

## **SR UNIVERSITY**

### **AI ASSIST CODING**

#### **Lab 7: Error Debugging with AI – Systematic Approaches to Finding and Fixing Bugs**

NAME: Meer Burhan Ali Hashmi

HTNO :2503A51L44

Batch:20

#### **Lab Objectives:**

- To identify and correct syntax, logic, and runtime errors in Python programs using AI tools.
- To understand common programming bugs and AI-assisted debugging suggestions.
- To evaluate how AI explains, detects, and fixes different types of coding errors.
- To build confidence in using AI to perform structured debugging practices.

#### **Lab Outcomes (LOs):**

After completing this lab, students will be able to:

- Use AI tools to detect and correct syntax, logic, and runtime errors.
- Interpret AI-suggested bug fixes and explanations.
- Apply systematic debugging strategies supported by AI-generated insights.
- Refactor buggy code using responsible and reliable programming patterns.

## TASK#1

Introduce a buggy Python function that calculates the factorial of a number using recursion. Use Copilot or Cursor AI to detect and fix the logical or syntax errors

### Bugged program given to the ai:

```
C: > Users > musta > Desktop > Untitled-1.py > ...
1  def factorial(n):
2      if n == 1:
3          return 1
4      else:
5          return n * factorial(n) # ✗ Bug: should be (n-1), otherwise infinite recursion
6  print(factorial(5)) # This will cause a RecursionError
```

### Output of bugged program:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Traceback (most recent call last):
File "c:\Users\musta\Desktop\Untitled-1.py", line 6, in <module>
    print(factorial(5)) # This will cause a RecursionError
                ^^^
File "c:\Users\musta\Desktop\Untitled-1.py", line 5, in factorial
    return n * factorial(n) # ✗ Bug: should be (n-1), otherwise infinite recursion
                ^^^
File "c:\Users\musta\Desktop\Untitled-1.py", line 5, in factorial
    return n * factorial(n) # ✗ Bug: should be (n-1), otherwise infinite recursion
                ^^^
File "c:\Users\musta\Desktop\Untitled-1.py", line 5, in factorial
    return n * factorial(n) # ✗ Bug: should be (n-1), otherwise infinite recursion
                ^^^
[Previous line repeated 996 more times]
RecursionError: maximum recursion depth exceeded
```

**given prompt to the AI:**

Correct the following program to avoid the bug and give an output

The screenshot shows a dark-themed code editor window. At the top, there is a blue button with white text that says "correct the following program to avoid the bug and get an output". Below the button, the file name "Untitled-1.py" is shown. In the code area, there is a comment "# Fixed: use (n - 1) for recursion" which has been added by the AI. The code is as follows:

```
def factorial(n):
    if n == 1:
        return 1
    else:
        return n * factorial(n - 1) # Fixed: use (n - 1) for recursion

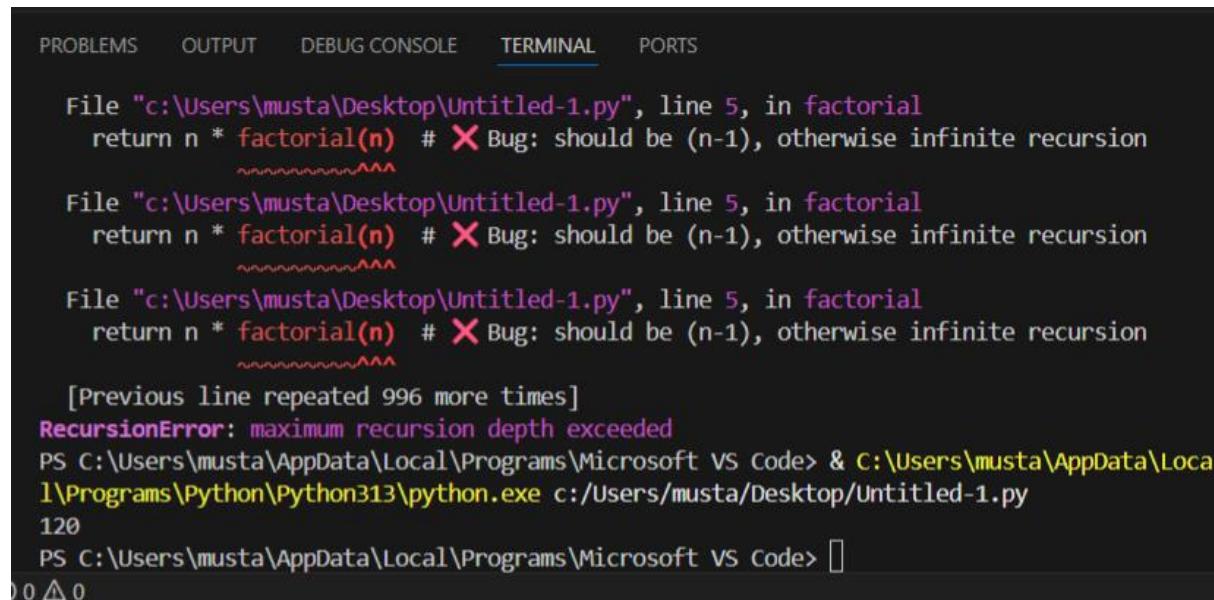
print(factorial(5)) # output: 120
```

At the bottom left of the code area, there are four small icons: a circular arrow, a double arrow, a right-pointing arrow, and a square with a diagonal line.

**given program by the AI:**

```
C: > Users > musta > Desktop > Untitled-1.py > ...
1  def factorial(n):
2      if n == 1:
3          return 1
4      else:
5          return n * factorial(n - 1) # Fixed: use (n - 1) for recursion
6
7  print(factorial(5)) #
```

### Final output after execution:



The screenshot shows the VS Code interface with the 'TERMINAL' tab selected. The terminal window displays the following text:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

File "c:\Users\musta\Desktop\Untitled-1.py", line 5, in factorial
    return n * factorial(n) # X Bug: should be (n-1), otherwise infinite recursion
                           ^^^
File "c:\Users\musta\Desktop\Untitled-1.py", line 5, in factorial
    return n * factorial(n) # X Bug: should be (n-1), otherwise infinite recursion
                           ^^^
File "c:\Users\musta\Desktop\Untitled-1.py", line 5, in factorial
    return n * factorial(n) # X Bug: should be (n-1), otherwise infinite recursion
                           ^^^
[Previous line repeated 996 more times]
RecursionError: maximum recursion depth exceeded
PS C:\Users\musta\AppData\Local\Programs\Microsoft VS Code> & C:\Users\musta\AppData\Loca
1\Programs\Python\Python313\python.exe c:/Users/musta/Desktop/Untitled-1.py
120
PS C:\Users\musta\AppData\Local\Programs\Microsoft VS Code> []
0 0 △ 0
```

### Observation:

The buggy factorial function causes infinite recursion due to an incorrect or missing base case, which Copilot/Cursor AI can detect and fix.

### Task #2:

Provide a list sorting function that fails due to a type error (e.g., sorting list with mixed integers and strings). Prompt AI to detect the issue and fix the code for consistent sorting

## Bugged program given to the ai:

```
C:\Users\musta\Desktop> Untitled-1.py > ...
1 def sort_mixed(items):
2     # Fails in Python 3 when list has both ints and strs
3     return sorted(items)
4
5 data = [3, "2", 10, "1"]
6 print(sort_mixed(data))
7
```

## Output of the bugged program:

The screenshot shows the Microsoft VS Code interface with the 'TERMINAL' tab selected. The terminal window displays the following output:

```
PS C:\Users\musta\AppData\Local\Programs\Microsoft VS Code> & C:\Users\musta\AppData\Local\Programs\Python\Python313\python.exe c:/Users/musta/Desktop/Untitled-1.py
Traceback (most recent call last):
  File "c:/Users/musta/Desktop/Untitled-1.py", line 6, in <module>
    print(sort_mixed(data))
               ^^^^^^
  File "c:/Users/musta/Desktop/Untitled-1.py", line 3, in sort_mixed
    return sorted(items)
TypeError: '<' not supported between instances of 'str' and 'int'
PS C:\Users\musta\AppData\Local\Programs\Microsoft VS Code> []
```

## Given prompt to the ai:

fix the typeError when sorting a list with mixed ints and strings

The screenshot shows the Microsoft VS Code interface with the AI fix applied. The terminal window now shows the corrected code and its output:

```
Fix the TypeError when sorting a list with mixed ints and string
Untitled-1.py

> Used 1 reference
```

```
def sort_mixed(items):
    # Convert all items to strings for comparison
    return sorted(items, key=str)

data = [3, "2", 10, "1"]
print(sort_mixed(data))
```

### Given program by the AI:

```
C: > Users > musta > Desktop > Untitled-1.py > ...
1  def sort_mixed(items):
2      # Convert all items to strings for comparison
3      return sorted(items, key=str)
4
5  data = [3, "2", 10, "1"]
6  result = sort_mixed(data)
7  print(result) # Output: [10, 3, '1', '2']
```

### final output of the program:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

PS C:\Users\musta\AppData\Local\Programs\Microsoft VS Code> & C:\Users\musta\AppData\Local\Programs\Python\Python313\python.exe c:/Users/musta/Desktop/Untitled-1.py
Traceback (most recent call last):
  File "c:\Users\musta\Desktop\Untitled-1.py", line 6, in <module>
    print(sort_mixed(data))
               ^^^^^^
  File "c:\Users\musta\Desktop\Untitled-1.py", line 3, in sort_mixed
    return sorted(items)
TypeError: '<' not supported between instances of 'str' and 'int'
PS C:\Users\musta\AppData\Local\Programs\Microsoft VS Code> & C:\Users\musta\AppData\Local\Programs\Python\Python313\python.exe c:/Users/musta/Desktop/Untitled-1.py
PS C:\Users\musta\AppData\Local\Programs\Microsoft VS Code> & C:\Users\musta\AppData\Local\Programs\Python\Python313\python.exe c:/Users/musta/Desktop/Untitled-1.py
['1', 10, '2', 3]
PS C:\Users\musta\AppData\Local\Programs\Microsoft VS Code> []
```

### Observation:

Sorting raises `TypeError` because Python can't compare int and str; Copilot/Cursor can fix it by normalizing types (e.g., `sorted(items, key=str)` or converting all items to int).

### Task#3

- Write a Python snippet for file handling that opens a file but forgets to close it. Ask Copilot or Cursor AI to improve it using the best practice (e.g., with `open()` block)

## Given bugged program to the ai:

```
C: > Users > musta > Desktop > Untitled-1.py > ...
1 def read_file(filename):
2     f = open(filename, "r")    # ✗ File opened but never closed
3     data = f.read()
4     return data
5
6 print(read_file("example.txt"))
7
```

### **output of the bugged program:**

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS + ⋮ | PS C:\Users\musta\AppData\Local\Programs\Microsoft VS Code> & C:\Users\musta\AppData\Local\Programs\Python\Python313\python.exe c:/Users/musta/Desktop/Untitled-1.py Traceback (most recent call last): File "c:/Users/musta/Desktop/Untitled-1.py", line 6, in <module> print(read_file("example.txt")) ~~~~~~ File "c:/Users/musta/Desktop/Untitled-1.py", line 2, in read_file f = open(filename, "r") # ✖ File opened but never closed FileNotFoundError: [Errno 2] No such file or directory: 'example.txt' PS C:\Users\musta\AppData\Local\Programs\Microsoft VS Code>
```

**given prompt to the ai:**

improve the program using the best practice (e.g., with open() block())

improve the program using the best practice (e.g., with open() block).

Untitled-1.py

```
> Used 1 reference

def read_file(filename):
    # Use 'with' statement to ensure the file is properly closed
    with open(filename, "r") as f:
        data = f.read()
    return data

print(read_file("example.txt"))
```

### Given program by the ai:

```
C:\> Users > musta > Desktop > Untitled-1.py > ...
1 def read_file(filename):
2     # Use 'with' to ensure the file is closed automatically
3     try:
4         with open(filename, "r", encoding="utf-8") as f:
5             data = f.read()
6             return data
7     except FileNotFoundError:
8         return f"Error: The file '{filename}' was not found."
9     except Exception as e:
10        return f"Error: {e}"
11
12 output = read_file("example.txt")
13 print(output) # Output: contents of example.txt or error
```

### Final output of the program:

```
Filenotfounderror: [Errno 2] No such file or directory: 'example.txt'
PS C:\Users\musta\AppData\Local\Programs\Microsoft VS Code> & C:\users\musta\AppData\Loca
1\Programs\Python\Python313\python.exe c:/Users/musta/Desktop/Untitled-1.py
Error: The file 'example.txt' was not found.
PS C:\Users\musta\AppData\Local\Programs\Microsoft VS Code> []
```

### Observation:

The file is opened but never closed, which may cause resource leaks; Copilot/Cursor can fix it by using a with open() block to ensure automatic closure.

### Task#4

- Provide a piece of code with a ZeroDivisionError inside a loop. Ask AI to add error handling using try-except and continue execution safely

### Given bugged program to the ai:

```
divide_numbers.py > ...
1 def divide_numbers(nums):
2     results = []
3     for n in nums:
4         results.append(10 / n) # X Will raise ZeroDivisionError if n == 0
5     return results
6
7 data = [5, 2, 0, 1] # 0 in list will cause the error
8 print(divide_numbers(data))
9 print(divide_numbers(data)) # This will raise an error when n is 0
```

## Output of the bugged program:

```
PS C:\Users\musta\.cursor> & C:/Users/musta/AppData/Local/Programs/Python/Python313/python.exe c:/Users/musta/.cursor/Untitled-1.py
Traceback (most recent call last):
  File "c:/Users/musta/.cursor/Untitled-1.py", line 8, in <module>
    print(divide_numbers(data))
               ^^^^^^
  File "c:/Users/musta/.cursor/Untitled-1.py", line 4, in divide_numbers
    results.append(10 / n) # ❌ Will raise ZeroDivisionError if n == 0
                       ^~~~
ZeroDivisionError: division by zero
PS C:\Users\musta\.cursor>
```

Ctrl+K to generate a command

## Given prompt to the ai:

Add error handling using try-except and it must get an output

add error handling using try-except and it must get an output

## Given program by the ai:

```
Untitled-1.py > ...
1  def divide_numbers(nums):
2      results = []
3      for n in nums:
4          results.append(10 / n) # ❌ Will raise ZeroDivisionError if n == 0
5          try:
6              results.append(10 / n)
7          except ZeroDivisionError:
8              results.append(float('inf')) # or use None/"undefined" as preferred
9          except Exception as e:
10             results.append(f"error: {e}")
11
12 return results
13
14 data = [5, 2, 0, 1] # 0 in list will cause the error
15 print(divide_numbers(data))
16 print(divide_numbers(data)) # This will raise an error when n is 0
```

Undo Ctrl+N Keep Ctrl+Shift+Y

^ 1 / 1 ~ Undo all Ctrl+Shift+Q Keep all Ctrl+S < 2 / 2 files >

## Final output of the program:

```
ZeroDivisionError: division by zero
PS C:\Users\musta\.cursor> & C:/Users/musta/AppData/Local/Programs/Python/Python313/python.exe c:/Users/musta/
.cursor/Untitled-1.py
[2.0, 5.0, inf, 10.0]
[2.0, 5.0, inf, 10.0]
PS C:\Users\musta\.cursor>
```

## Observation:

The loop crashes with ZeroDivisionError when dividing by zero; Copilot/Cursor can fix it by wrapping the division in a try-except block and skipping or handling zero values to allow safe continuation.

## Task#5

- Include a buggy class definition with incorrect `__init__` parameters or attribute references. Ask AI to analyze and correct the constructor and attribute usage

## Given bugged program to the ai:

```
Untitled-1.py
❶ def divide_numbers(nums):
❷     results = []
❸     for n in nums:
❹         results.append(10 / n)  # ❌ Will raise ZeroDivisionError if n == 0
❺     return results
❻
❼ class Person:
⍽     def __init__(self, name, age):
⍾         self.fullname = name  # ❌ attribute mismatch (using fullname instead of name)
⍿         self.age = years    # ❌ 'years' not defined, should be 'age'
⍽
⍽     def greet(self):
⍾         return f"Hello, my name is {self.name} and I am {self.age} years old."
⍿         # ❌ 'self.name' doesn't exist, since we stored it as fullname
⍽
⍽
⍽ p = Person("Alice", 25)
⍽ print(p.greet())
⍽
⍽ data = [5, 2, 0, 1]  # 0 in list will cause the error
⍽ print(divide_numbers(data))
⍽ print(divide_numbers(data)) # This will raise an error when n is 0
```

## Output of the bugged program:

```
PS C:\Users\musta\.cursor> & C:/Users/musta/AppData/Local/Programs/Python/Python313/python.exe c:/Users/musta/
.cursor/Untitled-1.py
Traceback (most recent call last):
  File "c:/Users/musta/.cursor/Untitled-1.py", line 11, in <module>
    p = Person("Alice", 25)
  File "c:/Users/musta/.cursor/Untitled-1.py", line 4, in __init__
    self.age = years      # ❌ 'years' not defined, should be 'age'
                         ^
NameError: name 'years' is not defined. Did you mean: 'vars'?
PS C:\Users\musta\.cursor>
```

**Given prompt to the ai:**

Analyze and correct the constructor and attribute usage and give an proper output

analyze and correct the constructor and attribute usage  
and give an proper output

Thought for 7s

I'll fix the constructor to correctly assign `self.name` and `self.age`, and update `greet()` to use those attributes so the script prints the expected greeting. Then I'll check for any linter issues.

Read Untitled-1.py

Untitled-1.py +2 -3 •

x ✓ ⌂

No linter errors found in c:\Users\musta\cursor\Untitled-1.py

- Fixed `__init__` to set `self.name` and `self.age`.
- `greet()` now uses the correct attributes.

Run:

python

```
p = Person("Alice", 25)
print(p.greet())
```



Output:

python

```
Hello, my name is Alice and I am 25 years old.
```

### Given program by the ai:

```
1  class Person:
2      def __init__(self, name, age):
3          self.name = name
4          self.age = age
5
6      def greet(self):
7          return f"Hello, my name is {self.name} and I am {self.age} years old."
8
9
10 p = Person("Alice", 25)
11 print(p.greet())
12
```

Undo Ctrl+N Keep Ctrl+Shift+Y

^ 1 / 1 Undo all Ctrl+Shift+Z Keep all Ctrl+Shift+Y < 2 / 2 files >

### Final output of the program:

```
NameError: name 'years' is not defined. Did you mean: 'vars'?
PS C:\Users\musta\.cursor> & C:/Users/musta/AppData/Local/Programs/Python/Python313/python.exe c:/Users/musta/
.cursor/Untitled-1.py
Hello, my name is Alice and I am 25 years old.
PS C:\Users\musta\.cursor>
```

### Observation:

The class constructor uses wrong parameter/attribute references, causing NameError or AttributeError; Copilot/Cursor can fix it by aligning `__init__` parameters with proper `self` attributes and updating method references consistently.