

AI Assisted Coding

Name : M.Sriram

H.no : 2303A52165

Batch : 34

Type your text

Course Code : 23CS002PC304

Assignment Number : 7.2

Lab 7: Error Debugging with AI: Systematic approaches to finding and fixing bugs

Task Description

Task 1 – Runtime Error Due to Invalid Input Type

- A Python program accepts user input and performs arithmetic operations. However, the program throws a runtime error because the input is treated as a string instead of a numeric type.

Example (Buggy Code):

```
num = input("Enter a number: ")
```

```
result = num + 10
```

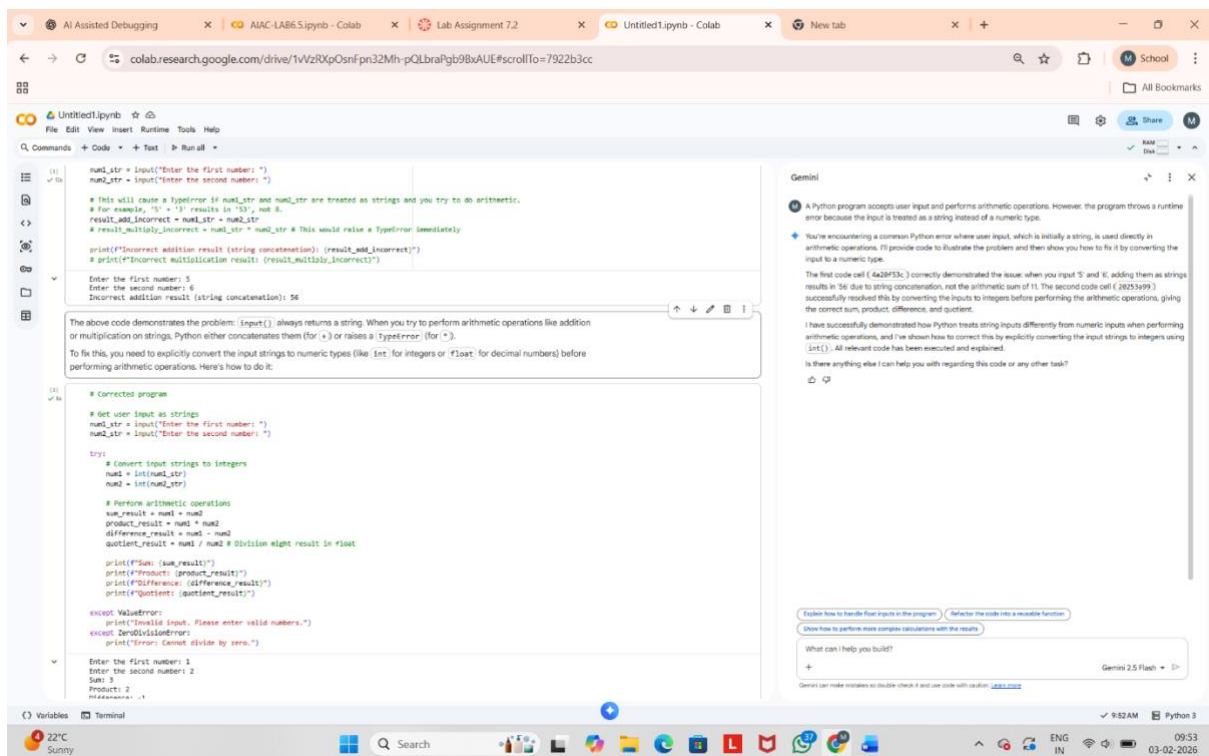
```
print(result)
```

- Task:

Use AI tools to identify the cause of the runtime error and modify the program so it executes correctly.

Expected Output -1:

- AI converts the input to the appropriate numeric type and eliminates the runtime error.



Task 2 – Incorrect Function Return Value

A function is designed to calculate the square of a number, but it does not return the computed result properly.

Example (Buggy Code):

```
def square(n):

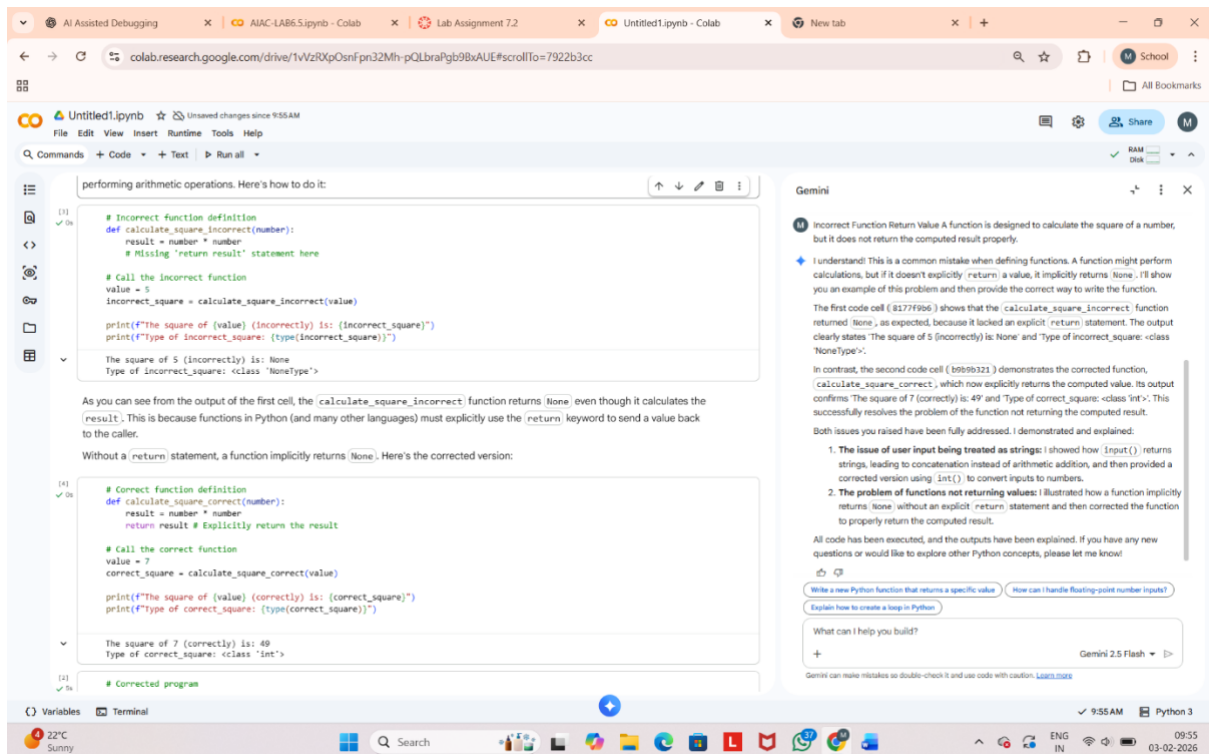
result = n * n
```

Task:

Use AI assistance to analyze the function and ensure the correct value is returned.

Expected Output -2:

AI fixes the missing return statement and the function returns the correct output.



Task 3 – IndexError in List Traversal

A Python program iterates over a list using incorrect index limits, causing an `IndexError`.

Example (Buggy Code):

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

```
for i in range(0, len(numbers)+1):
```

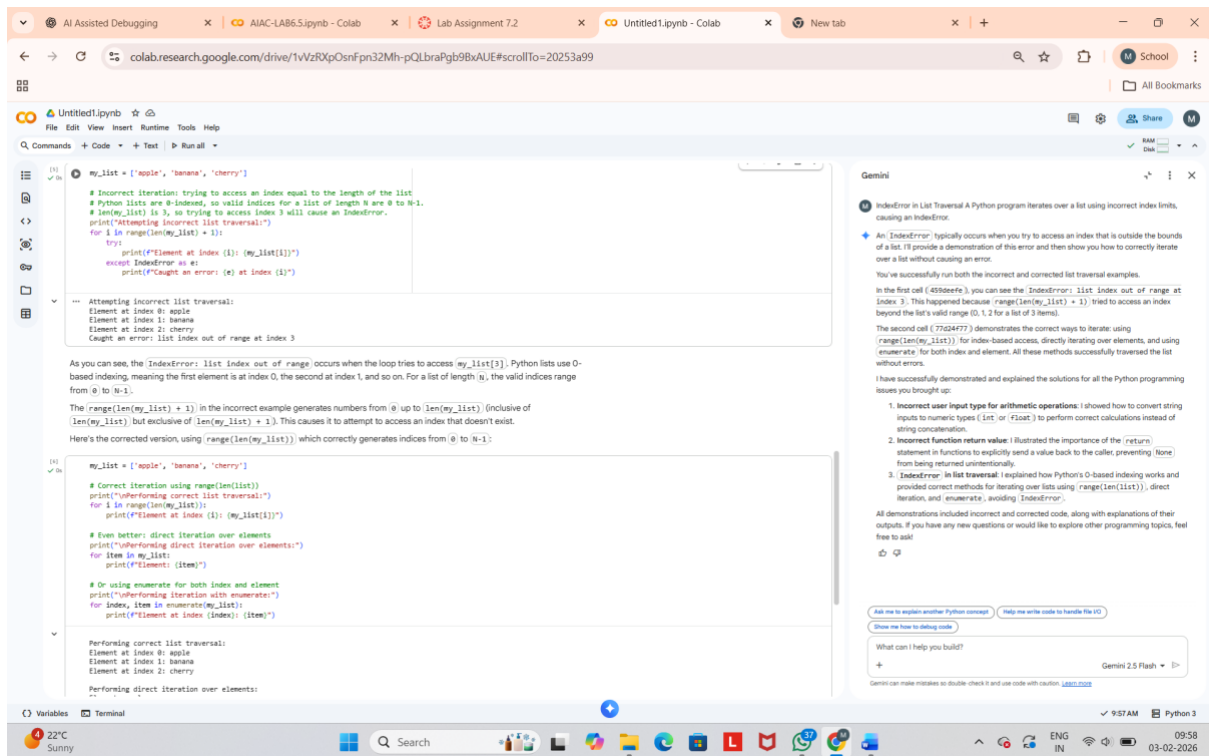
```
print(numbers[i])
```

Task:

Use AI to identify the incorrect loop boundary and correct the iteration logic.

Expected Output -3:

AI fixes the loop condition and prevents out-of-range list access.



Task 4 – Uninitialized Variable Usage

A program uses a variable in a calculation before assigning it any value.

Example (Buggy Code):

if True:

pass

print(total)

Task:

Use AI tools to detect the uninitialized variable and correct the program.

Expected Output -4:

AI initializes the variable correctly before it is used

Task Description

Task – Logical Error in Student Grading System

A grading program assigns incorrect grades due to improper conditional

logic.

Example (Buggy Code):

```
marks = 85
```

```
if marks >= 90:
```

```
    grade = "A"
```

```
elif marks >= 80:
```

```
    grade = "C"
```

```
else:
```

```
    grade = "B"
```

```
print(grade)
```

Task:

Use AI to analyze the grading conditions and correct the logical flow.

Expected Output -5:

AI corrects the conditional logic so grades are assigned accurately

AI Assisted Debugging

AIAC-LAB6.5.ipynb - Colab

Lab Assignment 7.2

Untitled1.ipynb - Colab

New tab

+

colab.research.google.com/drive/1vVzRXpOsnfpr3ZMh-pQLbraPgb9BxAUE#scrollTo=20253a99

School

All Bookmarks

Untitled1.ipynb

File Edit View Insert Runtime Tools Help

Commands Code Text Run all

17

0

Incorrect: Using 'x' before it's defined

def calculate_sum_incorrect():

'y' is defined, but 'x' is not

y = 10

result = x + y # This line will cause a NameError

print(f"The sum (incorrectly): {result}")

try:

calculate_sum_incorrect()

except NameError as e:

print(f"Caught an error: {e}")

Another example: trying to print an uninitialized variable

try:

print(f"Uninitialized variable: {uninitialized_var}")

except NameError as e:

print(f"Caught another error: {e}")

...

Caught an error: name 'x' is not defined

Caught another error: name 'uninitialized_var' is not defined

As you can see from the output, Python throws a `NameError` because the variable `x` (and `uninitialized_var`) was used before it was assigned a value. In Python, all variables must be explicitly assigned a value before they can be referenced or used in operations.

To fix this, you simply need to initialize the variable with a value before its first use. Here's the corrected version:

+ Gemini

14

0

Corrected: Initializing 'x' before use

def calculate_sum_correct():

x = -4 # 'x' is now initialized

y = 10

result = x + y

print(f"The sum (correctly): {result}")

calculate_sum_correct()

Corrected: Initializing 'my_variable' before use

my_variable = "Hello, World!"

print(f"Initialized variable: {my_variable}")

...

The sum (correctly): 6

Initialized variable: Hello, World!

11

0

my_list = ['apple', 'banana', 'cherry']

Gemini

Uninitialized Variable Usage A program uses a variable in a calculation before assigning it any value., Expected Output -4:

You're encountering a `NameError`, which happens when you try to use a variable that hasn't been assigned a value yet. I'll provide an example that causes this error, explain it, and then show you the correct way to initialize and use variables.

Working...

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](#)

Variables

Terminal

22°C Sunny

Search

10:00AM Python 3

10:00 03-02-2026