AI Assisted Coding Assignment- 7.2

> 2303A51543 BT: 29 VELDI.HRUTHIKA 10.02.2026

💡 **Task 1 – Runtime Error Due to Invalid Input Type**

Description:

- 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 +10print(result)
```

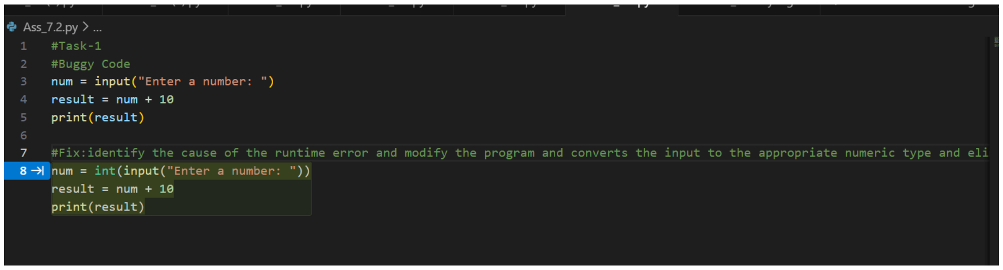
Task:

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

Expected Output - 1:

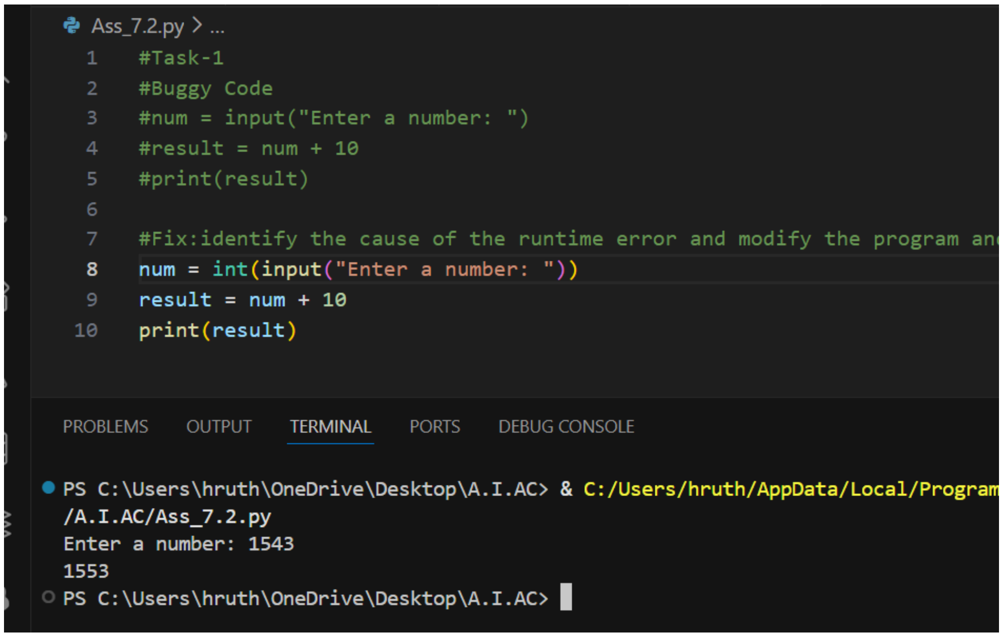
- AI converts the input to the appropriate numeric type and eliminates the runtime error.

```
#Task-1
#Buggy Code
#num = input("Enter a number: ")
#result = num + 10
#print(result)

#Fix:identify the cause of the runtime error and modify the program and converts the inpu
t to the appropriate numeric type and eliminates the runtime error.
num = int(input("Enter a number: "))
result = num + 10
print(result)
```



Ass_7.2.py > ...

```
#Task-1
#Buggy Code
num = input("Enter a number: ")
result = num + 10
print(result)

#Fix:identify the cause of the runtime error and modify the program and converts the input to the appropriate numeric type and eli
num = int(input("Enter a number: "))
result = num + 10
print(result)
```



```python
#Task-1
#Buggy Code
#num = input("Enter a number: ")
#result = num + 10
#print(result)

#Fix:identify the cause of the runtime error and modify the program an
num = int(input("Enter a number: "))
result = num + 10
print(result)
```

```
PS C:\Users\hruth\OneDrive\Desktop\A.I.AC> & C:/Users/hruth/AppData/Local/Program
/A.I.AC/Ass_7.2.py
Enter a number: 1543
1553
PS C:\Users\hruth\OneDrive\Desktop\A.I.AC>
```

💡 **Task 2 – Incorrect Function Return Value**

Description:

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

Example (Buggy Code):

```
defsquare(n):
    result = n * n
```

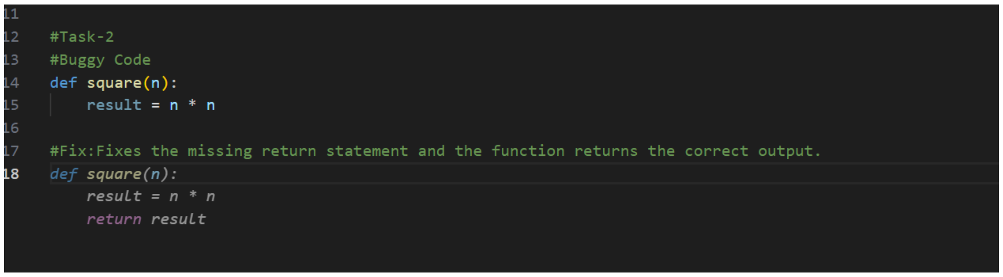
Task:

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

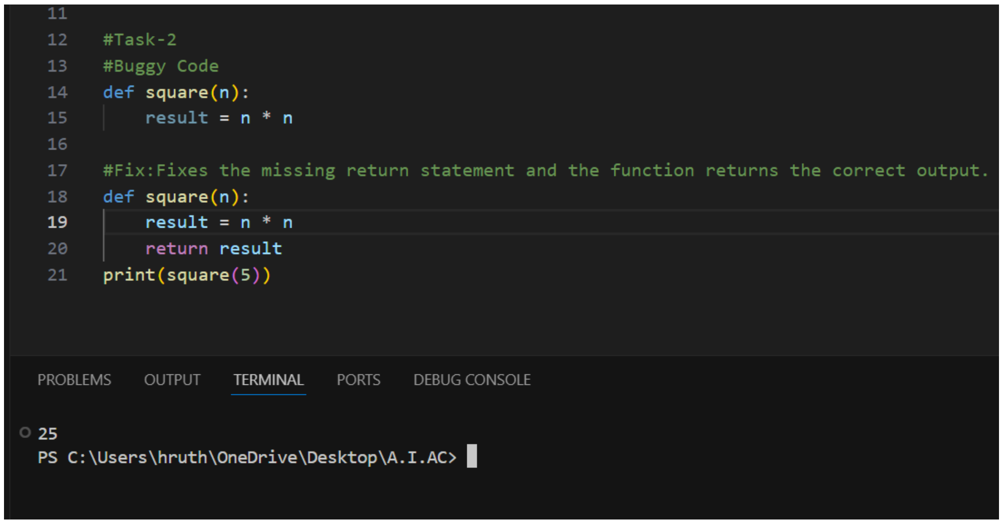
Expected Output - 2:

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

```python
#Task-2
#Buggy Code
def square(n):
    result = n * n

#Fix:Fixes the missing return statement and the function returns the correct output.
def square(n):
    result = n * n
    return result
print(square(5))
```



```
#Task-2
#Buggy Code
def square(n):
    result = n * n

#Fix:Fixes the missing return statement and the function returns the correct output.
def square(n):
    result = n * n
    return result
```



```
#Task-2
#Buggy Code
def square(n):
    result = n * n

#Fix:Fixes the missing return statement and the function returns the correct output.
def square(n):
    result = n * n
    return result
print(square(5))
```

```
25
PS C:\Users\hruth\OneDrive\Desktop\A.I.AC>
```

💡 Task 3 – IndexError in List Traversal

Description:

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

Example (Buggy Code):

```
numbers = [10,20,30]for iinrange(0,len(numbers)+1):print(numbers[i])
```

Task:

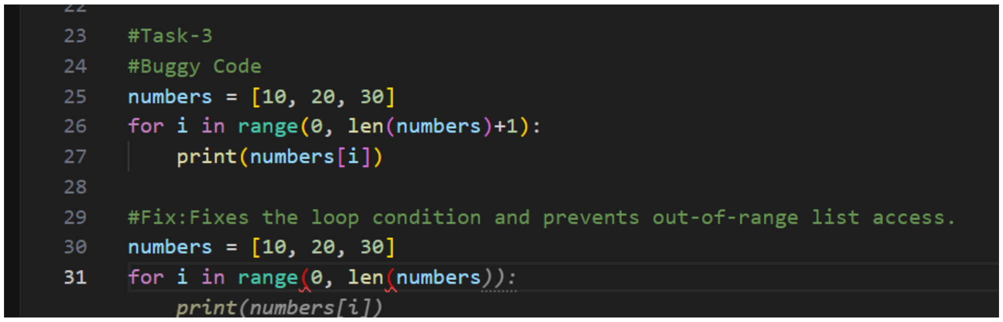
- Use AI to identify the incorrect loop boundary and correct the iteration logic.

Expected Output - 3:

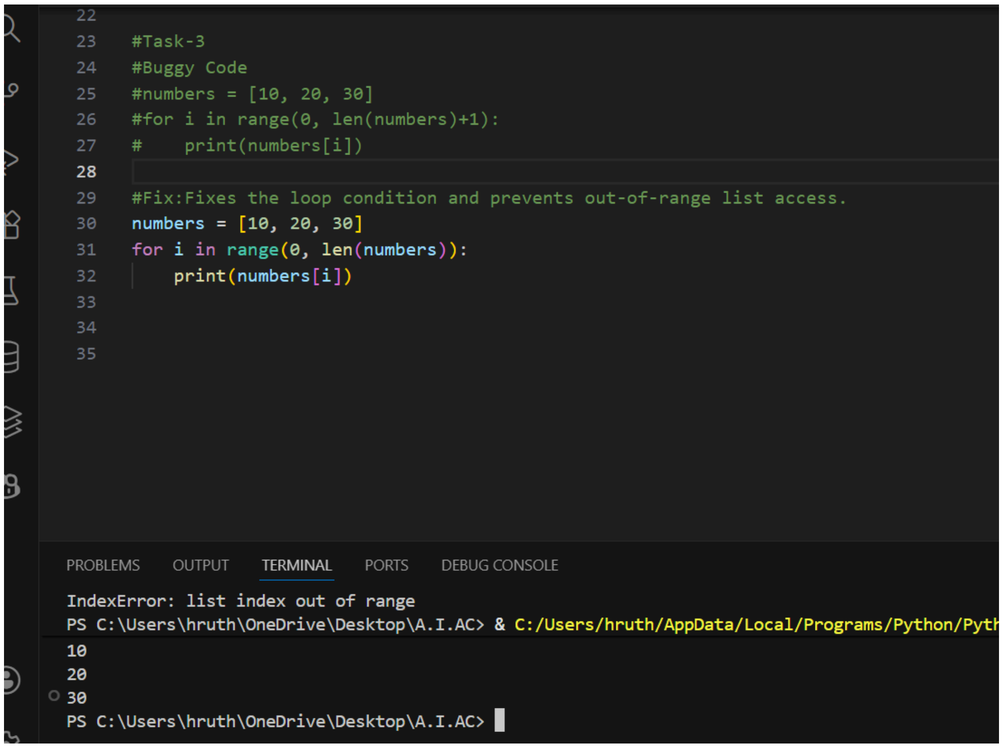
- AI fixes the loop condition and prevents out-of-range list access.

```
#Task-3
#Buggy Code
#numbers = [10, 20, 30]
#for i in range(0, len(numbers)+1):
#    print(numbers[i])

#Fix:Fixes the loop condition and prevents out-of-range list access.
numbers = [10, 20, 30]
```

```
for i in range(0, len(numbers)):
    print(numbers[i])
```



```
#Task-3
#Buggy Code
numbers = [10, 20, 30]
for i in range(0, len(numbers)+1):
    print(numbers[i])

#Fix:Fixes the loop condition and prevents out-of-range list access.
numbers = [10, 20, 30]
for i in range(0, len(numbers)):
    print(numbers[i])
```



```
#Task-3
#Buggy Code
#numbers = [10, 20, 30]
#for i in range(0, len(numbers)+1):
#    print(numbers[i])

#Fix:Fixes the loop condition and prevents out-of-range list access.
numbers = [10, 20, 30]
for i in range(0, len(numbers)):
    print(numbers[i])
```

```
IndexError: list index out of range
PS C:\Users\hruth\OneDrive\Desktop\A.I.AC> & C:/Users/hruth/AppData/Local/Programs/Python/Pyth
10
20
30
PS C:\Users\hruth\OneDrive\Desktop\A.I.AC>
```

💡 **Task 4 – Uninitialized Variable Usage**

Description:

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

Example (Buggy Code):

```
ifTrue:passprint(total)
```

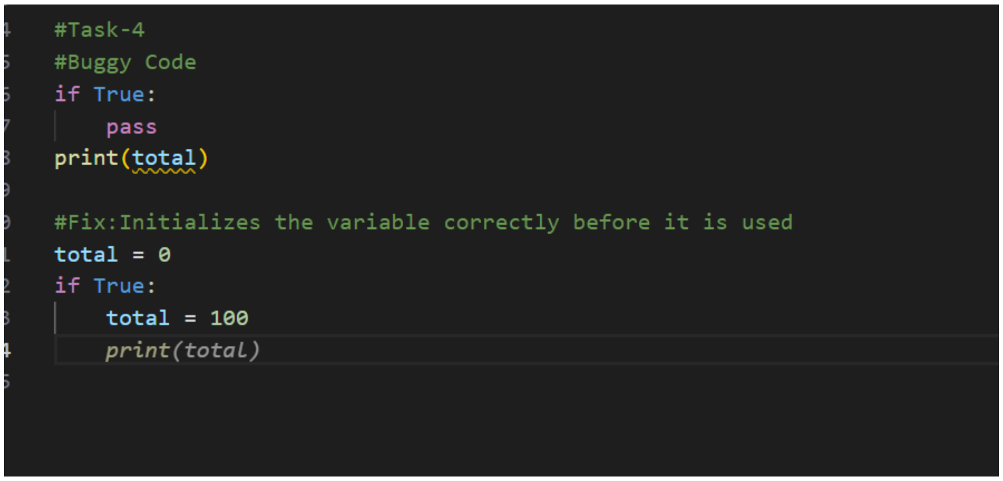
Task:

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

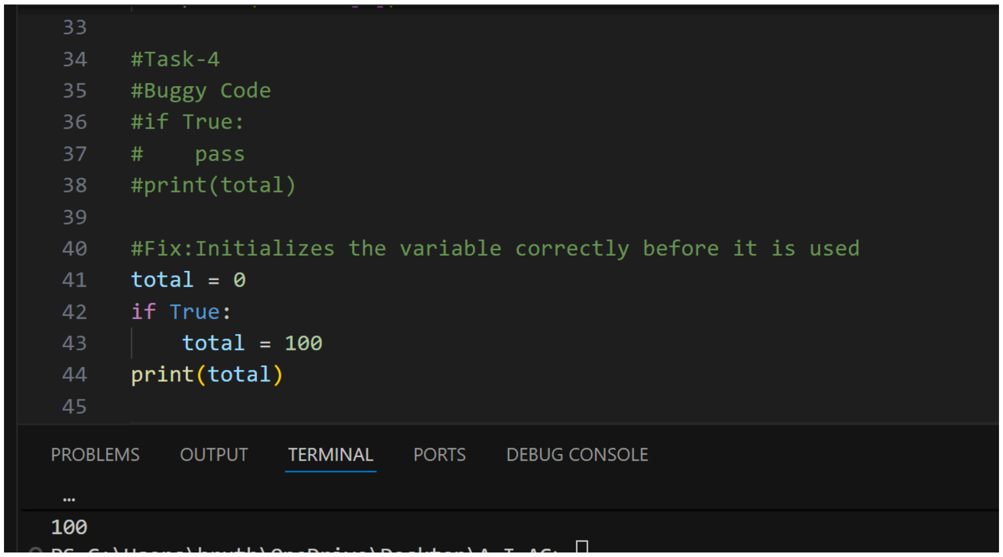
Expected Output - 4:

- AI initializes the variable correctly before it is used.

```
#Task-4
#Buggy Code
#if True:
#    pass
#print(total)

#Fix:Initializes the variable correctly before it is used
total = 0
if True:
    total = 100
print(total)
```



```
#Task-4
#Buggy Code
if True:
    pass
print(total)

#Fix:Initializes the variable correctly before it is used
total = 0
if True:
    total = 100
    print(total)
```



```
#Task-4
#Buggy Code
#if True:
#    pass
#print(total)

#Fix:Initializes the variable correctly before it is used
total = 0
if True:
    total = 100
print(total)

```

```
…
100
```

💡 **Task 5 – Logical Error in Student Grading System**

Description:

- 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:

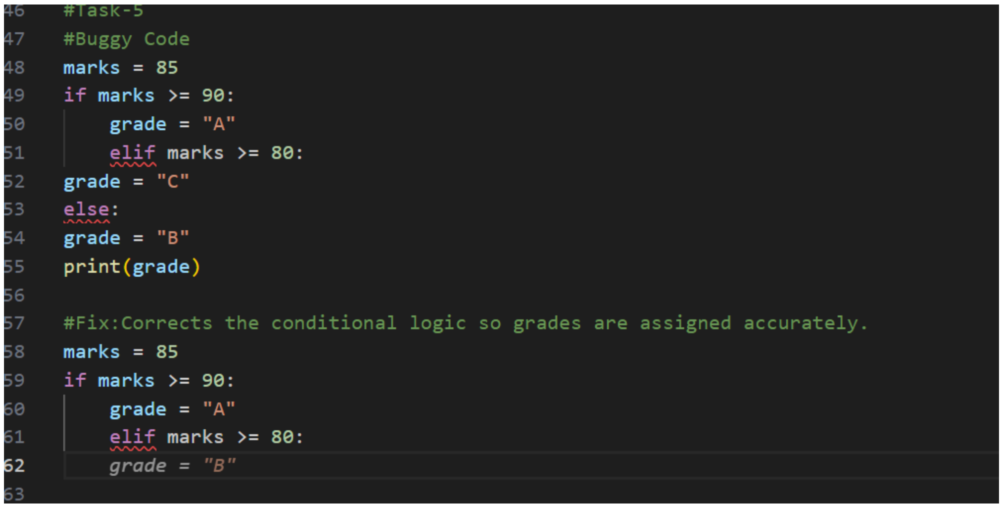
- Use AI to analyze the grading conditions and correct the logical flow.

Expected Output - 5:

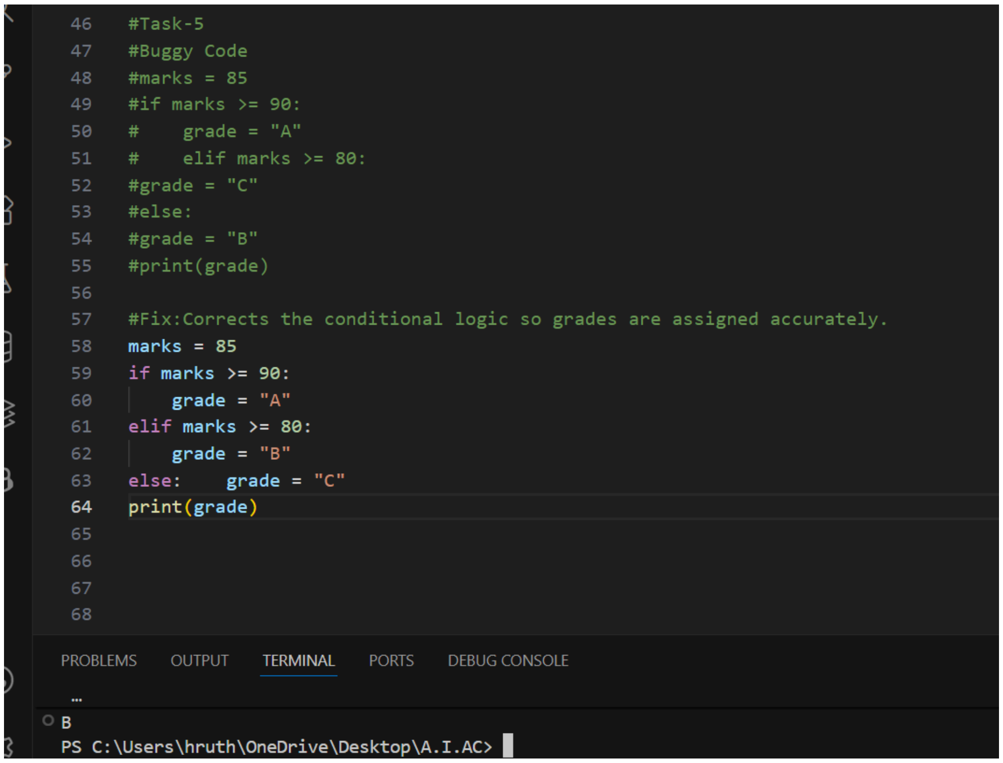
- AI corrects the conditional logic so grades are assigned accurately.

```
#Task-5
#Buggy Code
#marks = 85
#if marks >= 90:
#    grade = "A"
#    elif marks >= 80:
#grade = "C"
#else:
#grade = "B"
#print(grade)
```

```
#Fix:Corrects the conditional logic so grades are assigned accurately.
marks = 85
if marks >= 90:
    grade = "A"
elif marks >= 80:
    grade = "B"
else:    grade = "C"
print(grade)
```



```
#Task-5
#Buggy Code
marks = 85
if marks >= 90:
    grade = "A"
    elif marks >= 80:
grade = "C"
else:
grade = "B"
print(grade)

#Fix:Corrects the conditional logic so grades are assigned accurately.
marks = 85
if marks >= 90:
    grade = "A"
    elif marks >= 80:
    grade = "B"
```



```
#Task-5
#Buggy Code
#marks = 85
#if marks >= 90:
#    grade = "A"
#    elif marks >= 80:
#grade = "C"
#else:
#grade = "B"
#print(grade)

#Fix:Corrects the conditional logic so grades are assigned accurately.
marks = 85
if marks >= 90:
    grade = "A"
elif marks >= 80:
    grade = "B"
else:    grade = "C"
print(grade)
```

```
B
PS C:\Users\hruth\OneDrive\Desktop\A.I.AC>
```