Task 1: Fixing Syntax Errors

Scenario

You are reviewing a Python program where a basic function definition contains a syntax error.

Requirements

• Provide a Python function add(a, b) with a missing colon

• Use an AI tool to detect the syntax error

• Allow AI to correct the function definition

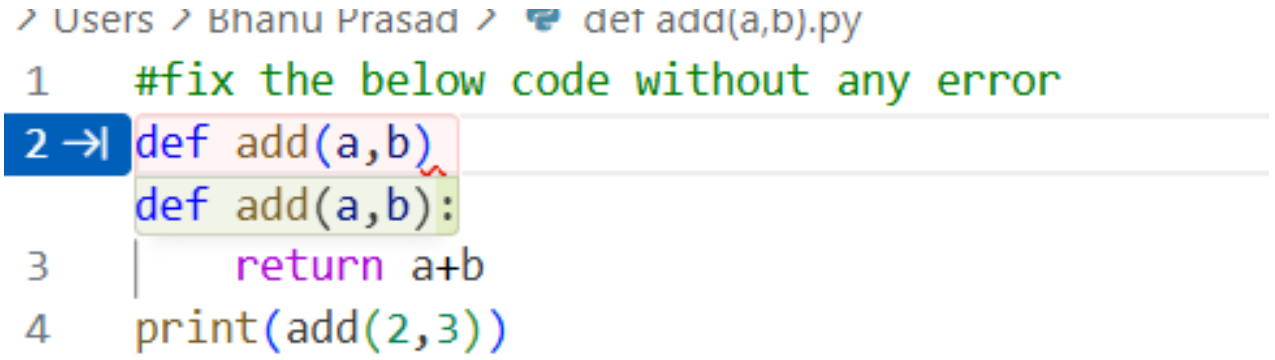
• Observe how AI explains the syntax issue

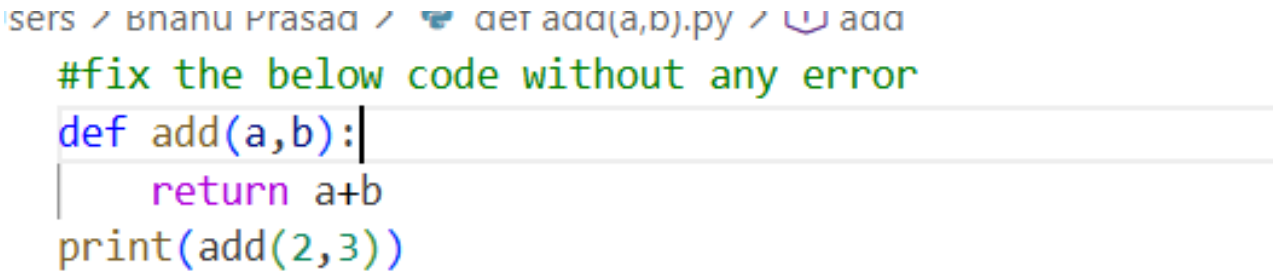
Expected Output

• Corrected function with proper syntax

• Syntax error resolved successfully

• AI-generated explanation of it.





EXPLANATION:

In the I have missed a colon after the declaration of the function It has raised me the error .

Debugging Logic Errors in Loops

Scenario

You are debugging a loop that runs infinitely due to a logical mistake.

Requirements

• Provide a loop with an increment or decrement error

• Use AI to identify the cause of infinite iteration

• Let AI fix the loop logic

• Analyze the corrected loop behavior

Expected Output

• Infinite loop issue resolved

• Correct increment/decrement logic applied

• AI explanation of the logic error.

```
#fix the below code without any error
def countdown(n):
    while n > 0:
        print(n)
        n += 1
# The abov ↳ de will result in an infinite loop because n is being incremented.
# To fix it, we should decrement n instead of incrementing it.
```

```
#fix the below code without any error
def countdown(n):
    while n > 0:
        print(n)
        n -= 1
# The above code will result in an infinite loop because n is being incremented.
# To fix it, we should decrement n instead of incrementing it.
# example usage:
countdown(5)
```

```
PS C:\Users\Bhanu Prasad> & C:/Python314/python.exe "c:/Users/Bhanu Prasad/def add(a,b).py"
5
4
3
2
Manage
PS C:\Users\Bhanu Prasad>
```

EXPLANATION:

Here the logic is incorrect where the n is incremented it gooes into the infinite loops

Then I decremented the value of n the code is logically collected.

Task

3: Handling Runtime Errors (Division by Zero)

Scenario

A Python function crashes during execution due to a division by zero error.

Requirements

• Provide a function that performs division without validation

• Use AI to identify the runtime error

• Let AI add try-except blocks for safe execution

• Review AI's error-handling approach

Expected Output

• Function executes safely without crashing

• Division by zero handled using try-except

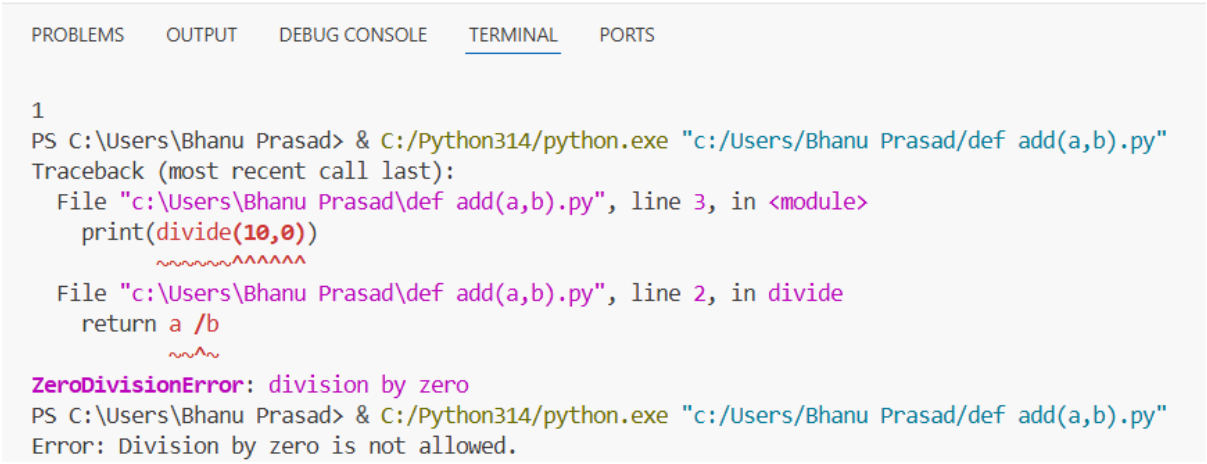
• Clear AI-generated explanation of runtime error handling

```
C: > Users > Bhanu Prasad > def add(a,b).py > ...
1   def divide(a,b):
2       return a /b
3   print(divide(10,0))
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

3
2
1
PS C:\Users\Bhanu Prasad> & C:/Python314/python.exe "c:/Users/Bhanu Prasad/def add(a,b).py"
Traceback (most recent call last):
  File "c:\Users\Bhanu Prasad\def add(a,b).py", line 3, in <module>
    print(divide(10,0))
          ~~~~~~^^^^^^
  File "c:\Users\Bhanu Prasad\def add(a,b).py", line 2, in divide
    return a /b
           ~~^~
ZeroDivisionError: division by zero
```

```python
def divide(a,b):
    try:
        result = a / b
    except ZeroDivisionError:
        return "Error: Division by zero is not allowed."
    return result
print(divide(10, 0))
```



EXPLANATION:

Here according the logic where the number is not divisible by zero it raises me the error.this error is exceped by using the try and exception handling in the code . where it will not raise any error it shows the output as a statement i.e number is not divisible by zero.

Task 4: Debugging Class Definition Errors

Scenario

You are given a faulty Python class where the constructor is incorrectly defined.

Requirements

• Provide a class definition with missing self-parameter

• Use AI to identify the issue in the __init__() method

• Allow AI to correct the class definition

• Understand why self is required

Expected Output

• Corrected __init__() method

• Proper use of self in class definition

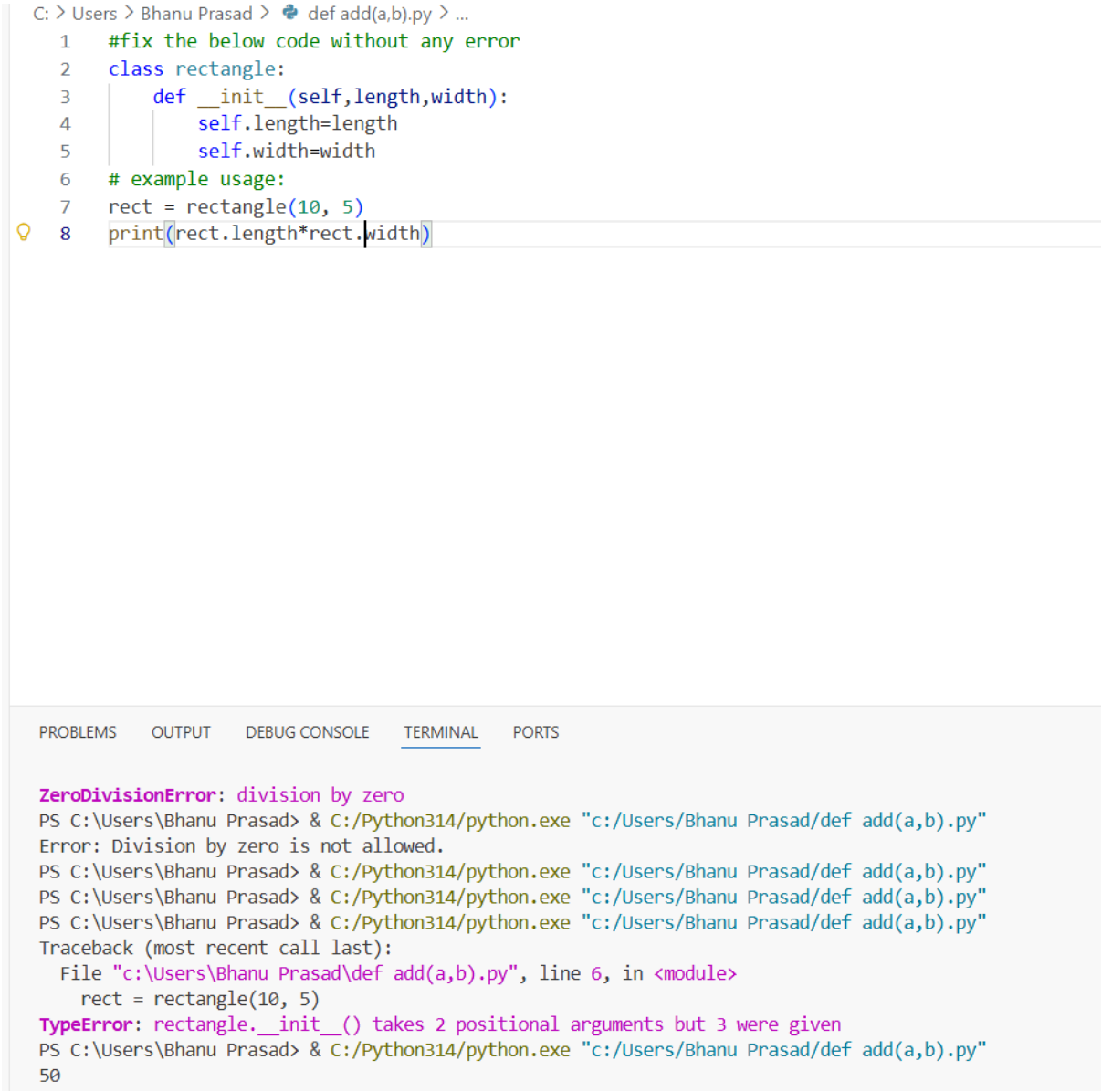
• AI explanation of object-oriented error

C: > Users > Bhanu Prasad > def add(a,b).py > ...

```python
class rectangle:
    def __init__(length,width):
        self.length=length
        self.width=width
# example usage:
rect = rectangle(10, 5)
print(length*width)  # Output: 50
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
    return a /b
           ~~~
ZeroDivisionError: division by zero
PS C:\Users\Bhanu Prasad> & C:/Python314/python.exe "c:/Users/Bhanu Prasad/def add(a,b).py"
Error: Division by zero is not allowed.
PS C:\Users\Bhanu Prasad> & C:/Python314/python.exe "c:/Users/Bhanu Prasad/def add(a,b).py"
PS C:\Users\Bhanu Prasad> & C:/Python314/python.exe "c:/Users/Bhanu Prasad/def add(a,b).py"
PS C:\Users\Bhanu Prasad> & C:/Python314/python.exe "c:/Users/Bhanu Prasad/def add(a,b).py"
Traceback (most recent call last):
  File "c:\Users\Bhanu Prasad\def add(a,b).py", line 6, in <module>
    rect = rectangle(10, 5)
TypeError: rectangle.__init__() takes 2 positional arguments but 3 were given
PS C:\Users\Bhanu Prasad>
```



EXPLANATION:

Here I missed the self parameter while using the constructor it has raised me the error.

After the error occurs I have give the prompt to check the code then ai fixes the error.

Task 5: Resolving Index Errors in Lists

Scenario

A program crashes when accessing an invalid index in a list.

Requirements

• Provide code that accesses an out-of-range list index

• Use AI to identify the Index Error

• Let AI suggest safe access methods

• Apply bounds checking or exception handling

Expected Output

• Index error resolved

• Safe list access logic implemented

• AI suggestion using length checks or exception handlingd which results in thbe output without going in to the infinite loop.

C: > Users > Bhanu Prasad > def add(a,b).py > ...

```python
#fix the below code without any error using the try and exception handling
numbers = [1,2,3]
try:
    print(numbers[5])
except IndexError:
    print("Error: Index is out of bounds.")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\Bhanu Prasad> & C:/Python314/python.exe "c:/Users/Bhanu Prasad/def add(a,b).py"
Traceback (most recent call last):
  File "c:\Users\Bhanu Prasad\def add(a,b).py", line 2, in <module>
    print(numbers[5])
          ~~~~~~~^^^
IndexError: list index out of range
PS C:\Users\Bhanu Prasad> & C:/Python314/python.exe "c:/Users/Bhanu Prasad/def add(a,b).py"
Error: Index is out of bounds.
PS C:\Users\Bhanu Prasad>
```

C: › Users › Bhanu Prasad › def add(a,b).py › ...

```
1   numbers = [1,2,3]
2   print(numbers[5])
        try:
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS C:\Users\Bhanu Prasad> & C:/Python314/python.exe "c:/Users/Bhanu Prasad/def add(a,b).py"
Traceback (most recent call last):
  File "c:\Users\Bhanu Prasad\def add(a,b).py", line 2, in <module>
    print(numbers[5])
          ~~~~~~~^^^
IndexError: list index out of range
PS C:\Users\Bhanu Prasad>
```

EXPLANATION:

Here I have declared number array of size of 3.then I have printed the number[5].i have given error has the index is out of range . by using the try and exception I have showed the output as the index is out of range.Without any error