

ASSIGNMENT – 7.3

2303A51396

Task 1: Fixing Syntax Errors

Scenario

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

Requirements

Python

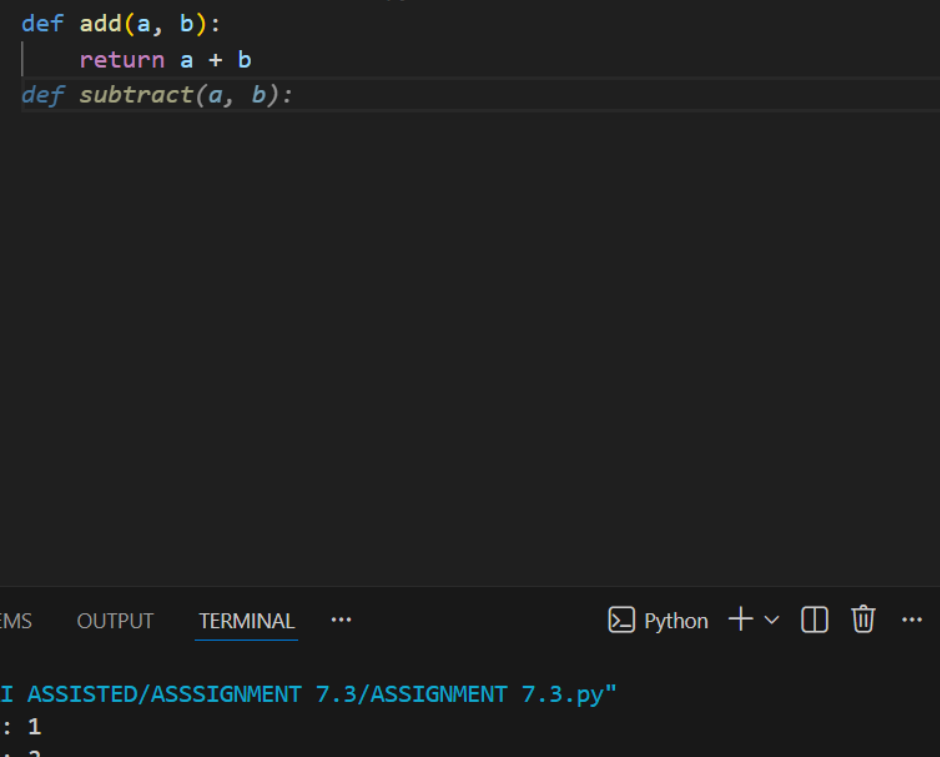
```
def add(a,b)
```

```
    return a + b
```

- 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 the fix.



The image shows a code editor window with a dark theme. The top bar displays the file name "ASSIGNMENT 7.3.py" and a close button. Below the top bar, the editor shows the following Python code:

```
1 def add(a, b):  
2     return a + b  
3 def subtract(a, b):
```

The bottom panel of the editor is divided into three tabs: "PROBLEMS", "OUTPUT", and "TERMINAL". The "TERMINAL" tab is active, showing the output of the program. The output text is:

```
nts/AI ASSISTED/ASSSIGNMENT 7.3/ASSIGNMENT 7.3.py"  
Count: 1  
Count: 2  
Count: 3  
Count: 4  
Count: 5
```

The image shows a VS Code editor window with a file named 'ASSIGNMENT 7.3.py'. The code is a Python script with a while loop. The terminal at the bottom shows the command to run the script using the Python interpreter and debugpy launcher. The output of the script is visible in the terminal, showing the numbers 1 through 5.

```
ASSIGNMENT 7.3 > ASSIGNMENT 7.3.py > ...
1
2
3 count = 1
4 while count <= 5:
5     print(count)
6     count += 1 # FIX: Increment the counter to avoid infinite loop
7
8
```

```
PS C:\Users\PRANAY\OneDrive\Documents\AI ASSISTED> & 'c:\Users\PRANAY\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\PRANAY\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '60242' '--' 'C:\Users\PRANAY\OneDrive\Documents\AI ASSISTED\ASSIGNMENT 7.3\ASSIGNMENT 7.3.py'
1
2
3
4
5
PS C:\Users\PRANAY\OneDrive\Documents\AI ASSISTED>
```

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

The screenshot shows a VS Code editor window with a file named 'ASSIGNMENT 7.3.py'. The code defines a function 'divide(a, b)' that uses a try-except block to handle a ZeroDivisionError. It then prints the result of 'divide(10, 0)'. The terminal at the bottom shows the command to run the script, which results in the output 'Task 3 Output: Cannot divide by zero'.

```

ASSIGNMENT 7.3.py X
ASSIGNMENT 7.3 > ASSIGNMENT 7.3.py > ...
1 def divide(a, b):
2     try:
3         return a / b
4     except ZeroDivisionError:
5         return "Cannot divide by zero"
6
7 print("\nTask 3 Output:", divide(10, 0))

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\PRANAY\OneDrive\Documents\AI ASSISTED> & 'c:\Users\PRANAY\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\PRANAY\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '53658' '--' 'C:\Users\PRANAY\OneDrive\Documents\AI ASSISTED\ASSIGNMENT 7.3\ASSIGNMENT 7.3.py'

Task 3 Output: Cannot divide by zero
PS C:\Users\PRANAY\OneDrive\Documents\AI ASSISTED>
  
```

4. Scenario

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

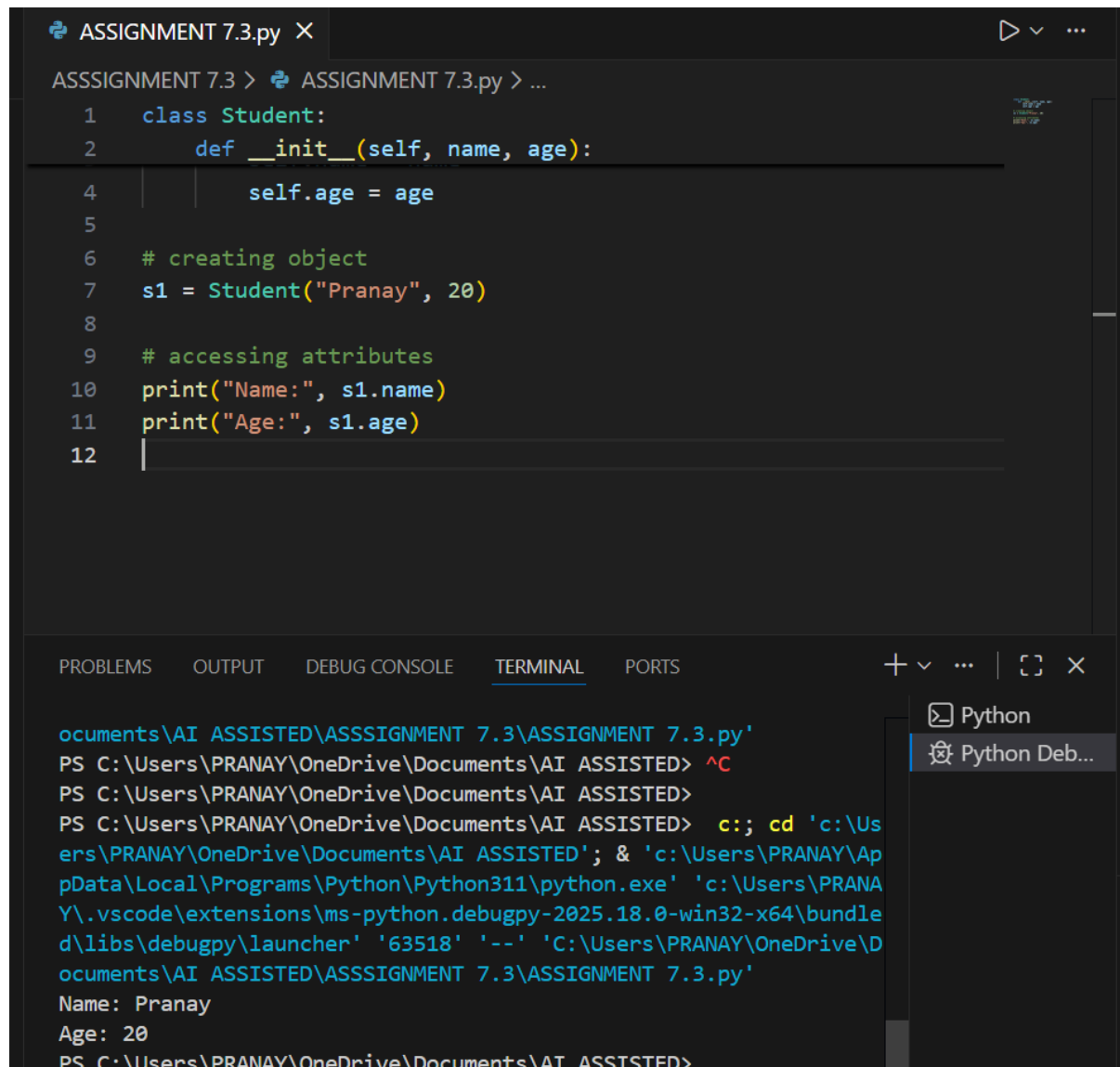
Requirements

- 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



The screenshot shows a VS Code editor window with a file named 'ASSIGNMENT 7.3.py'. The code defines a 'Student' class with an '__init__' method that takes 'name' and 'age' as arguments. It then creates an instance 's1' of the 'Student' class with 'Pranay' as the name and '20' as the age. Finally, it prints the name and age of 's1'.

```

1  class Student:
2      def __init__(self, name, age):
3
4          self.age = age
5
6      # creating object
7      s1 = Student("Pranay", 20)
8
9      # accessing attributes
10     print("Name:", s1.name)
11     print("Age:", s1.age)
12

```

The terminal output shows the command prompt running the script, followed by the printed output: 'Name: Pranay' and 'Age: 20'.

```

documents\AI ASSISTED\ASSIGNMENT 7.3\ASSIGNMENT 7.3.py'
PS C:\Users\PRANAY\OneDrive\Documents\AI ASSISTED> ^C
PS C:\Users\PRANAY\OneDrive\Documents\AI ASSISTED>
PS C:\Users\PRANAY\OneDrive\Documents\AI ASSISTED> c:; cd 'c:\Users\PRANAY\OneDrive\Documents\AI ASSISTED'; & 'c:\Users\PRANAY\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\PRANAY\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundle\d\libs\debugpy\launcher' '63518' '--' 'C:\Users\PRANAY\OneDrive\Documents\AI ASSISTED\ASSIGNMENT 7.3\ASSIGNMENT 7.3.py'
Name: Pranay
Age: 20
PS C:\Users\PRANAY\OneDrive\Documents\AI ASSISTED>

```

5. Task 5: Resolving Index Errors in Lists

Scenario

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

Requirements

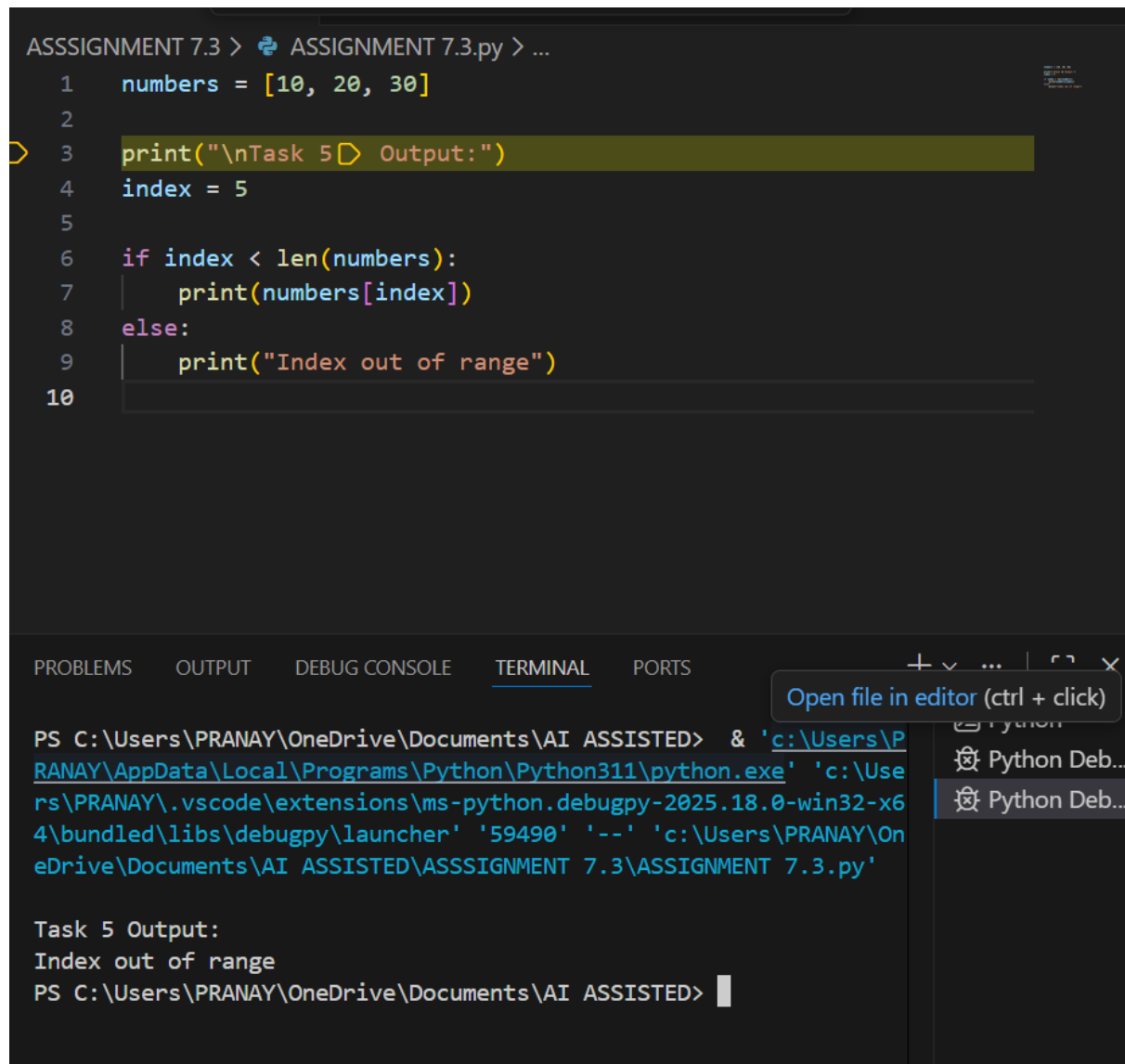
- Provide code that accesses an out-of-range list index
- Use AI to identify the Index Error
- Let AI suggest safe access methods
- Apply bounds checking or exception handling

Expected Output

- Index error resolved

Safe list access logic implemented

- AI suggestion using length checks or exception handling



The image shows a VS Code editor window with a Python file named 'ASSIGNMENT 7.3.py'. The code defines a list 'numbers' with values [10, 20, 30] and attempts to access 'numbers[5]'. A conditional check 'if index < len(numbers):' is implemented to handle the out-of-range error. The terminal output shows the command to run the script, followed by the printed output 'Task 5 Output: Index out of range'.

```
ASSIGNMENT 7.3 > ASSIGNMENT 7.3.py > ...
1  numbers = [10, 20, 30]
2
3  print("\nTask 5 Output:")
4  index = 5
5
6  if index < len(numbers):
7      print(numbers[index])
8  else:
9      print("Index out of range")
10
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\PRANAY\OneDrive\Documents\AI ASSISTED> & 'c:\Users\PRANAY\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\PRANAY\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundle\libs\debugpy\launcher' '59490' '--' 'c:\Users\PRANAY\OneDrive\Documents\AI ASSISTED\ASSIGNMENT 7.3\ASSIGNMENT 7.3.py'

Task 5 Output:
Index out of range
PS C:\Users\PRANAY\OneDrive\Documents\AI ASSISTED>