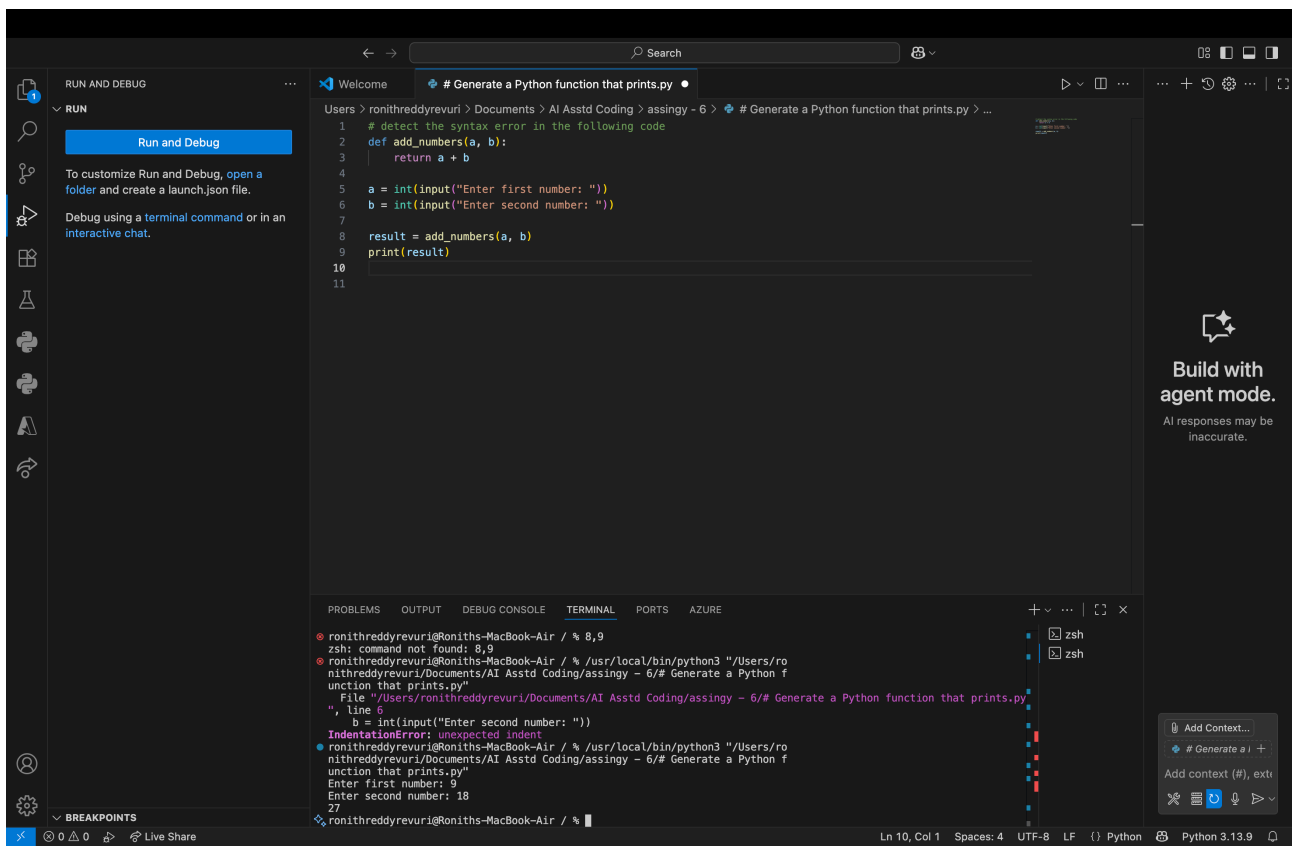
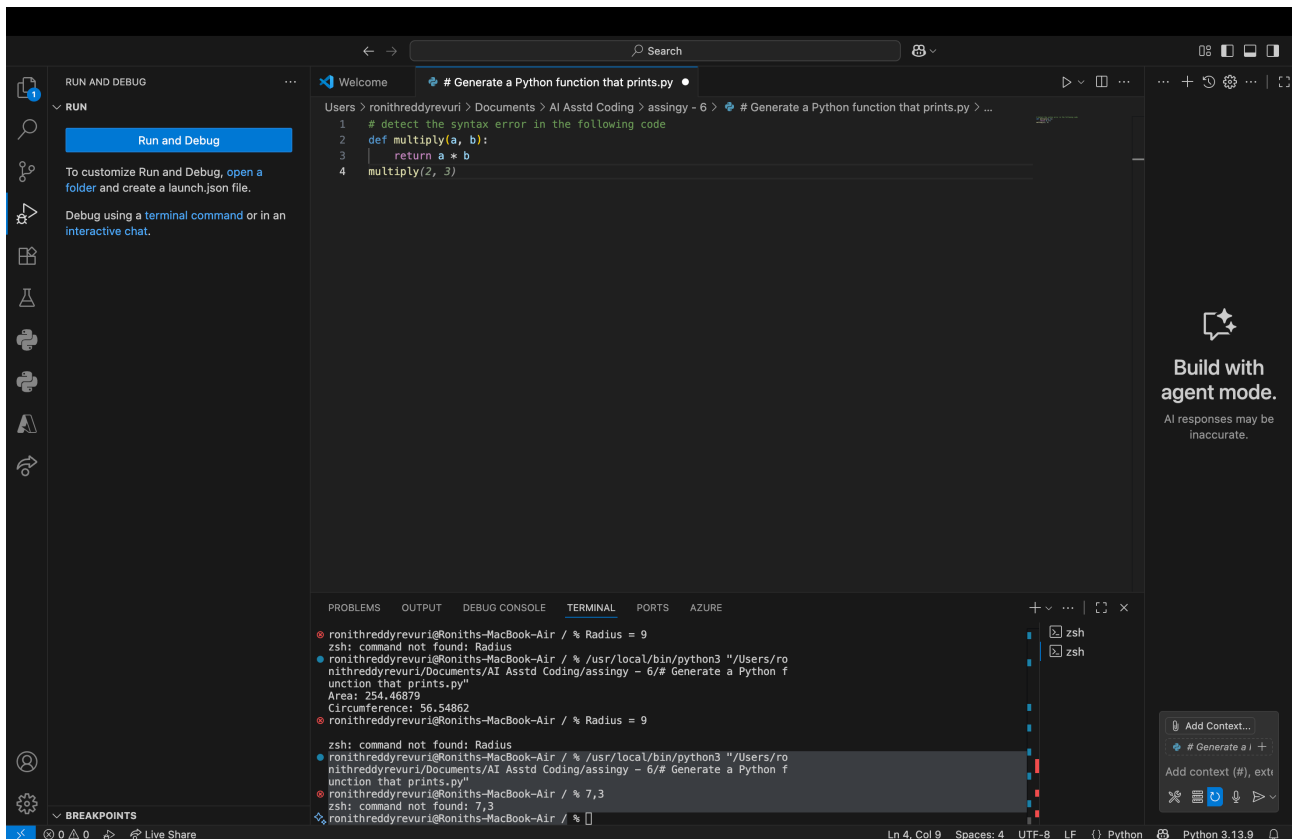


AI ASSISTED CODING - ASSIGNMENT - 7

NAME : R. RONITH REDDY
HTNO : 2303A52280

TASK - 1 : FIXING SYNTAX ERRORS SCENARIO

PROMPT :
DETECT THE SYNTAX ERRORS

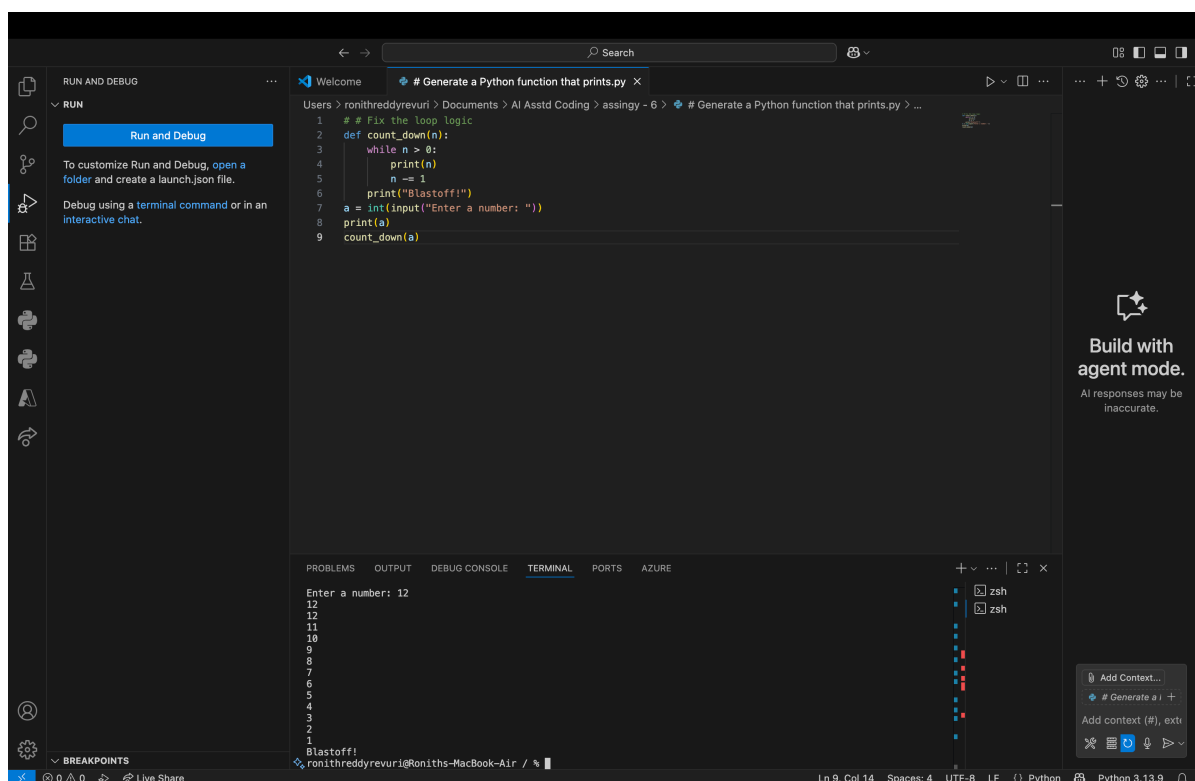
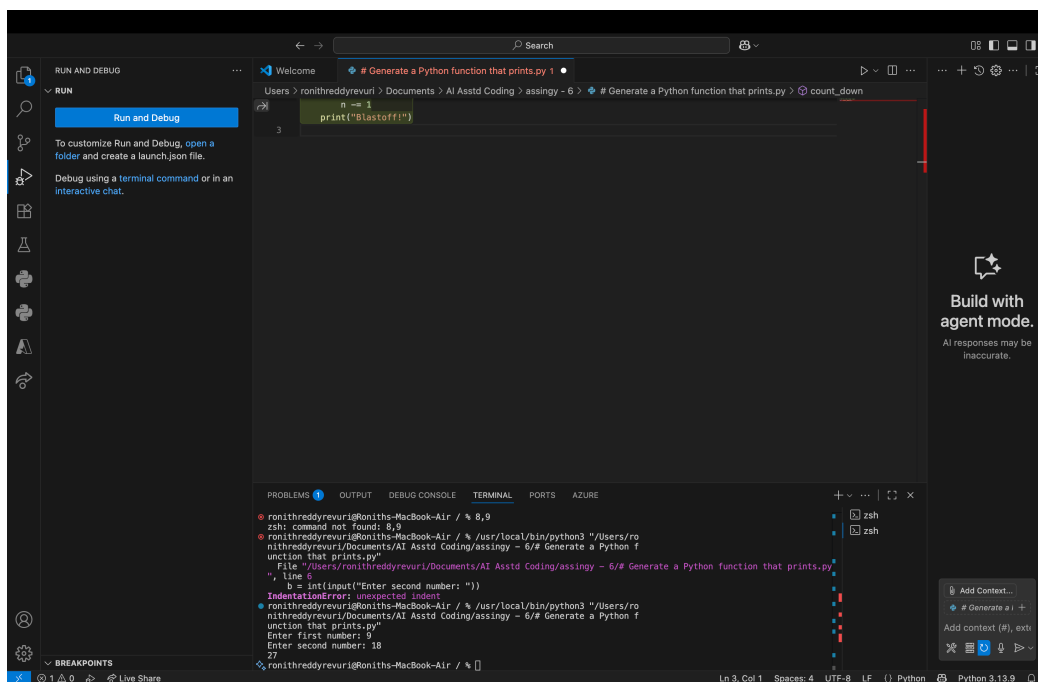


EXPLANATION:

THE ERROR HAPPENED BECAUSE **RETURN** WAS WRITTEN OUTSIDE THE FUNCTION, WHICH PYTHON DOESN'T ALLOW. THERE WAS ALSO A SMALL INDENTATION MISTAKE. ONCE THESE WERE FIXED AND THE CODE WAS PROPERLY STRUCTURED, THE PROGRAM WORKED AS EXPECTED.

TASK - 2 : DEBUGGING LOGIC ERRORS IN LOOPS

PROMPT : FIX THE LOOP LOGIC



EXPLANATION:

THE LOOP WAS RUNNING INFINITELY BECAUSE THE VALUE WAS INCREASING INSTEAD OF DECREASING. SINCE THE CONDITION NEVER BECAME FALSE, THE LOOP NEVER STOPPED. BY DECREMENTING THE VALUE CORRECTLY, THE LOOP NOW ENDS AS EXPECTED.

TASK - 3 : HANDLING RUNTIME ERRORS (DIVISION BY ZERO)

PROMPT : IDENTIFY THE RUNTIME ERROR AND FIX IT SO THAT THE FUNCTION EXECUTES SAFELY

The screenshot shows the VS Code editor with a file named `# Generate a Python function that prints.py`. The code defines a function `divide_numbers(a, b)` that checks for a zero divisor and prints an error message. It then calls the function with `divide_numbers(10, 2)`, which outputs `5.0`. The terminal shows the execution of the code, which runs without errors.

```
1 # Identify the runtime error and fix it so that the function executes safely
2 def divide_numbers(a, b):
3     if b == 0:
4         print("Error: Cannot divide by zero")
5         return None
6     return a / b
7
8 print(divide_numbers(10, 2)) # Output: 5.0
9
```

The terminal output shows the execution of the code, which runs without errors.

```
print(divide_numbers(10, 0)) # This will raise a ZeroDivisionError
File "/Users/ronithreddyrevuri/Documents/AI Asstd Coding/assingy - 6/# Generate a Python function that prints.py", line 3, in divide_numbers
    return a / b
ZeroDivisionError: division by zero
ronithreddyrevuri@Roniths-MacBook-Air / % /usr/local/bin/python3 "/Users/ronithreddyrevuri/Documents/AI Asstd Coding/assingy - 6/# Generate a Python function that prints.py"
Error: Cannot divide by zero
None
ronithreddyrevuri@Roniths-MacBook-Air / % /usr/local/bin/python3 "/Users/ronithreddyrevuri/Documents/AI Asstd Coding/assingy - 6/# Generate a Python function that prints.py"
5.0
ronithreddyrevuri@Roniths-MacBook-Air / %
```

The screenshot shows the VS Code editor with the same file `# Generate a Python function that prints.py`. The code is modified to include a loop that repeatedly calls the `divide_numbers` function with a decreasing value of `b` until it reaches 1. The terminal shows the execution of the code, which runs without errors.

```
1 # Identify the runtime error and fix it so that the function executes safely
2 def divide_numbers(a, b):
3     if b == 0:
4         return "Error: Division by zero is not allowed."
5     return a / b
6
7 # Loop to test the function with decreasing values of b
8 for i in range(12, 0, -1):
9     result = divide_numbers(12, i)
10    print(f"12 / {i} = {result}")
11
12 # End of loop
13
```

The terminal output shows the execution of the code, which runs without errors.

```
Enter a number: 12
12
12
11
10
9
8
7
6
5
4
3
2
1
Blastoff!
ronithreddyrevuri@Roniths-MacBook-Air / %
```

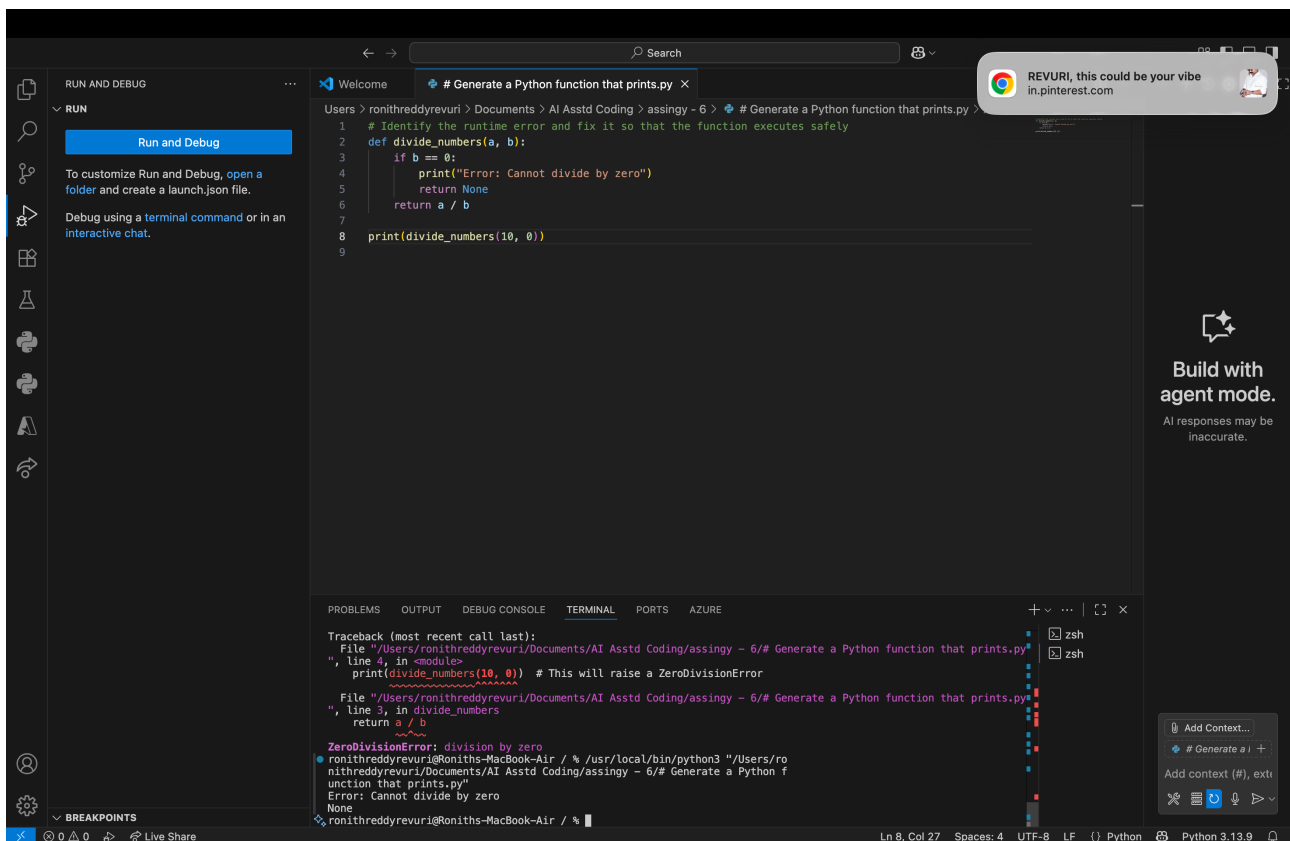
EXPLANATION :

THE ERROR OCCURRED BECAUSE DIVISION BY ZERO IS NOT PERMITTED IN PYTHON. BY CHECKING THE DIVISOR BEFORE PERFORMING THE OPERATION, THE FUNCTION AVOIDS CRASHING AND EXECUTES SAFELY.

TASK - 4 : DEBUGGING CLASS DEFINITION ERRORS

PROMPT :

IDENTIFY THE ISSUE IN THE `__INIT__` CONSTRUCTOR AND CORRECT THE CLASS DEFINITION



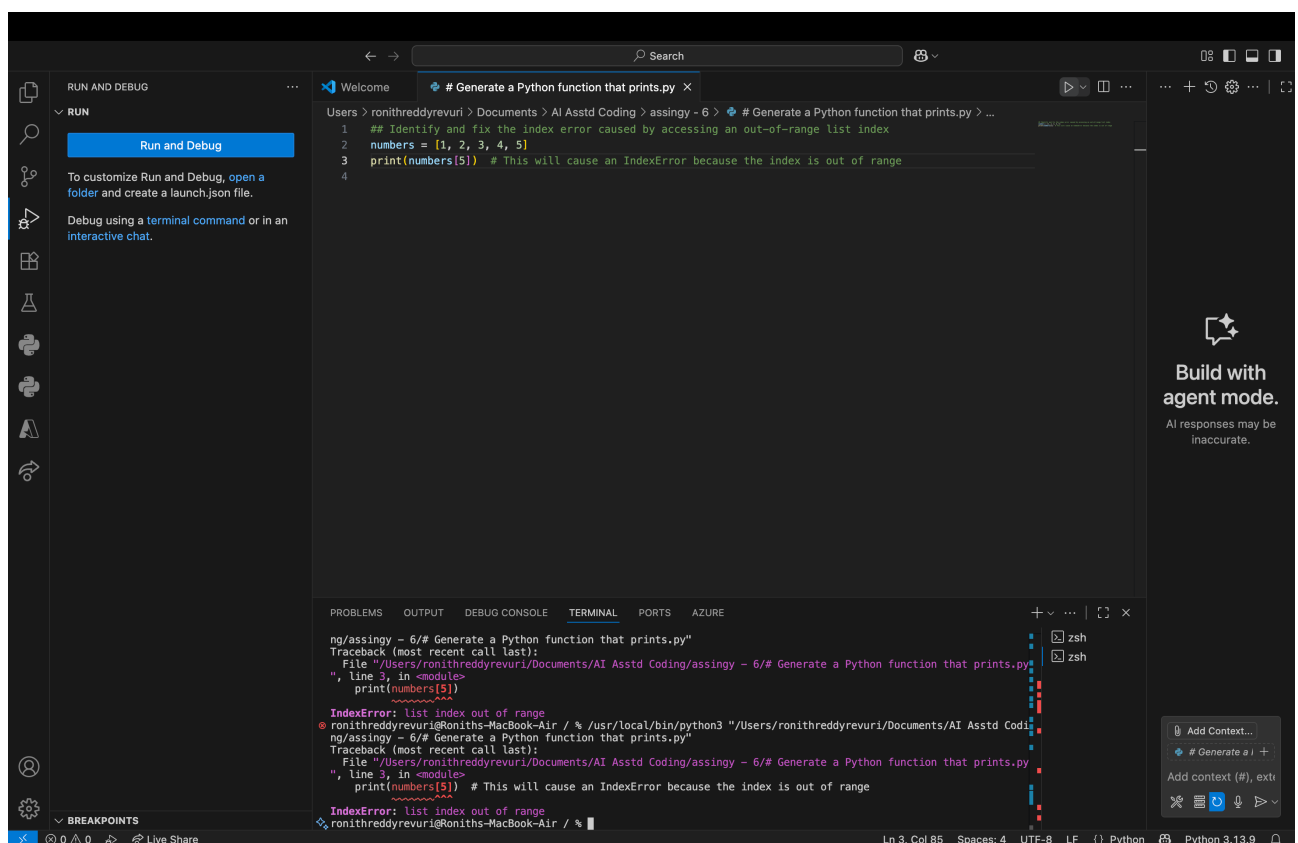
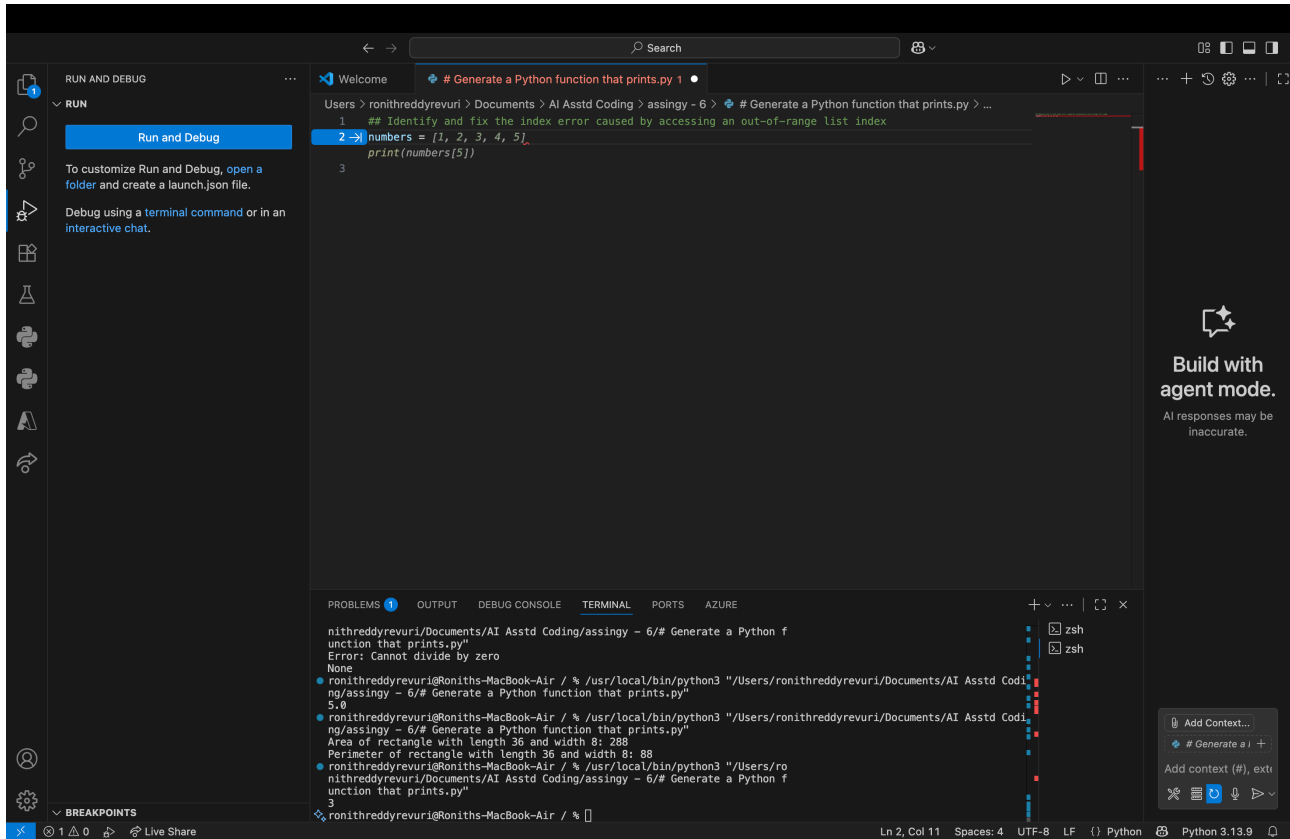
EXPLANATION :

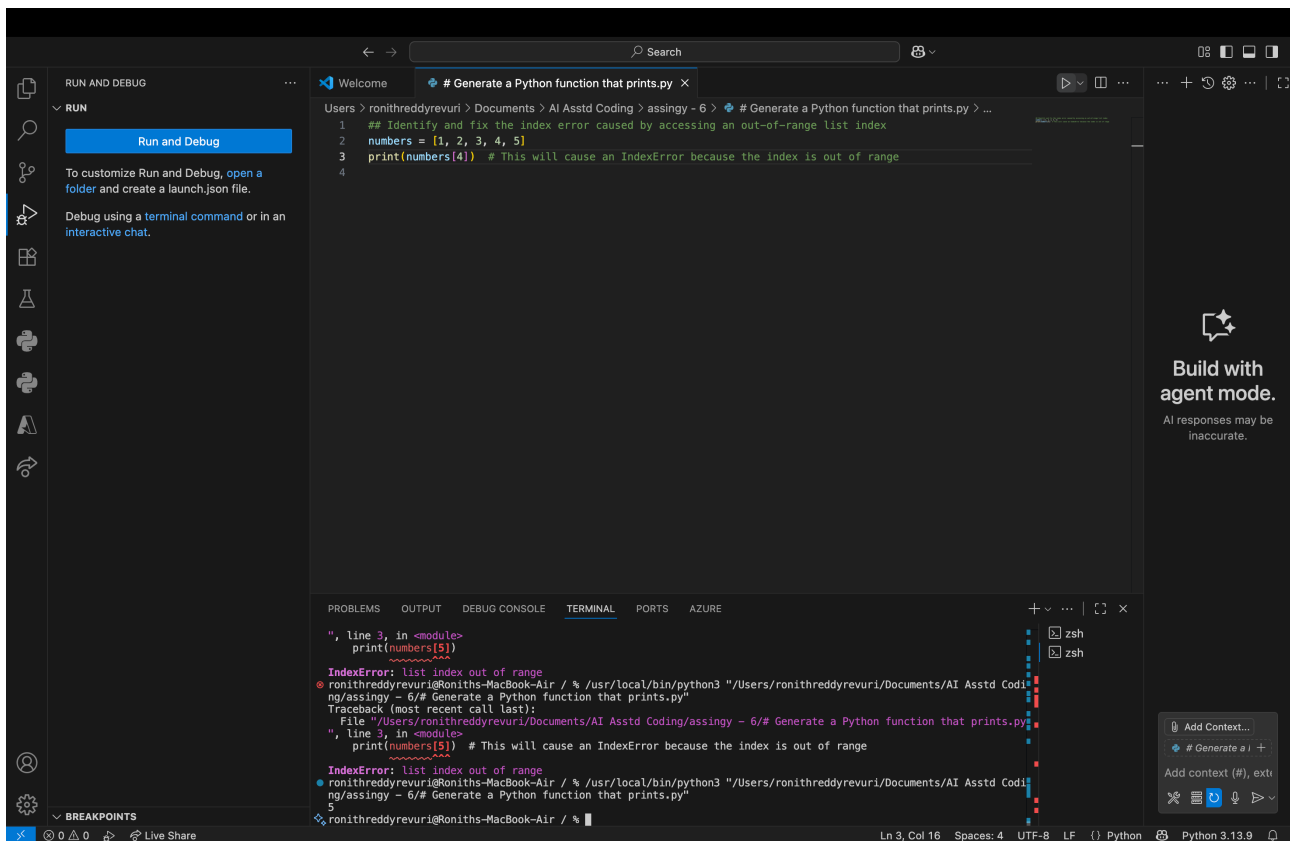
THE CONSTRUCTOR WAS MISSING THE **SELF** PARAMETER, WHICH IS REQUIRED TO ACCESS OBJECT VARIABLES. ADDING **SELF** FIXES THE CLASS DEFINITION AND ALLOWS THE OBJECT TO STORE VALUES PROPERLY.

TASK - 5 : RESOLVING INDEX ERRORS IN LISTS

PROMPT :

IDENTIFY AND FIX THE INDEX ERROR CAUSED BY ACCESSING AN OUT-OF-RANGE LIST INDEX





EXPLANATION :
THE ERROR OCCURRED BECAUSE THE PROGRAM TRIED TO ACCESS A LIST INDEX THAT DOES NOT EXIST. BY ACCESSING A VALID INDEX OR CHECKING THE LIST LENGTH, THE PROGRAM RUNS WITHOUT CRASHING.

CONCLUSION :
THIS LAB SHOWS HOW AI TOOLS CAN HELP IDENTIFY AND FIX SYNTAX, LOGIC, AND RUNTIME ERRORS IN PYTHON PROGRAMS IN A CLEAR AND SYSTEMATIC WAY.

