

# 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 input 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 > ...
1 #Task-1
2 #Buggy Code
3 num = input("Enter a number: ")
4 result = num + 10
5 print(result)
6
7 #Fix:identify the cause of the runtime error and modify the program and converts the input to the appropriate numeric type and eliminates the runtime error.
8 num = int(input("Enter a number: "))
9 result = num + 10
10 print(result)
```

```
 Ass_7.2.py > ...
1  #Task-1
2  #Buggy Code
3  #num = input("Enter a number: ")
4  #result = num + 10
5  #print(result)
6
7  #Fix:identify the cause of the runtime error and modify the program and correct it
8  num = int(input("Enter a number: "))
9  result = num + 10
10 print(result)

PROBLEMS    OUTPUT    TERMINAL    PORTS    DEBUG CONSOLE

● PS C:\Users\hruth\OneDrive\Desktop\A.I.AC> & C:/Users/hruth/AppData/Local/Programs/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.

```
#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))
```

```

11
12  #Task-2
13  #Buggy Code
14  def square(n):
15  |    result = n * n
16
17 #Fix:Fixes the missing return statement and the function returns the correct output.
18 def square(n):
19 |    result = n * n
20 |    return result

```

```

11
12  #Task-2
13  #Buggy Code
14  def square(n):
15  |    result = n * n
16
17 #Fix:Fixes the missing return statement and the function returns the correct output.
18 def square(n):
19 |    result = n * n
20 |    return result
21 print(square(5))

```

PROBLEMS    OUTPUT    TERMINAL    PORTS    DEBUG CONSOLE

○ 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])
```

```
22  
23  #Task-3  
24  #Buggy Code  
25  numbers = [10, 20, 30]  
26  for i in range(0, len(numbers)+1):  
27  |   print(numbers[i])  
28  
29  #Fix:Fixes the loop condition and prevents out-of-range list access.  
30  numbers = [10, 20, 30]  
31  for i in range(0, len(numbers)):  
      print(numbers[i])
```

The screenshot shows a code editor interface with a dark theme. On the left, there is a sidebar with various icons. The main area contains the following Python code:

```
22  
23  #Task-3  
24  #Buggy Code  
25  numbers = [10, 20, 30]  
26  #for i in range(0, len(numbers)+1):  
27  #   print(numbers[i])  
28  
29  #Fix:Fixes the loop condition and prevents out-of-range list access.  
30  numbers = [10, 20, 30]  
31  for i in range(0, len(numbers)):  
32  |   print(numbers[i])  
33  
34  
35
```

Below the code editor, there is a navigation bar with tabs: PROBLEMS, OUTPUT, TERMINAL, PORTS, and DEBUG CONSOLE. The TERMINAL tab is currently selected. The terminal window displays the following output:

```
IndexError: list index out of range  
PS C:\Users\hruth\OneDrive\Desktop\A.I.AC> & C:/Users/hruth/AppData/Local/Programs/Python/Python39/python.exe C:/Users/hruth/OneDrive/Desktop/A.I.AC.py  
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)
```

```
4  #Task-4
5  #Buggy Code
6  if True:
7  |    pass
8  print(total)
9
10 #Fix:Initializes the variable correctly before it is used
11 total = 0
12 if True:
13 |    total = 100
14 |    print(total)
15
```

```
33
34 #Task-4
35 #Buggy Code
36 #if True:
37 #    pass
38 #print(total)
39
40 #Fix:Initializes the variable correctly before it is used
41 total = 0
42 if True:
43     total = 100
44 print(total)
45
```

PROBLEMS    OUTPUT    TERMINAL    PORTS    DEBUG CONSOLE

...

```
100
C:\Users\Hussain\OneDrive\Desktop\AI Task\
```



## 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)
```

```
46  #Task-5  
47  #Buggy Code  
48  marks = 85  
49  if marks >= 90:  
50      grade = "A"  
51      elif marks >= 80:  
52          grade = "C"  
53      else:  
54          grade = "B"  
55  print(grade)  
56  
57  #Fix:Corrects the conditional logic so grades are assigned accurately.  
58  marks = 85  
59  if marks >= 90:  
60      grade = "A"  
61      elif marks >= 80:  
62          grade = "B"  
63  
64
```

```
46  #Task-5  
47  #Buggy Code  
48  #marks = 85  
49  #if marks >= 90:  
50  #    grade = "A"  
51  #    elif marks >= 80:  
52  #grade = "C"  
53  #else:  
54  #grade = "B"  
55  #print(grade)  
56  
57  #Fix:Corrects the conditional logic so grades are assigned accurately.  
58  marks = 85  
59  if marks >= 90:  
60      grade = "A"  
61      elif marks >= 80:  
62          grade = "B"  
63  else:    grade = "C"  
64  print(grade)  
65  
66  
67  
68
```

PROBLEMS    OUTPUT    TERMINAL    PORTS    DEBUG CONSOLE

...

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