SR UNIVERSITY

AI ASSISTED CODING

ASSIGNMENT-01

NAME : SHAIK FAHEEM

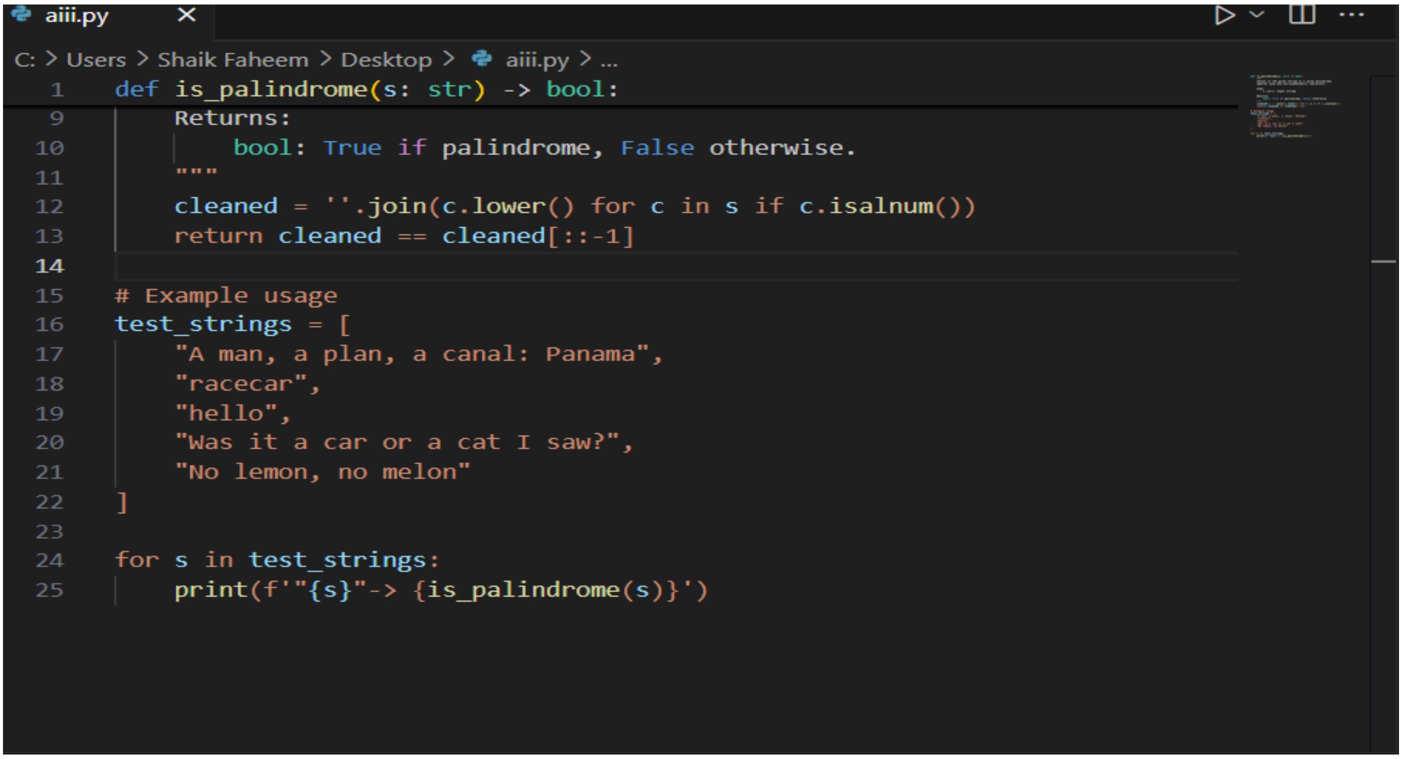
HTNO: 2503A51L39

TASK1

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

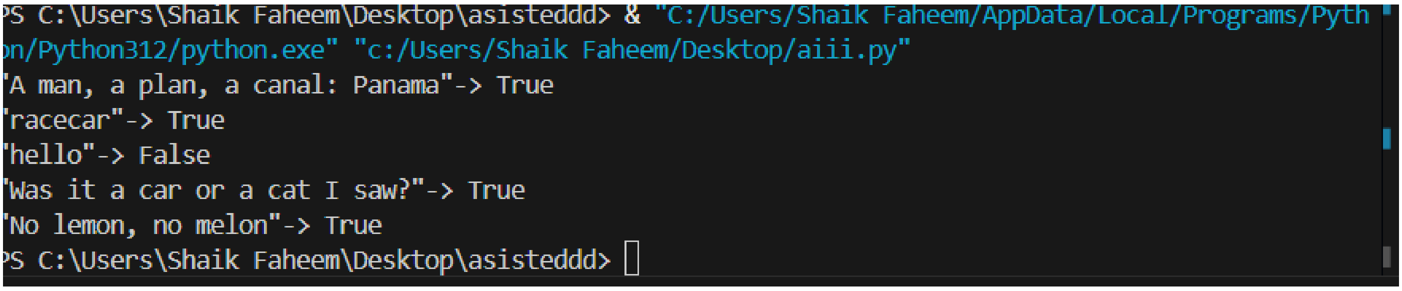
PROMPT:- Generate a comment: # Function to check if a string is a valid palindrome (ignoring spaces and case) and complete it.

CODE



```python
def is_palindrome(s: str) -> bool:
    Returns:
        bool: True if palindrome, False otherwise.
    """
    cleaned = ''.join(c.lower() for c in s if c.isalnum())
    return cleaned == cleaned[::-1]

# Example usage
test_strings = [
    "A man, a plan, a canal: Panama",
    "racecar",
    "hello",
    "Was it a car or a cat I saw?",
    "No lemon, no melon"
]

for s in test_strings:
    print(f'"{s}"-> {is_palindrome(s)}')
```

OUTPUT



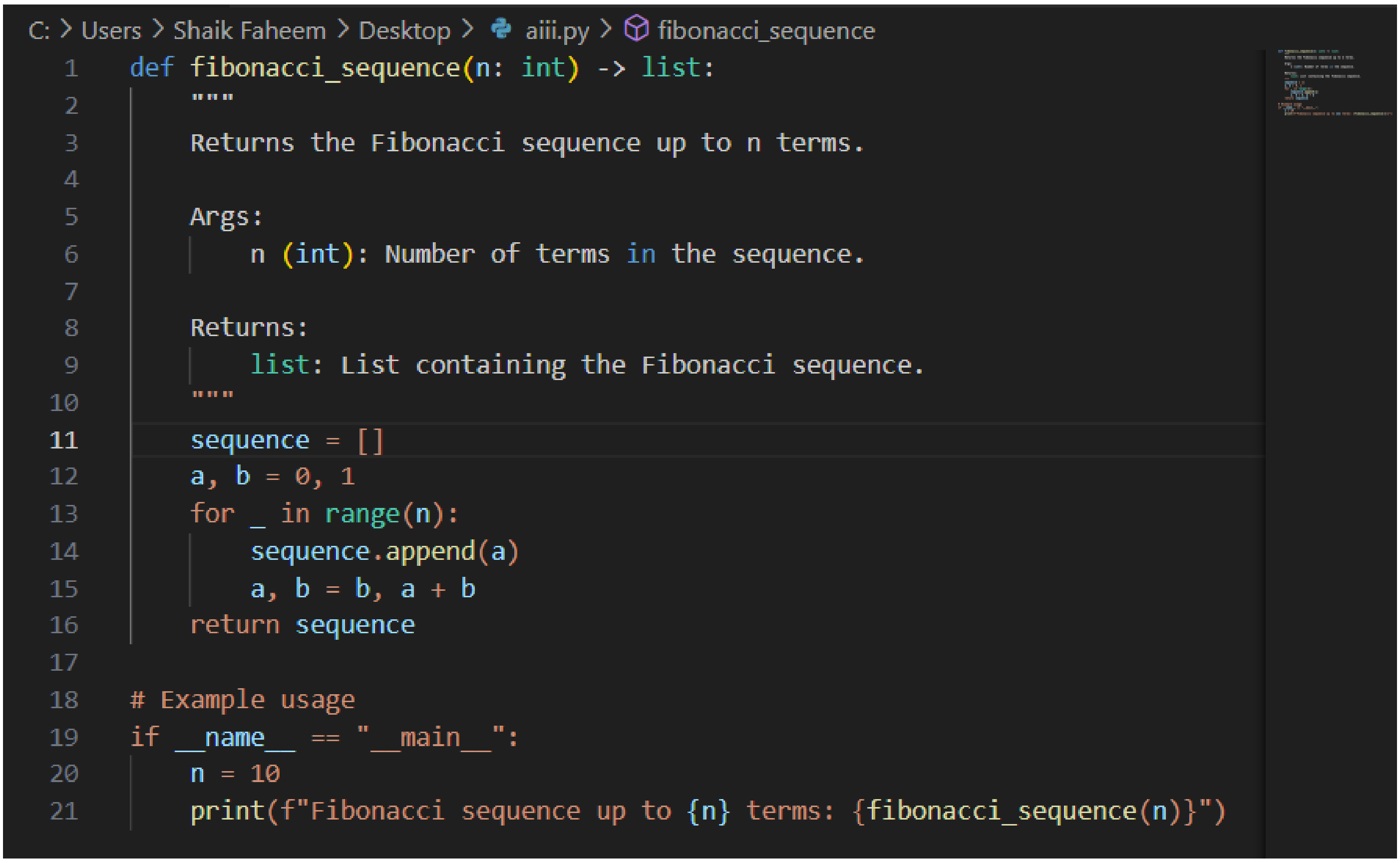
```
PS C:\Users\Shaik Faheem\Desktop\asisteddd> & "C:/Users/Shaik Faheem/AppData/Local/Programs/Pyth
on/Python312/python.exe" "c:/Users/Shaik Faheem/Desktop/aiii.py"
"A man, a plan, a canal: Panama"-> True
"racecar"-> True
"hello"-> False
"Was it a car or a cat I saw?"-> True
"No lemon, no melon"-> True
PS C:\Users\Shaik Faheem\Desktop\asisteddd> 
```

TASK2

TASK2 DESCRIPTION:- Generate a Python function that returns the Fibonacci sequence up to n terms. Prompt with only a function header and docstring

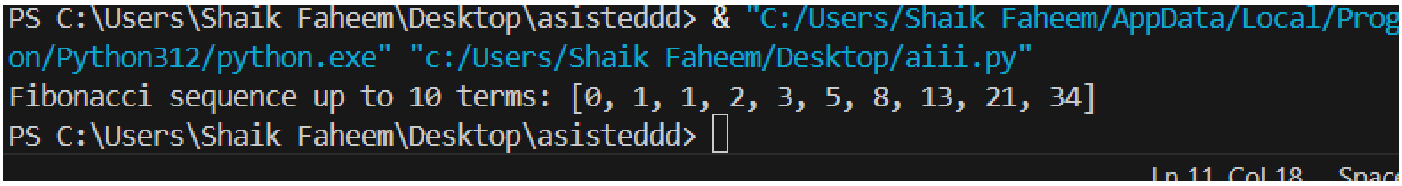
PROMPT:-Generate a Python function that returns the Fibonacci sequence up to n terms. The function should include only the function header and a detailed docstring and should implement with example

CODE


C: > Users > Shaik Faheem > Desktop > aiii.py > fibonacci_sequence

```python
def fibonacci_sequence(n: int) -> list:
    """
    Returns the Fibonacci sequence up to n terms.

    Args:
        n (int): Number of terms in the sequence.

    Returns:
        list: List containing the Fibonacci sequence.
    """
    sequence = []
    a, b = 0, 1
    for _ in range(n):
        sequence.append(a)
        a, b = b, a + b
    return sequence

# Example usage
if __name__ == "__main__":
    n = 10
    print(f"Fibonacci sequence up to {n} terms: {fibonacci_sequence(n)}")
```

OUTPUT

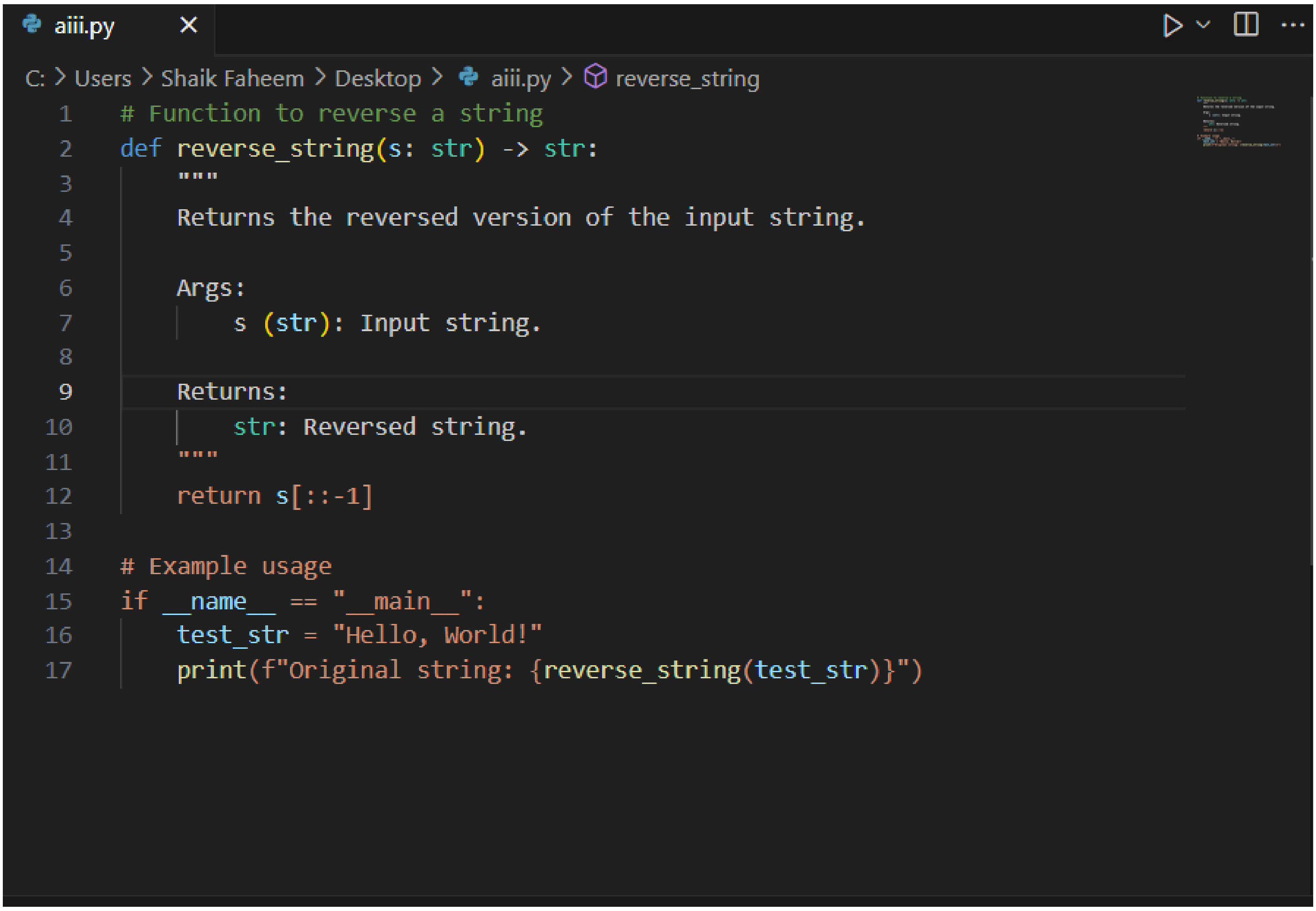


```
PS C:\Users\Shaik Faheem\Desktop\asisteddd> & "C:/Users/Shaik Faheem/AppData/Local/Prog
on/Python312/python.exe" "c:/Users/Shaik Faheem/Desktop/aiii.py"
Fibonacci sequence up to 10 terms: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
PS C:\Users\Shaik Faheem\Desktop\asisteddd> 
```

TASK3

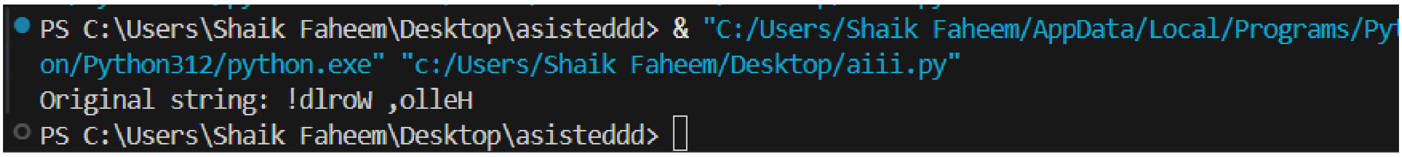
TASK3 DESCRIPTION:- Write a comment like # Function to reverse a string and use Copilot to generate the function.

PROMPT:-Write a comment like # Function to reverse a string and generate the function


aiii.py
C: > Users > Shaik Faheem > Desktop > aiii.py > reverse_string

```python
# Function to reverse a string
def reverse_string(s: str) -> str:
    """
    Returns the reversed version of the input string.

    Args:
        s (str): Input string.

    Returns:
        str: Reversed string.
    """
    return s[::-1]

# Example usage
if __name__ == "__main__":
    test_str = "Hello, World!"
    print(f"Original string: {reverse_string(test_str)}")
```

OUTPUT



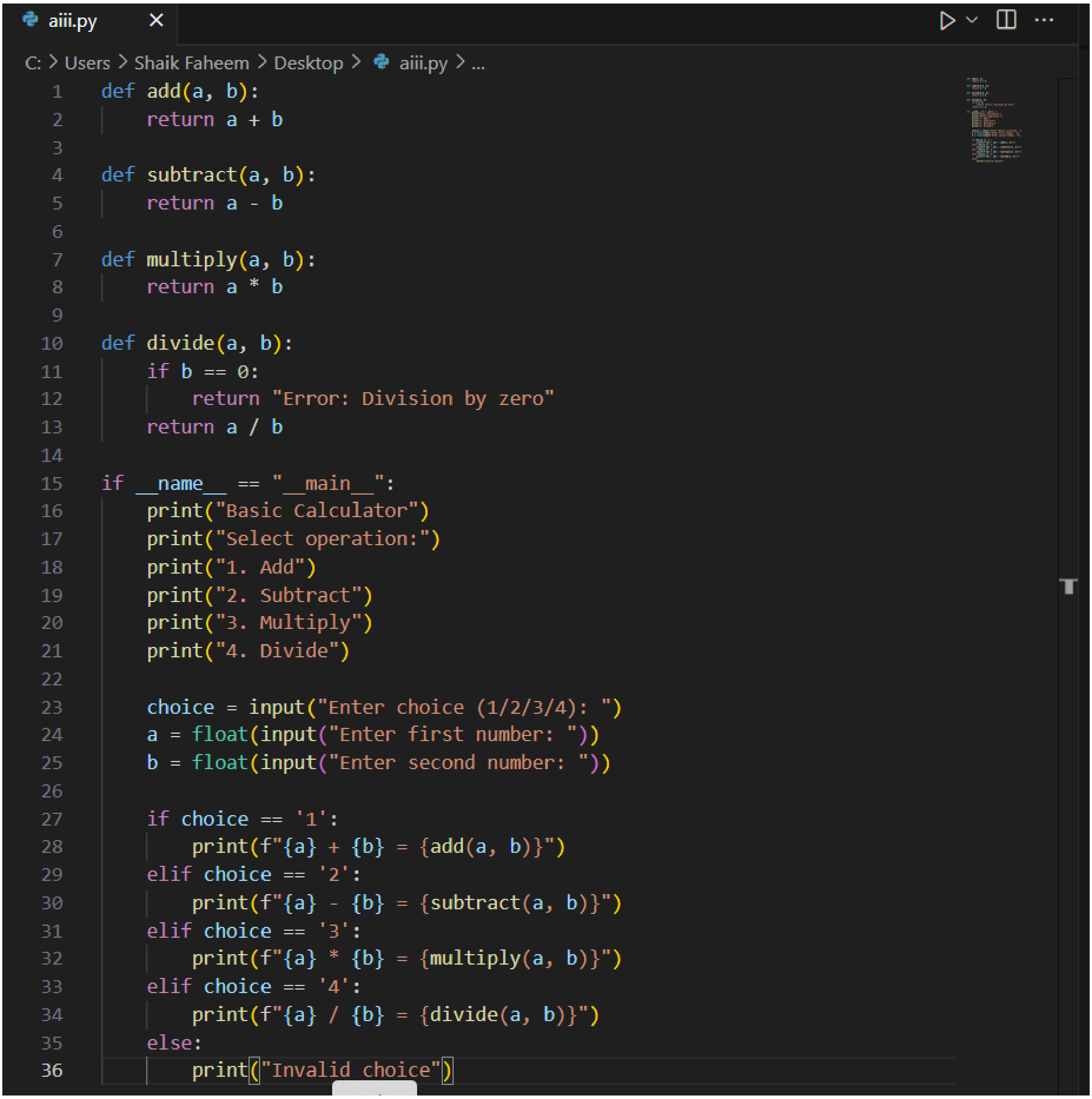
```
PS C:\Users\Shaik Faheem\Desktop\asisteddd> & "C:/Users/Shaik Faheem/AppData/Local/Programs/Pyt
on/Python312/python.exe" "c:/Users/Shaik Faheem/Desktop/aiii.py"
Original string: !dlroW ,olleH
PS C:\Users\Shaik Faheem\Desktop\asisteddd> 
```

TASK4

TASK4 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.

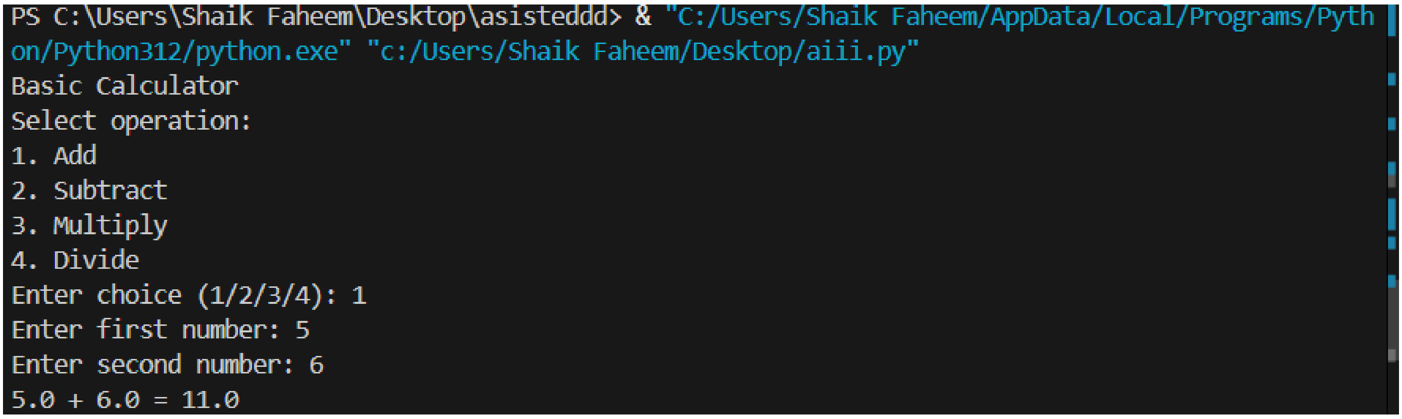
PROMPT:-Write the comment: "# Simple calculator with 4 operations" and let AI complete it. The program should be a fully working calculator with input/output and operator selection logic, supporting add, subtract, multiply, and divide operations.

CODE



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

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

def multiply(a, b):
    return a * b

def divide(a, b):
    if b == 0:
        return "Error: Division by zero"
    return a / b

if __name__ == "__main__":
    print("Basic Calculator")
    print("Select operation:")
    print("1. Add")
    print("2. Subtract")
    print("3. Multiply")
    print("4. Divide")

    choice = input("Enter choice (1/2/3/4): ")
    a = float(input("Enter first number: "))
    b = float(input("Enter second number: "))

    if choice == '1':
        print(f"{a} + {b} = {add(a, b)}")
    elif choice == '2':
        print(f"{a} - {b} = {subtract(a, b)}")
    elif choice == '3':
        print(f"{a} * {b} = {multiply(a, b)}")
    elif choice == '4':
        print(f"{a} / {b} = {divide(a, b)}")
    else:
        print("Invalid choice")
```

OUTPUT



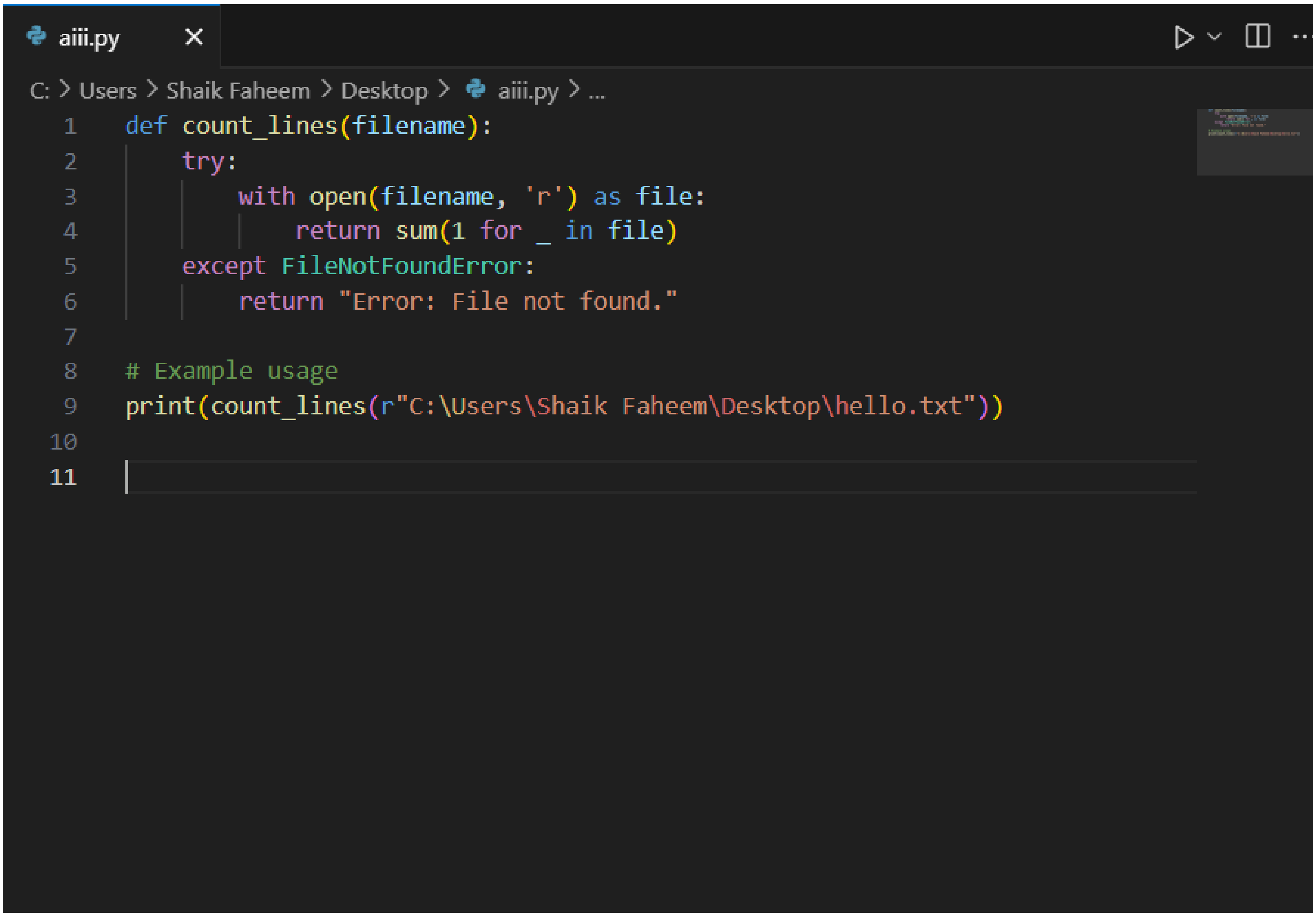
```
PS C:\Users\Shaik Faheem\Desktop\asisteddd> & "C:/Users/Shaik Faheem/AppData/Local/Programs/Pyth
on/Python312/python.exe" "c:/Users/Shaik Faheem/Desktop/aiii.py"
Basic Calculator
Select operation:
1. Add
2. Subtract
3. Multiply
4. Divide
Enter choice (1/2/3/4): 1
Enter first number: 5
Enter second number: 6
5.0 + 6.0 = 11.0
```

TASK5

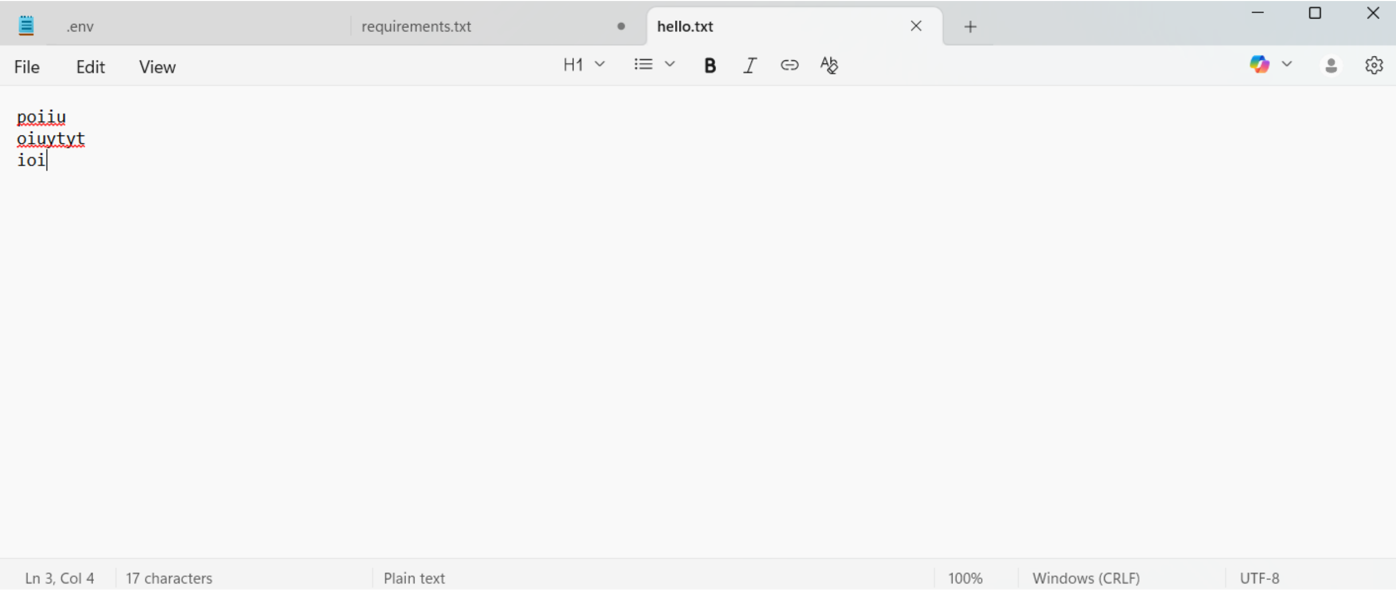
TASK5 DESCRIPTION:- Use a comment to instruct AI to write a function that reads a file and returns the number of lines.

PROMPT:-Write a comment instructing AI to write a function that reads a file and returns the number of lines.

CODE

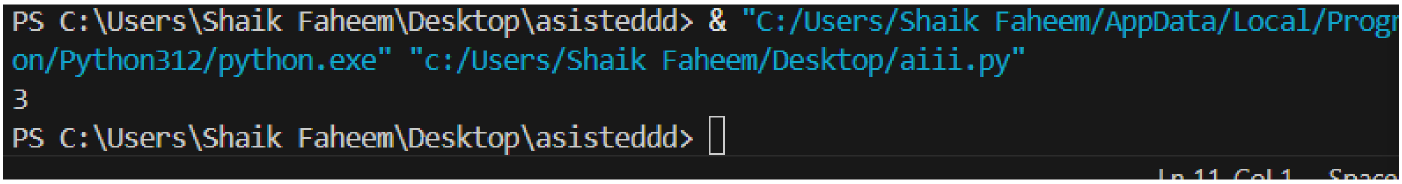


```python
def count_lines(filename):
    try:
        with open(filename, 'r') as file:
            return sum(1 for _ in file)
    except FileNotFoundError:
        return "Error: File not found."

# Example usage
print(count_lines(r"C:\Users\Shaik Faheem\Desktop\hello.txt"))

```



```
poiiu
oiuytyt
ioi
```

OUTPUT



```
PS C:\Users\Shaik Faheem\Desktop\asisteddd> & "C:/Users/Shaik Faheem/AppData/Local/Progr
on/Python312/python.exe" "c:/Users/Shaik Faheem/Desktop/aiii.py"
3
PS C:\Users\Shaik Faheem\Desktop\asisteddd> 
```

OBSERVATION:- I explored how GitHub Copilot can generate complete and functional Python programs from minimal prompts such as comments, function headers, or docstrings. Each task demonstrated the usefulness of AI-assisted coding in reducing effort, saving time, and improving productivity**.**

Task 1.2 obv: This code defines a function to generate the Fibonacci sequence up to n terms and prints the result for n = 10. It uses a loop to build the sequence in a list anD displays it when run.

Task 1.3 obv: This program defines a function to reverse any input string. It demonstrates the function by reversing "Hello, World!" and printing both the original and reversed strings. The slicing operation `s[::-1]` efficiently reverses the string in Python.

Task1.4 obv: This program implements a basic calculator with add, subtract, multiply, and divide functions. It prompts the user to select an operation and enter two numbers, then displays the result. Division by zero is handled gracefully by returning an error message.

Task 1.5 obv: The function successfully counts the number of lines in a given file.It uses a generator expression, making the code efficient and concise.