ASSIGNMENT-7.2

2303A51600

BATCH-29

TASK-1

PROMPT:

num = input("Enter a number: ")

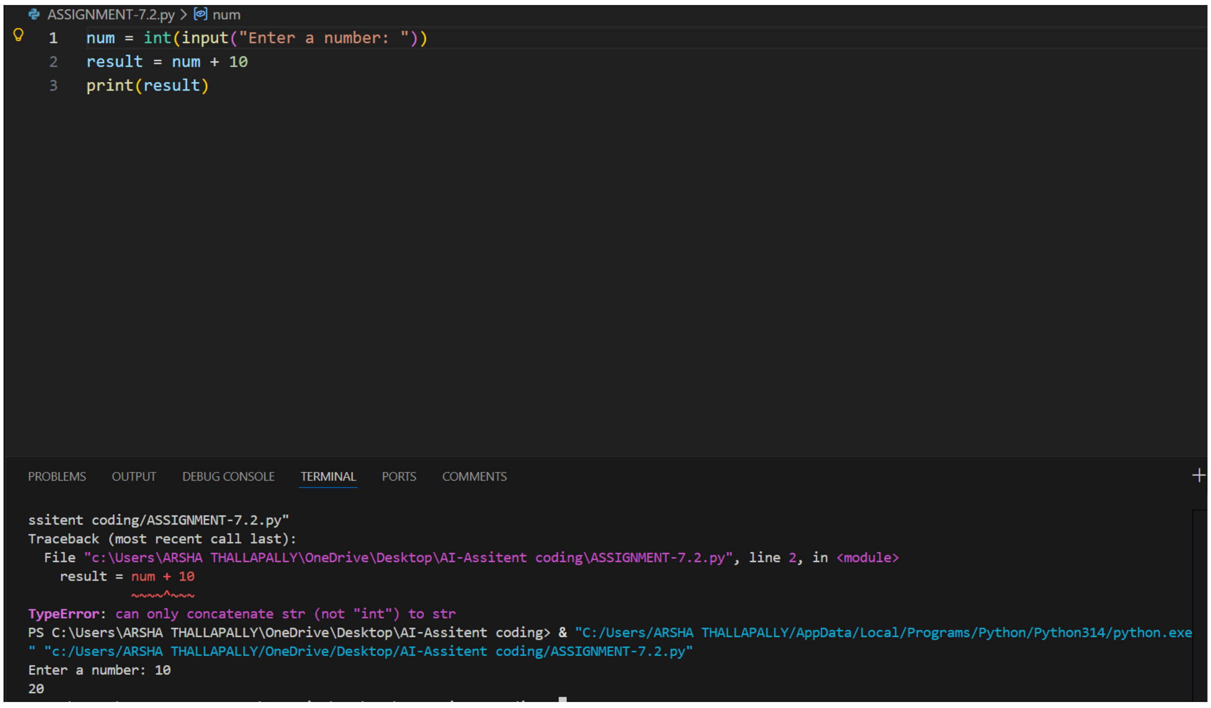
result = num + 10

print(result)

#identify the cause of the runtime error and modify

the program so it executes correctly.

CODE:



```python
num = int(input("Enter a number: "))
result = num + 10
print(result)
```

```
ssitent coding/ASSIGNMENT-7.2.py"
Traceback (most recent call last):
  File "c:\Users\ARSHA THALLAPALLY\OneDrive\Desktop\AI-Assitent coding\ASSIGNMENT-7.2.py", line 2, in <module>
    result = num + 10
             ~~~~^~~~
TypeError: can only concatenate str (not "int") to str
PS C:\Users\ARSHA THALLAPALLY\OneDrive\Desktop\AI-Assitent coding> & "C:/Users/ARSHA THALLAPALLY/AppData/Local/Programs/Python/Python314/python.exe
" "c:/Users/ARSHA THALLAPALLY/OneDrive/Desktop/AI-Assitent coding/ASSIGNMENT-7.2.py"
Enter a number: 10
20
```

OBSERVATION:

The program takes user input using input(), which returns a string by default.
When the code tries to add 10 to this string, a type mismatch occurs, causing a runtime error.
The error happens because arithmetic operations cannot be performed between a string and an integer without converting the input to a numeric type.

TASK-2

PROMPT:

def square(n):

result = n * n

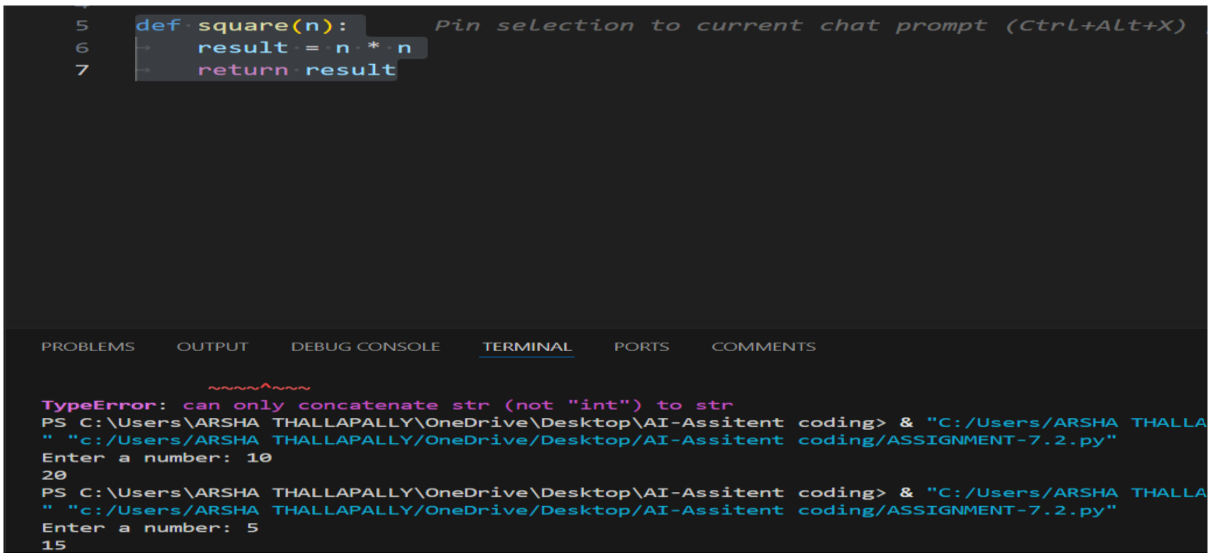
# Analyze the function and ensure the correct value is

returned.

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

output.

CODE:



```
def square(n):        Pin selection to current chat prompt (Ctrl+Alt+X)
    result = n * n
    return result
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   COMMENTS

```
TypeError: can only concatenate str (not "int") to str
PS C:\Users\ARSHA THALLAPALLY\OneDrive\Desktop\AI-Assitent coding> & "C:/Users/ARSHA THALLA
" "c:/Users/ARSHA THALLAPALLY/OneDrive/Desktop/AI-Assitent coding/ASSIGNMENT-7.2.py"
Enter a number: 10
20
PS C:\Users\ARSHA THALLAPALLY\OneDrive\Desktop\AI-Assitent coding> & "C:/Users/ARSHA THALLA
" "c:/Users/ARSHA THALLAPALLY/OneDrive/Desktop/AI-Assitent coding/ASSIGNMENT-7.2.py"
Enter a number: 5
15
```

OBSERVATION:

In the given function, the value of n * n is calculated and stored in the variable result, but it is never returned from the function.

Because of the missing return statement, the function does not send any value back to the caller and returns None by default.

As a result, the expected output is not produced even though the computation is performed.
The bug occurs due to the absence of a return statement in the function definition.

TASK-3

PROMPT:

numbers = [10, 20, 30]

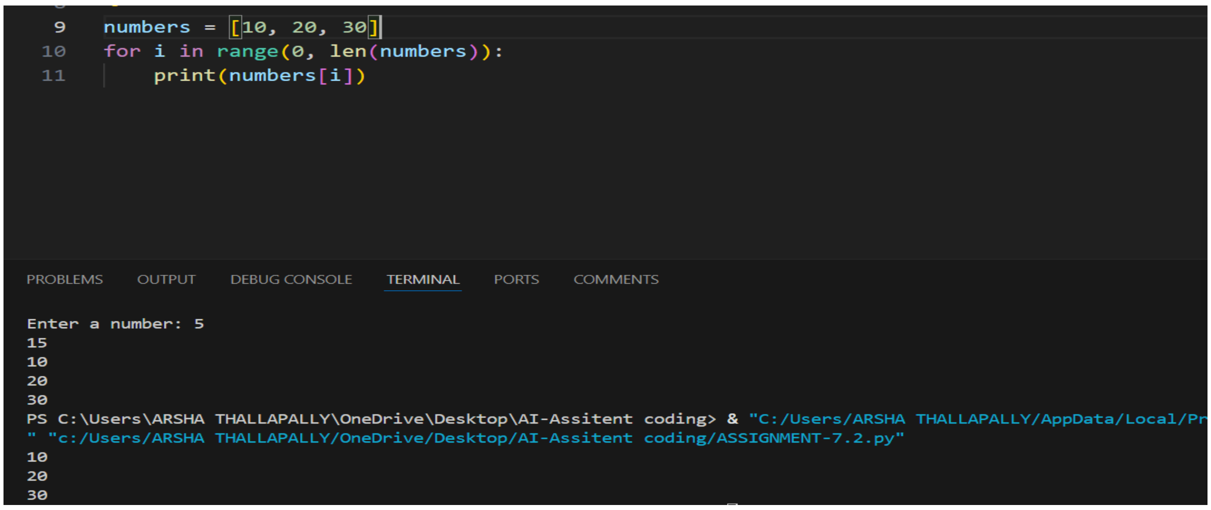
for i in range(0, len(numbers)+1):

print(numbers[i])

#incorrect loop boundary and correct the iteration

logic.

CODE:



```python
numbers = [10, 20, 30]
for i in range(0, len(numbers)):
    print(numbers[i])
```

```
Enter a number: 5
15
10
20
30
PS C:\Users\ARSHA THALLAPALLY\OneDrive\Desktop\AI-Assitent coding> & "C:/Users/ARSHA THALLAPALLY/AppData/Local/Pr
" "c:/Users/ARSHA THALLAPALLY/OneDrive/Desktop/AI-Assitent coding/ASSIGNMENT-7.2.py"
10
20
30
```

OBSERVATION:

The loop runs one step beyond the last valid index because it uses len(numbers) + 1 as the upper limit. This causes the program to try accessing an index that does not exist in the list, resulting in an IndexError. The error occurs due to an incorrect loop boundary that goes out of the list's valid range.

TASK-4

PROMPT:

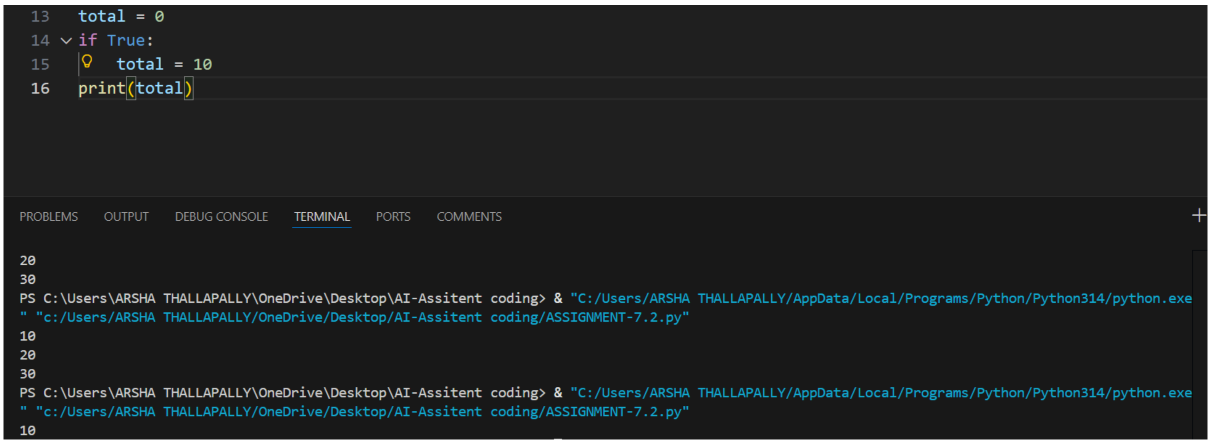
if True:

pass

print(total)

To detect the uninitialized variable and correct the program.

CODE:



```python
total = 0
if True:
    total = 10
print(total)
```

```
20
30
PS C:\Users\ARSHA THALLAPALLY\OneDrive\Desktop\AI-Assitent coding> & "C:/Users/ARSHA THALLAPALLY/AppData/Local/Programs/Python/Python314/python.exe
" "c:/Users/ARSHA THALLAPALLY/OneDrive/Desktop/AI-Assitent coding/ASSIGNMENT-7.2.py"
10
20
30
PS C:\Users\ARSHA THALLAPALLY\OneDrive\Desktop\AI-Assitent coding> & "C:/Users/ARSHA THALLAPALLY/AppData/Local/Programs/Python/Python314/python.exe
" "c:/Users/ARSHA THALLAPALLY/OneDrive/Desktop/AI-Assitent coding/ASSIGNMENT-7.2.py"
10
```

OBSERVATION:

In the given program, the variable total is used in the print statement without being assigned any value beforehand. Since total is never initialized, Python raises a **NameError** when trying to access it.

The error occurs because the program attempts to use a variable before defining it.

To fix this, the variable must be initialized with a value before it is used in any calculation or output.

TASK-5

PROMPT:

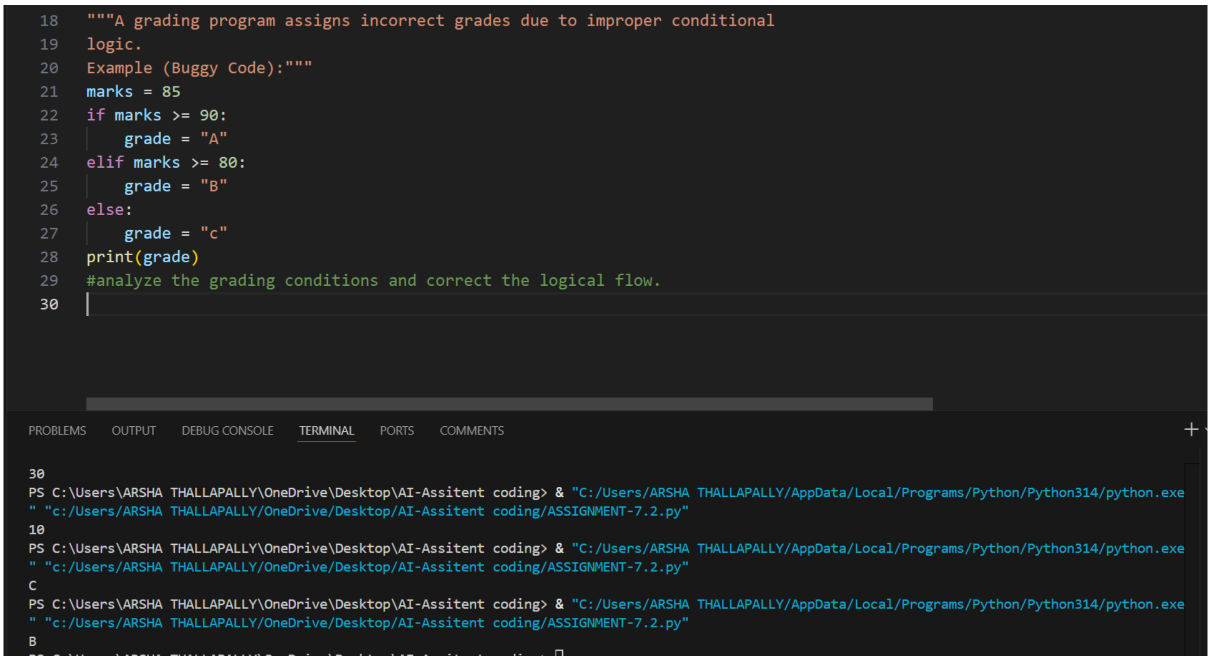
"""A grading program assigns incorrect grades due to improper conditional

logic.

Example (Buggy Code):"""

```
marks = 85

if marks >= 90:

   grade = "A"

elif marks >= 80:

   grade = "B"

else:

   grade = "c"

print(grade)

#analyze the grading conditions and correct the logical flow.
```

CODE:



```python
"""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 = "B"
else:
    grade = "c"
print(grade)
#analyze the grading conditions and correct the logical flow.
```

```
30
PS C:\Users\ARSHA THALLAPALLY\OneDrive\Desktop\AI-Assitent coding> & "C:/Users/ARSHA THALLAPALLY/AppData/Local/Programs/Python/Python314/python.exe
" "c:/Users/ARSHA THALLAPALLY/OneDrive/Desktop/AI-Assitent coding/ASSIGNMENT-7.2.py"
10
PS C:\Users\ARSHA THALLAPALLY\OneDrive\Desktop\AI-Assitent coding> & "C:/Users/ARSHA THALLAPALLY/AppData/Local/Programs/Python/Python314/python.exe
" "c:/Users/ARSHA THALLAPALLY/OneDrive/Desktop/AI-Assitent coding/ASSIGNMENT-7.2.py"
C
PS C:\Users\ARSHA THALLAPALLY\OneDrive\Desktop\AI-Assitent coding> & "C:/Users/ARSHA THALLAPALLY/AppData/Local/Programs/Python/Python314/python.exe
" "c:/Users/ARSHA THALLAPALLY/OneDrive/Desktop/AI-Assitent coding/ASSIGNMENT-7.2.py"
B
```

OBSERVATION:

In the given program, the grading conditions are logically incorrect.
For marks greater than or equal to 80, the grade is assigned as "C" instead of a higher grade, and the else block assigns "B" for lower marks.

This causes wrong grade assignment because higher marks should correspond to better grades.
The logical flow of conditions is reversed, leading to incorrect output.
The issue occurs due to improper ordering and incorrect grade mapping in the conditional statements.