub":
    print(f"Result: {subtract(num1, num2)}")
elif operation == "mul":
    print(f"Result: {multiply(num1, num2)}")
elif operation == "div":
    try:
        print(f"Result: {divide(num1, num2)}")
    except ValueError as e:
        print(e)
else:
    print("Invalid operation. Please choose add, sub, mul, or div.")
```

OUTPUT:



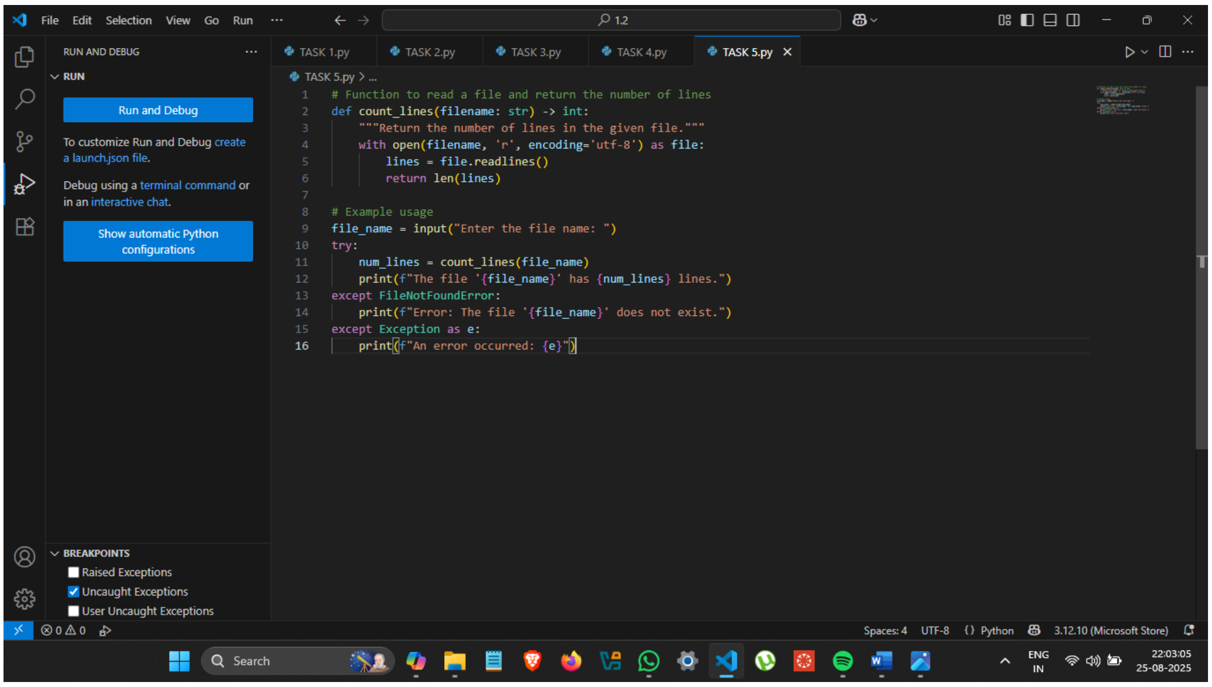
```
PS C:\B.TECH\1.2>  c:; cd 'c:\B.TECH\1.2'; & 'c:\Users\kamer\AppData\Local\Microsoft\WindowsApps\python3.12.exe' 'c:\Users\kamer\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '62379' '--' 'C:\B.TECH\1.2\TASK 4.py'
Enter the first number: 16
Enter the second number: 26
Choose operation (add, sub, mul, div): mul
Result: 416.0
PS C:\B.TECH\1.2>  c:; cd 'c:\B.TECH\1.2'; & 'c:\Users\kamer\AppData\Local\Microsoft\WindowsApps\python3.12.exe' 'c:\Users\kamer\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '62402' '--' 'C:\B.TECH\1.2\TASK 4.py'
Enter the first number: 41
Enter the second number: 26
Choose operation (add, sub, mul, div): add
Result: 67.0
PS C:\B.TECH\1.2>  c:; cd 'c:\B.TECH\1.2'; & 'c:\Users\kamer\AppData\Local\Microsoft\WindowsApps\python3.12.exe' 'c:\Users\kamer\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '62427' '--' 'C:\B.TECH\1.2\TASK 4.py'
Enter the first number: 16
Enter the second number: 26
Choose operation (add, sub, mul, div): mul
Result: 416.0
PS C:\B.TECH\1.2>  c:; cd 'c:\B.TECH\1.2'; & 'c:\Users\kamer\AppData\Local\Microsoft\WindowsApps\python3.12.exe' 'c:\Users\kamer\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '62469' '--' 'C:\B.TECH\1.2\TASK 4.py'
Enter the first number: 246
Enter the second number: 24
Choose operation (add, sub, mul, div): div
Result: 10.25
PS C:\B.TECH\1.2>
```

TASK 5

Task Description: Use a comment to instruct AI to write a function that reads a file and returns the number of lines...

PROMPT: Generate a Python function that opens a file, counts how many lines it contains, and returns that number.

CODE:

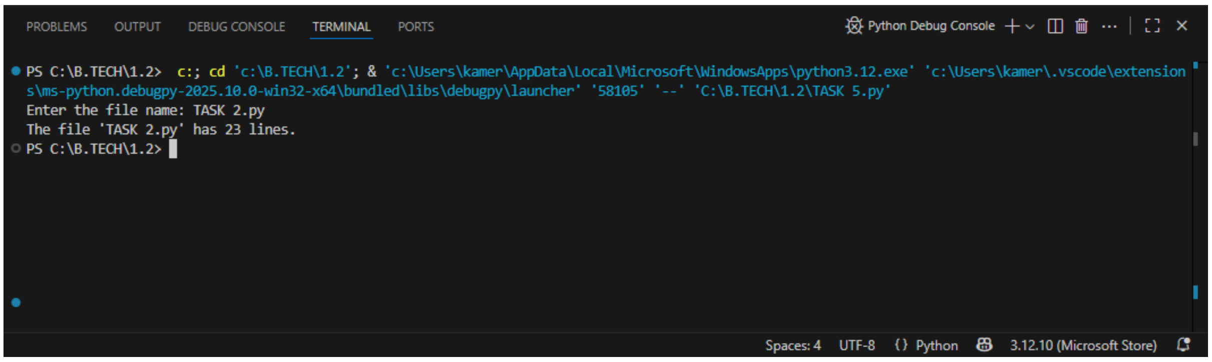


To customize Run and Debug create a launch.json file.

Debug using a terminal command or in an interactive chat.

```python
# Function to read a file and return the number of lines
def count_lines(filename: str) -> int:
    """Return the number of lines in the given file."""
    with open(filename, 'r', encoding='utf-8') as file:
        lines = file.readlines()
        return len(lines)

# Example usage
file_name = input("Enter the file name: ")
try:
    num_lines = count_lines(file_name)
    print(f"The file '{file_name}' has {num_lines} lines.")
except FileNotFoundError:
    print(f"Error: The file '{file_name}' does not exist.")
except Exception as e:
    print(f"An error occurred: {e}")
```

OUTPUT:



```
PS C:\B.TECH\1.2>  c:; cd 'c:\B.TECH\1.2'; & 'c:\Users\kamer\AppData\Local\Microsoft\WindowsApps\python3.12.exe' 'c:\Users\kamer\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '58105' '--' 'C:\B.TECH\1.2\TASK 5.py'
Enter the file name: TASK 2.py
The file 'TASK 2.py' has 23 lines.
PS C:\B.TECH\1.2>
```

OBSERVATION: I observed that GitHub copilot can quickly generate working code for tasks such as login systems, loan approvals, Fibonacci functions, and job applicant scoring. However, the generated code sometimes contains issues like hardcoded values, lack of encryption, or biased decision logic. This shows that AI tools are helpful for faster coding but require human review for security, fairness, and correctness. GitHub Copilot is a fascinating tool to observe—especially in how it transforms the developer experience. Here's a breakdown of key observations across its functionality, impact, and adoption