Assignment-7.5

2303A51062
Batch:29

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)
```

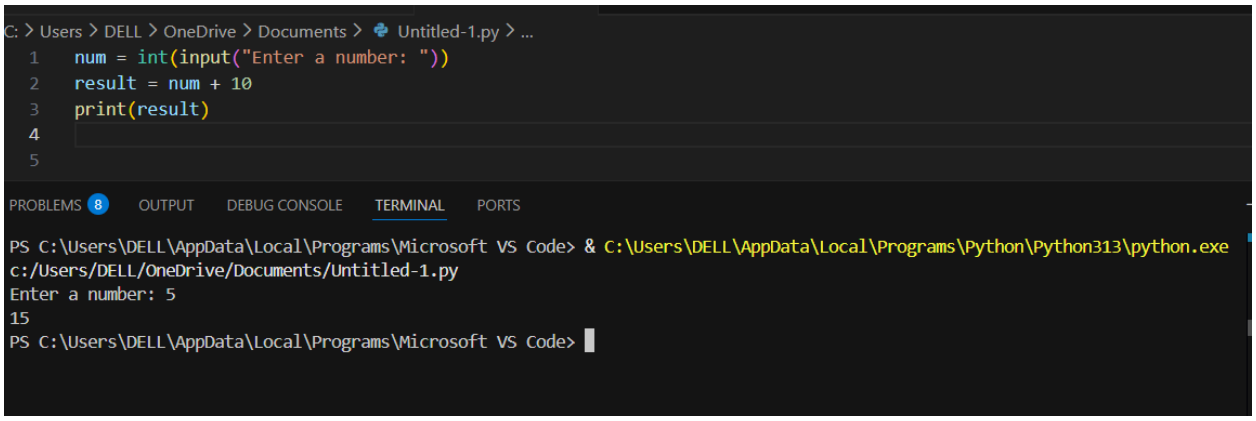
• Task:

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.

Code:



Justification:

AI identifies that the runtime error occurs due to a type mismatch between string input and integer arithmetic. By converting the user input to an integer using `int()`, the program ensures compatible data types, allowing the arithmetic operation to execute successfully and eliminating 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

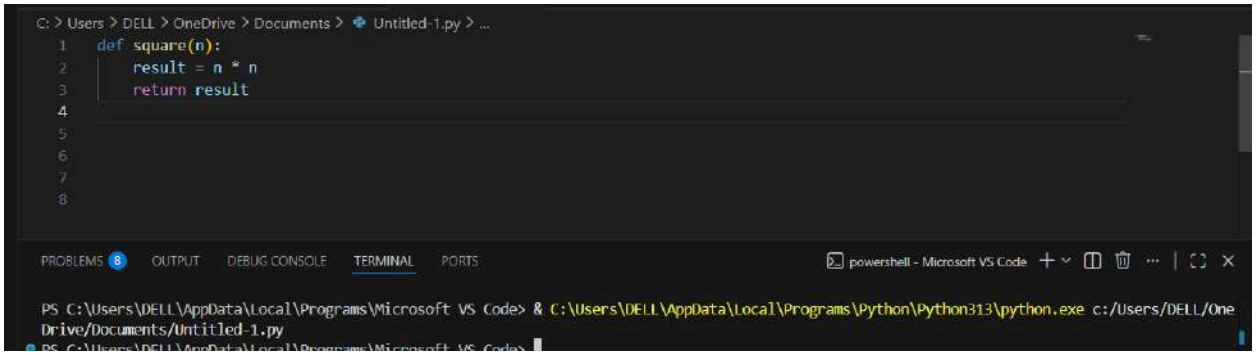
returned.

Expected Output -2:

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

Output.

Code:



Justification:

AI analyzes the function and identifies the missing `return` statement. By adding `return result`, the function correctly sends the computed square back to the caller, ensuring the expected output is produced instead of `None`.

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.

code:

Justification:

AI identifies the incorrect loop boundary that causes out-of-range access. By adjusting the loop to stop at `len(numbers) - 1` (or iterating directly over the list), the program avoids the `IndexError` and runs correctly.

Task Description

Task 4 – Uninitialized Variable Usage

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

Example (Buggy Code):

if True:

pass

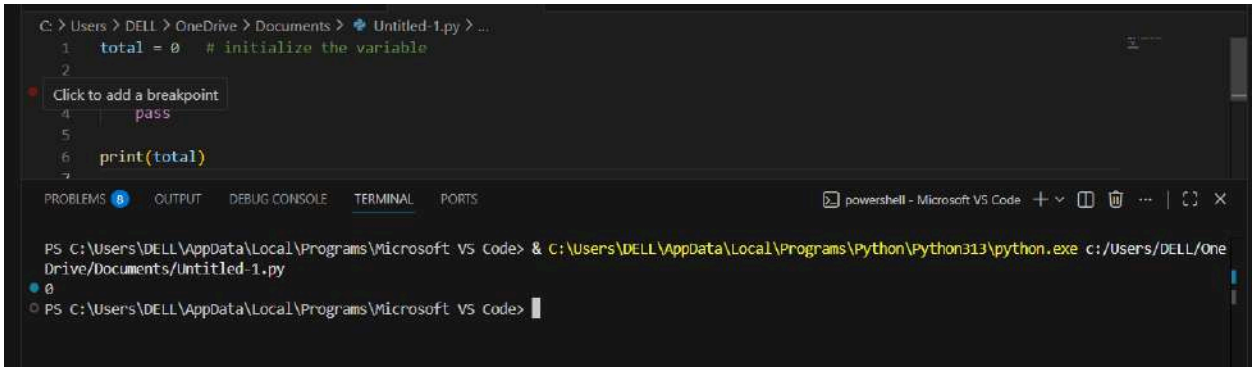
print(total)

Task:

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

Expected Output -4:

Code:



Justification:

AI detects that the variable `total` is referenced before assignment. By initializing `total` with a default value before it is used, the program avoids the runtime error and executes correctly.

Task Description

Task 5 – 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.

Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots.

Code:

Justification:

AI analyzes the conditional logic and identifies that grades are assigned incorrectly due to improper condition mapping. By correcting the grading flow (`A ≥ 90`, `B ≥ 80`, else `C`), the program now assigns accurate grades based on student marks.