

Assignment - 7.3

P.Naga Shiva Chaitanya
2303A51945
B - 27

Task 1: Fixing Syntax Errors

Buggy Code (Before Debugging):

```
def add(a, b)
    return a + b
print(add(3, 5))
```

The screenshot shows a Python code editor in VS Code. On the left, a file named `Lab_7_3.py` is open, containing the following code:

```
def add(a, b)
    return a + b
print(add(3, 5))
```

A syntax error is highlighted at the end of the first line of the function definition. The terminal on the right shows the following output:

```
PS C:\Users\nagas\OneDrive\Desktop\bobby\python> & C:/Python314/python.exe c:/Users/nagas/OneDrive/Desktop/bobby/python/Lab_7_3.py
File "c:/Users/nagas/OneDrive/Desktop/bobby/python/Lab_7_3.py", line 1
  def add(a, b)
          ^
SyntaxError: expected ':'
PS C:\Users\nagas\OneDrive\Desktop\bobby\python>
```

An error explanation request is open on the right side of the interface. It says:

← ERROR EXPLANATION REQUEST

explain the error

✓ Planning error explanation
I'll open `Lab_7_3.py` to inspect the code and reproduce the error.

✓ Identified syntax error and planned resolution
Creating a short todo list for inspecting, explaining, and optionally fixing the error.

✓ Managed and updated to-do items
Updating the todo list to mark the explanation step complete.

✓ Completed: Explain the error (2/3)

- **Error:** SyntaxError (invalid syntax) in `Lab_7_3.py:1-2`.
- **Cause:** The function definition is missing a colon after the parameter list: `def add(a, b)` should end with `:`.
- **Fix:** Change to:
 - `def add(a, b):`
 `return a + b`
- **Next:** I can apply this fix to `Lab_7_3.py:1-2` if you want.

AI-Detected Issue

- Missing colon (`:`) at the end of the function definition.
- This causes a **SyntaxError**.

Corrected Code (After Debugging):

```
def add(a, b):
    return a + b
print(add(3, 5))
```

The screenshot shows a code editor interface with a dark theme. In the top-left, there's a file icon followed by "Lab_7_3.py > ...". Below it is the Python code:

```
1 def add(a, b):
2     return a + b
3 print(add(3, 5))
4
```

Below the code editor is a navigation bar with tabs: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and others. The TERMINAL tab is underlined, indicating it is active. To the right of the tabs are icons for Python, a plus sign, a dropdown arrow, a trash can, and more.

The terminal window displays the command-line output:

```
PS C:\Users\nagas\OneDrive\Desktop\bobby\python> & C:/Python314/python.exe c:/Users/nagas/OneDrive/Desktop/bobby/python/Lab_7_3.py
8
PS C:\Users\nagas\OneDrive\Desktop\bobby\python>
```

AI Explanation

In Python, every function definition must end with a colon.
The missing colon caused the syntax error. Adding it fixes the issue.

Task 2: Debugging Logic Errors in Loops

Buggy Code (Before Debugging):

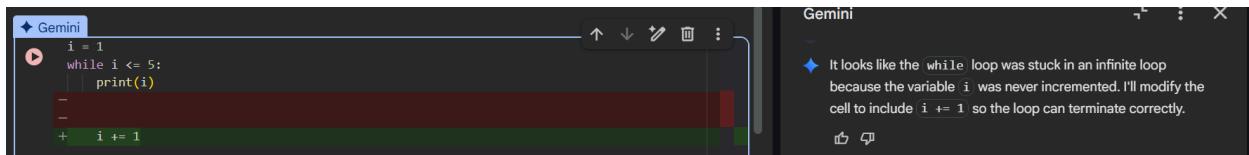
```
▶ i = 1
  while i <= 5:
    print(i)
```

```
... 1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
```

```
KeyboardInterrupt                                     Traceback (most recent call last)
/tmp/ipython-input-1792165919.py in <cell line: 0>()
      1 i = 1
      2 while i <= 5:
----> 3     print(i)
      4

----- 1 frames -----
/usr/local/lib/python3.12/dist-packages/ipykernel/iostream.py in _is_master_process(self)
 436
 437     def _is_master_process(self):
--> 438         return os.getpid() == self._master_pid
 439
 440     def set_parent(self, parent):
```

KeyboardInterrupt:



AI-Detected Issue

- The loop variable `i` is never incremented.
- This causes an **infinite loop**.

Corrected Code (After Debugging)

The screenshot shows the Gemini interface. At the top, there's a toolbar with a play button and a 'Gemini' logo. Below the toolbar, a code editor window displays the following Python code:

```
i = 1
while i <= 5:
    print(i)
    i += 1
```

Below the code editor, the output window shows the numbers 1 through 5, each preceded by an ellipsis (...).

Task 3: Handling Runtime Errors (Division by Zero)

Buggy Code (Before Debugging):

The screenshot shows the Gemini interface. On the left, a code editor window contains the following Python code:

```
def divide(a, b):
    return a / b
print(divide(10,0))
```

When run, it triggers a `ZeroDivisionError`. The right side of the interface shows a panel titled "Gemini" with the following text:

P def divide(a, b): return a / b print(divide(10,0))
◆ The code you provided would result in a `ZeroDivisionError` when `divide(10,0)` is called. I've created a new cell with an updated `divide` function that handles this error gracefully, and then called it with your example values.

At the bottom, there are buttons for "Accept & Run", "Accept", and "Cancel". A sidebar at the bottom right asks "What can I help you build?" with a "+" button.

AI-Detected Issue

- If `b` is zero, the program crashes with `ZeroDivisionError`.

Corrected Code (After Debugging):

```
6] 0s
▶ def divide(a, b):
    try:
        return a / b
    except ZeroDivisionError:
        return "Error: Cannot divide by zero"

print(divide(10,0))
...
... Error: Cannot divide by zero
```

AI Explanation

The AI used a `try-except` block to handle runtime errors safely. This prevents the program from crashing.

Task 4: Debugging Class Definition Errors

Buggy Code (Before Debugging):

The screenshot shows the Gemini IDE interface. On the left, a code editor displays Python code for a `Student` class. The code includes two `__init__` methods and a `print` statement. A red box highlights the first `__init__` method. On the right, a detailed error message is shown, explaining a `TypeError` related to the number of arguments. Below the error message is a text input field asking "What can I help you build?". At the bottom, there's a note about Gemini making mistakes and a link to learn more.

Next steps: [Explain error](#)

gemini

```
class Student:  
    def __init__(name, age):  
        name = name  
        age = age  
    def __init__(self, name, age):  
        self.name = name  
        self.age = age  
s = Student("Rahul", 20)  
print(s.name, s.age)
```

...

TypeError Traceback (most recent call last)
/tmp/ipython-input-1522557614.py in <cell line: 0>()
 3 name = name
 4 age = age
----> 5 s = Student("Rahul", 20)
 6 print(s.name, s.age)

TypeError: Student.__init__() takes 2 positional arguments but 3 were given

Gemini

① `TypeError: Student.__init__() takes 2 positional arg`

◆ The error `TypeError: Student.__init__() takes 2 positional arguments but 3 were given` means that the `__init__` method was called with more arguments than it was defined to accept. This is because all instance methods in Python, including `__init__`, automatically receive the instance itself as the first argument, conventionally named `self`. Your `__init__` method was missing this `self` parameter. Also, the `name` and `age` parameters were not correctly assigned to the `Student` object's attributes. I'll correct the `__init__` method to properly define `self` and assign the values.

What can I help you build?

Gemini 2.5 Flash

Gemini can make mistakes so double-check it and use code with caution. Learn more

AI-Detected Issue

- Missing `self` parameter in the constructor.
 - Instance variables are not assigned correctly.

Corrected Code (After Debugging):

◆ Gemini

```
▶ class Student:  
    def __init__(self, name, age):  
        self.name = name  
        self.age = age  
    s = Student("Rahul", 20)  
    print(s.name, s.age)  
... Rahul 20
```

Task 5: Resolving Index Errors in Lists

Buggy Code (Before Debugging):

The screenshot shows a Jupyter Notebook cell with the following code:

```
▶ numbers = [10, 20, 30]  
print(numbers[5])  
...  
IndexError: list index out of range
```

The output area shows the error message:

```
Traceback (most recent call last)  
/tmp/ipython-input-2134759640.py in <cell line: 0>()  
      1 numbers = [10, 20, 30]  
----> 2 print(numbers[5])  
  
IndexError: list index out of range
```

Next steps: Explain error

The right panel displays AI-generated feedback:

P Please explain this error:
ⓘ IndexError: list index out of range

◆ The error `IndexError: list index out of range` occurs because you are trying to access the element at index 5 in the `numbers` list, but the list only has three elements (at indices 0, 1, and 2). To fix this, you should use a valid index. For example, if you want to print the last element, you would use `numbers[2]`.

AI-Detected Issue

- Index 5 does not exist in the list.
- Causes an `IndexError`.

Corrected Code (After Debugging):

The screenshot shows a Jupyter Notebook cell with the following code:

```
s▶ numbers = [10, 20, 30]  
print(numbers[0])  
... 10
```

The right panel displays AI-generated feedback:

P Print a valid index from the numbers list

◆ Certainly! I'll modify the existing cell to print an element using a valid index. For example, here's how to print the first element:
The code executed successfully, and as requested, it printed the element at index 0 from the `numbers` list, which is 10.

