

Assignment 7.3 Ai Assisted Coding

Htno:2303A510C0

Btno:06

Task 1: Fixing Syntax Errors

Prompt: The following Python function has a syntax error. Identify the issue and correct it. Also explain what the syntax error is.

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

return a + b Input:

Bug Code:

The screenshot shows a Google Colab notebook titled "Untitled29.ipynb". In cell [13], there is a code snippet:`[13] 0s
def add(a, b)
 return a + b`

... File "/tmp/ipython-input-676827692.py", line 1
 def add(a, b)
 ^
SyntaxError: expected ':'

```
Next steps: Explain error
```

On the right side of the screen, there is a "Release notes" panel with the following content:

Please follow our [blog](#) to see more information about new features, tips and tricks, and featured notebooks such as [Analyzing a Bank Failure with Colab](#).

2026-01-20

- Launched Data Explorer - a new feature that lets you search Kaggle datasets, models, and competitions directly from a Colab notebook!
- Gemini 3 is now available in Colab.
- In addition to being available in VS Code, Colab is also available for use in Antigravity, Cursor, and Windsurf via the Open VSX Registry!
- H100 is being rolled out for more users.
- Launched a new modern design for the Colab UI.

Python package upgrades

- accelerate 1.11.0 -> 1.12.0
- astropy 7.1.1 -> 7.2.0
- bigframes 2.28.0 -> 2.31.0
- cachetools 5.5.2 -> 6.2.4

Variables Terminal 10:26 AM Python 3

2) corrected code:

The screenshot shows a Google Colab notebook titled "Untitled29.ipynb". In the code cell, there is a function definition:[14]:
def add(a, b):
 return a + b

Example usage:
result = add(10, 20)
print(f"The sum is: {result}")
... The sum is: 30

Below the code cell, there are three buttons: "+ Code", "+ Text", and "Add text cell". The output cell shows the result of the function call:The sum is: 30

In the top right corner, there is a "Release notes" section with a timestamp of "2026-01-20". The notes mention the launch of Data Explorer, the availability of Gemini 3, and several Python package upgrades.

Output:

The screenshot shows the browser window displaying the output of the Python code execution. The output cell contains the text:The sum is: 30

Explanation:

- In Python, a colon : is required after defining a function header.
- Without the colon, Python cannot recognize the start of the function block, causing a **SyntaxError**.
- AI correctly identified the missing colon and fixed the function definition.

Task 2: Debugging Logic Errors in Loops

Prompt: The following Python loop runs infinitely. Identify the logic error, correct the loop, and explain the issue.

```
i = 1 while i
```

```
<= 5:
```

```
    print(i)
```

```
i -= 1
```

Input: Bug code:

The screenshot shows a Google Colab notebook titled "Untitled29.ipynb". The code cell contains the following Python code:

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

A tooltip appears over the last line of the code, highlighting the line `i -= 1` and stating: "# Wrong update causing infinite loop". The notebook also displays a list of recent file versions on the left side.

On the right side of the screen, there is a "Release notes" panel for January 2026, which includes a list of changes such as the launch of Data Explorer and updates to various Python packages like accelerate, astropy, bigframes, and cachetools.

Corrected code:

A screenshot of a web browser window showing a Google Colab notebook titled "Untitled29.ipynb". The code cell [16] contains the following Python script:

```
i = 1
while i <= 5:
    print(i)
    i += 1 # Corrected: increment i instead of decrementing

print("Loop finished.")
```

The output pane shows the results of the loop:

```
1
2
3
```

Output:

A screenshot of a web browser window showing a Google Colab notebook titled "Untitled29.ipynb". The code cell [16] contains the following Python script:

```
print( Loop tnisnea. )
```

The output pane shows the results of the loop:

```
1
2
3
4
5
Loop finished.
```

Explanation: The variable `i` was decreasing (`i -= 1`) while the condition required it to increase, causing an infinite loop.

Changing it to `i += 1` allows the loop to reach the stopping condition and terminate correctly.

Task 3: Handling Runtime Errors (Division by Zero)

Prompt: This Python code causes a runtime error. Identify the problem, fix it using `tryexcept`, and explain the issue. `def divide(a, b): return a / b` `print(divide(10, 0))`

Input: Bug Code

The screenshot shows a Google Colab notebook titled "Untitled29.ipynb". In cell [17], there is a Python code snippet:

```
def divide(a, b):
    return a / b

print(divide(10, 0))
```

The output of the cell is:

```
ZeroDivisionError: division by zero
```

A tooltip message is displayed in the top right corner:

Enable browser notifications in Settings to get alerts when executions complete

OK No thanks

Below the tooltip, a news card is visible:

2026-01-20

- Launched Data Explorer - a new feature that lets you search Kaggle datasets, models, and competitions directly from a Colab notebook!
- Gemini 3 is now available in Colab.
- In addition to being available in VS Code, Colab is also available for use in Antigravity, Cursor, and Windsurf via the Open VSX Registry!
- H100 is being rolled out for more users.
- Launched a new modern design for the Colab UI.

Python package upgrades

- accelerate 1.11.0 -> 1.12.0
- astropy 7.1.1 -> 7.2.0
- bigframes 2.28.0 -> 2.31.0
- cachetools 5.5.2 -> 6.2.4

Variables Terminal 10:36 AM Python 3

Corrected Code:

The screenshot shows a Google Colab notebook titled "Untitled29.ipynb". In cell [18], there is a Python code snippet with error handling:

```
def divide(a, b):
    try:
        return a / b
    except ZeroDivisionError:
        print("Error: Cannot divide by zero!")
        return None # Return None or another appropriate value to indicate failure

# Example usage:
print("Attempting to divide 10 by 2:")
result1 = divide(10, 2)
if result1 is not None:
    print(f"Result: {result1}")

print("\nAttempting to divide 10 by 0:")
result2 = divide(10, 0)
if result2 is not None:
    print(f"Result: {result2}")
```

The output of the cell is:

```
Attempting to divide 10 by 2:
Result: 5.0

Attempting to divide 10 by 0:
Error: Cannot divide by zero!
```

A tooltip message is displayed in the top right corner:

Enable browser notifications in Settings to get alerts when executions complete

OK No thanks

Below the tooltip, a news card is visible:

2026-01-20

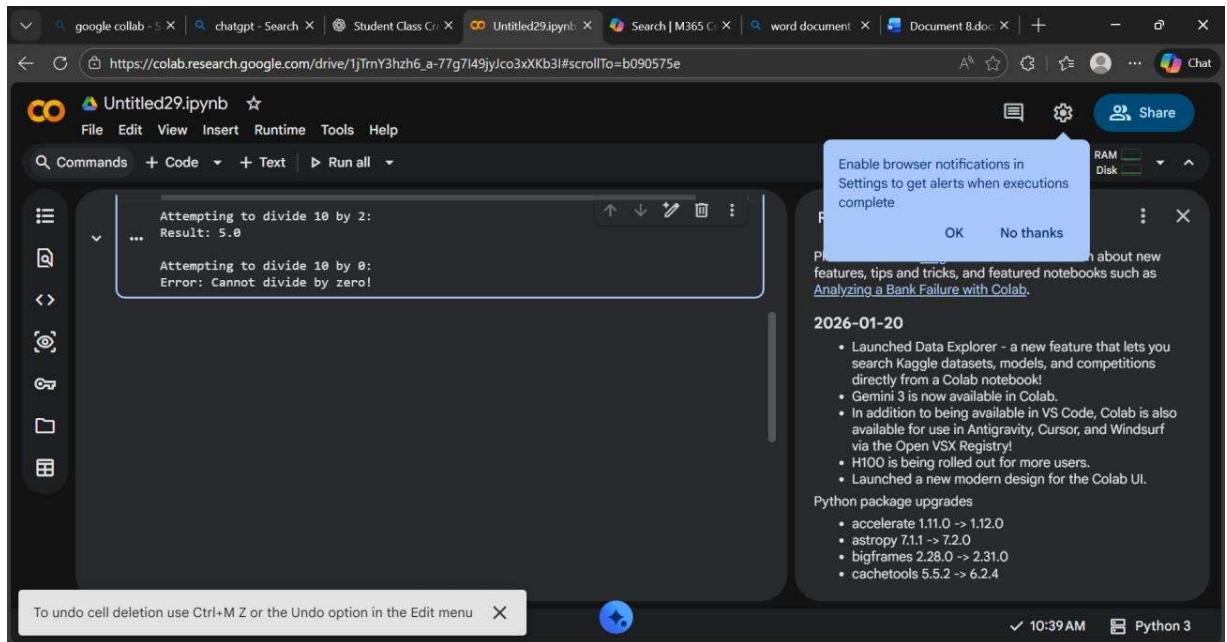
- Launched Data Explorer - a new feature that lets you search Kaggle datasets, models, and competitions directly from a Colab notebook!
- Gemini 3 is now available in Colab.
- In addition to being available in VS Code, Colab is also available for use in Antigravity, Cursor, and Windsurf via the Open VSX Registry!
- H100 is being rolled out for more users.
- Launched a new modern design for the Colab UI.

Python package upgrades

- accelerate 1.11.0 -> 1.12.0
- astropy 7.1.1 -> 7.2.0
- bigframes 2.28.0 -> 2.31.0
- cachetools 5.5.2 -> 6.2.4

Variables Terminal ✓ 10:39 AM Python 3

Output:



Explanation: the program crashes because division by zero is not allowed in Python, causing a `ZeroDivisionError`.

Using `try-except` prevents the crash and safely handles the error.

Task 4: Debugging Class Definition Errors

Prompt: The following Python class has an error in the constructor. Identify the issue, correct the class definition, and explain why the fix is needed.

```
class Student: def __init__(name, roll): name = name roll = roll
```

Input: Bug Code

Untitled29.ipynb

```
[20] 0s
class Student:
    def __init__(name, roll):
        name = name
        roll = roll
```

Enable browser notifications in Settings to get alerts when executions complete

OK No thanks

2026-01-20

- Launched Data Explorer - a new feature that lets you search Kaggle datasets, models, and competitions directly from a Colab notebook!
- Gemini 3 is now available in Colab.
- In addition to being available in VS Code, Colab is also available for use in Antigravity, Cursor, and Windsurf via the Open VSX Registry!
- H100 is being rolled out for more users.
- Launched a new modern design for the Colab UI.

Python package upgrades

- accelerate 1.11.0 → 1.12.0
- astropy 7.1.1 → 7.2.0
- bigframes 2.28.0 → 2.31.0
- cachetools 5.5.2 → 6.2.4

Variables Terminal ✓ 10:42 AM Python 3

Corrected code:

Untitled29.ipynb

```
[21] 0s
class Student:
    # Corrected constructor: 'self' is the first parameter
    def __init__(self, name, roll):
        self.name = name # Assign 'name' to the instance's 'name' attribute
        self.roll = roll # Assign 'roll' to the instance's 'roll' attribute

    def display_student_info(self):
        print(f"Student Name: {self.name}, Roll Number: {self.roll}")

    # Example usage:
    student1 = Student("Alice", 101)
    student1.display_student_info()

    student2 = Student("Bob", 102)
    student2.display_student_info()
```

Enable browser notifications in Settings to get alerts when executions complete

2026-01-20

- Launched Data Explorer - a new feature that lets you search Kaggle datasets, models, and competitions directly from a Colab notebook!
- Gemini 3 is now available in Colab.
- In addition to being available in VS Code, Colab is also available for use in Antigravity, Cursor, and Windsurf via the Open VSX Registry!
- H100 is being rolled out for more users.
- Launched a new modern design for the Colab UI.

Python package upgrades

Output:

The screenshot shows a Google Colab notebook titled "Untitled29.ipynb". In the code cell, the following Python code is run:

```
class Student:
    def __init__(self, name, roll):
        self.name = name
        self.roll = roll
    def __str__(self):
        return f"Student Name: {self.name}, Roll Number: {self.roll}"
s1 = Student("Alice", 101)
s2 = Student("Bob", 102)
print(s1)
print(s2)
```

The output of the code is displayed in the cell:

```
Student Name: Alice, Roll Number: 101
Student Name: Bob, Roll Number: 102
```

Explanation: The constructor was missing the `self` parameter, which is required to refer to the object instance.

Using `self.name` and `self.roll` stores values inside the object properly. Task 5:

Resolving Index Errors in Lists

Prompt: This Python code causes an `IndexError`. Identify the issue, correct the code using safe access methods, and explain the problem.

```
numbers = [10, 20, 30]
```

```
(numbers[5])
```

Input: Bug code

The screenshot shows a Google Colab notebook titled "Untitled29.ipynb". In the code cell, the following Python code is run:

```
numbers = [10, 20, 30]
print(numbers[5]) # Invalid index
```

An error message is displayed:

```
IndexError: list index out of range
```

A tooltip message says: "Enable browser notifications in Settings to get alerts when executions complete". A sidebar on the right contains a news feed with the following content:

2026-01-20

- Launched Data Explorer - a new feature that lets you search Kaggle datasets, models, and competitions directly from a Colab notebook!
- Gemini 3 is now available in Colab.
- In addition to being available in VS Code, Colab is also available for use in Antigravity, Cursor, and Windsurf via the Open VSX Registry!
- H100 is being rolled out for more users.
- Launched a new modern design for the Colab UI.

Python package upgrades:

- accelerate 1.11.0 → 1.12.0
- astropy 7.1.1 → 7.2.0
- bigframes 2.28.0 → 2.31.0
- cachetools 5.5.2 → 6.2.4

Variables Terminal 10:47 AM Python 3

Corrected Code:

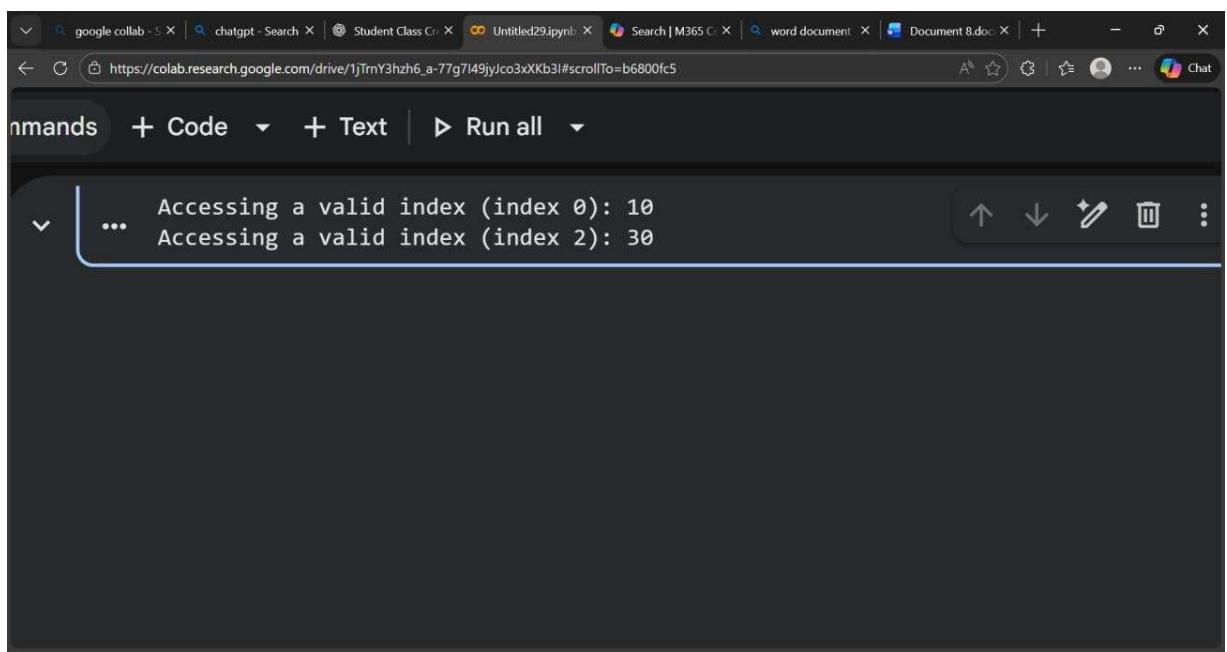
The screenshot shows a Google Colab notebook titled "Untitled29.ipynb". The code cell contains the following Python code:

```
[23] ✓ Os
numbers = [10, 20, 30]

# Attempt to access an element safely using try-except
try:
    print(f"Attempting to access index 5: {numbers[5]}")
except IndexError:
    print("Error: Index out of bounds! The list does not have an element at this index.")

# Example of valid access:
print(f"\nAccessing a valid index (index 0): {numbers[0]}")
print(f"Accessing a valid index (index 2): {numbers[2]}")
```

Output:



The screenshot shows a Google Colab interface with a dark theme. At the top, there are several tabs: "google collab - 5", "chatgpt - Search", "Student Class Cr", "Untitled29.ipynb", "Search | M365 C", "word document", "Document 8.doc", and a "+" button. Below the tabs is a toolbar with icons for "Commands", "+ Code", "+ Text", and "Run all". A dropdown menu is open under "Commands", showing the following code and its output:

```
Accessing a valid index (index 0): 10
...
Accessing a valid index (index 2): 30
```

To the right of the code, there are edit icons: up arrow, down arrow, pencil, trash, and three dots.

Explanation: The program tried to access an index that does not exist in the list, causing an `IndexError`.

Using `len()` to check bounds prevents the program from crashing.