

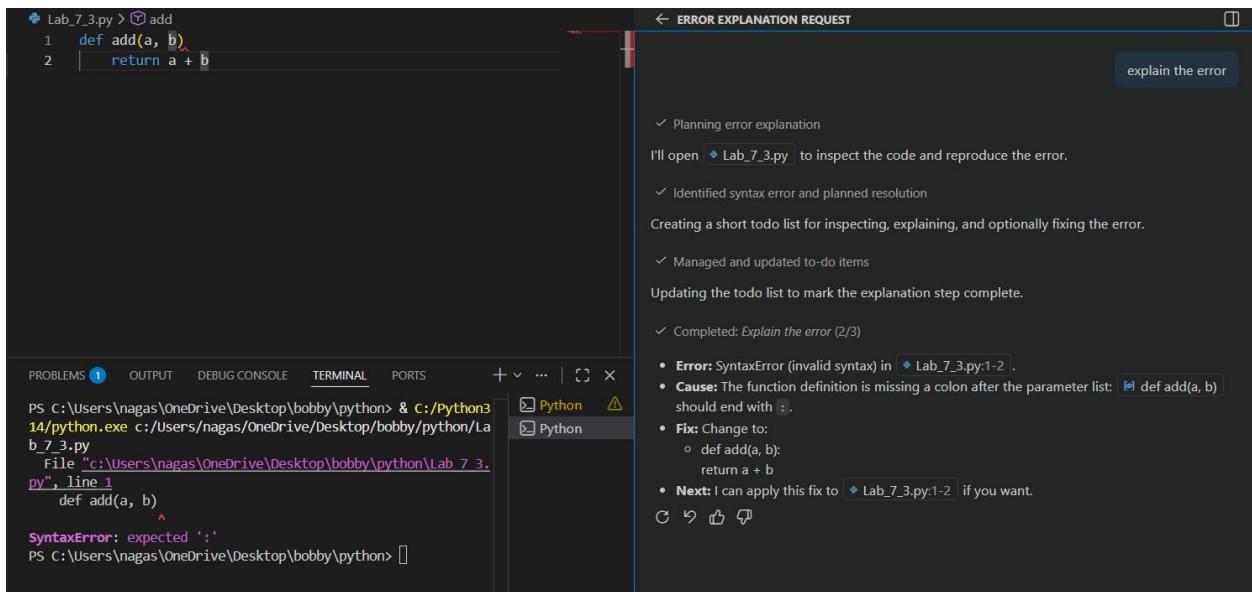
# Assignment - 7.3

A.Nikhil  
2303A51934  
B - 27

## Task 1: Fixing Syntax Errors

Buggy Code (Before Debugging):

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



## 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. A file named 'Lab\_7\_3.py' is open, containing the following code:

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

The code editor has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and other options. The TERMINAL tab is active, showing the following terminal 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>
```

A red box highlights the line 'print(add(3, 5))' in the code editor, indicating a syntax error.

## 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):**

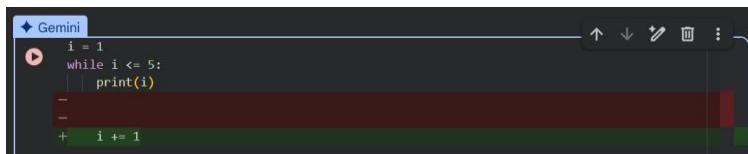
```
▶ i = 1
  while i <= 5:
    print(i)
```

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

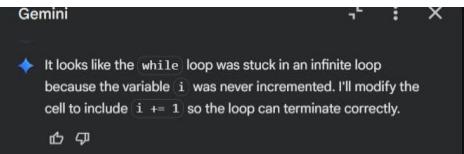
```
KeyboardInterrupt                                     Traceback (most recent call last)
/tmp/ipython-input-1792165919.py in <cell line: 0>()
      1 i = 1
      2 while i <= 5:
----> 3     print(i)
      4

----- 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:



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



Gemini

◆ It looks like the `while` loop was stuck in an infinite loop because the variable `i` was never incremented. I'll modify the cell to include `i += 1` so the loop can terminate correctly.

Copy

## AI-Detected Issue

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

## Corrected Code (After Debugging)

The screenshot shows the Gemini AI interface. In the top bar, there's a play button icon and the text "Gemini". Below the interface, a code editor window displays the following Python code:

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

Below the code, the output is shown as:

```
... 1
2
3
4
5
```

## Task 3: Handling Runtime Errors (Division by Zero) Buggy

### Code (Before Debugging):

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

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

When run, it produces the following traceback:

```
...ZeroDivisionError: division by zero
```

To the right of the notebook, the Gemini AI interface is open. It shows the same code and includes a message from Gemini:

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.

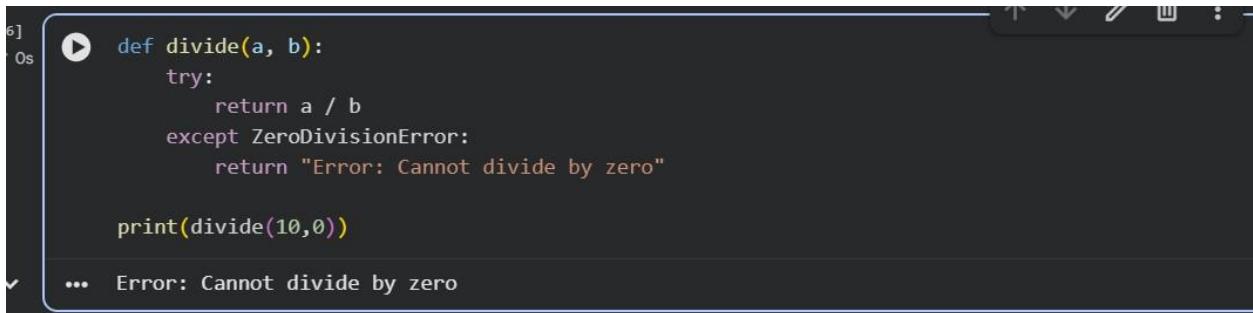
Accept & Run    Accept    Cancel

What can I help you build?

### 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
```

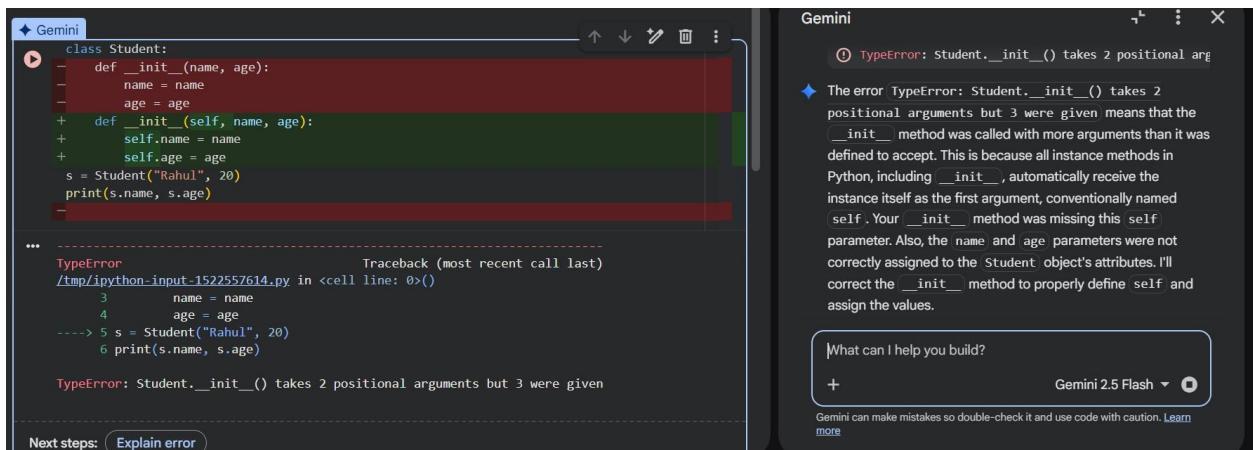
## AI Explanation

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):



```
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

Next steps: Explain error
```

Gemini

① `TypeError: student.__init__() takes 2 positional arguments but 3 were given`

◆ 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

### AI-Detected Issue

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

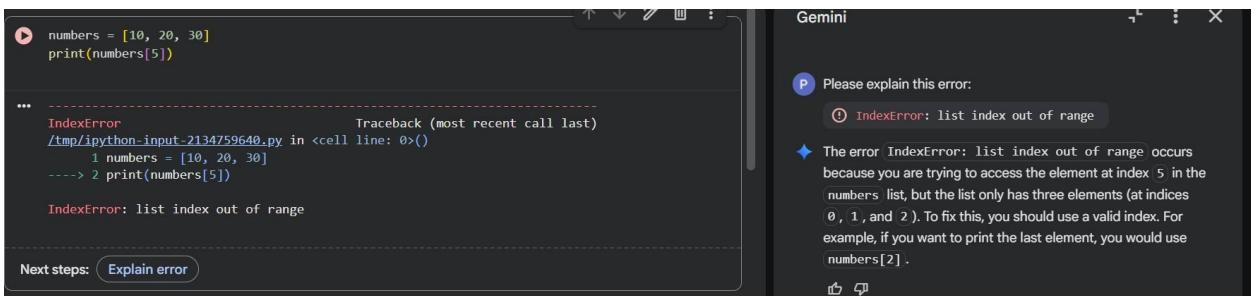
## Corrected Code (After Debugging):

◆ 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):



```
numbers = [10, 20, 30]  
print(numbers[5])  
  
...  
IndexError 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

Gemini

P Please explain this error:

① 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 interface. On the left, a code cell contains the following Python code:

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

The output cell below it shows the result: `10`. On the right, a Gemini AI interface displays a conversation:

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.