

# School of Computer Science and Artificial Intelligence

---

## Lab Assignment # 7.2

---

**Program** : B. Tech (CSE)  
**Specialization** : AIML  
**Course Title** : AI Assisted Coding  
**Course Code** : 23CS002PC304  
**Semester** : VI  
**Academic Session** : 2025-2026  
**Name of Student** : R . Anirudh  
**Enrollment No.** : 2303A52103  
**Batch No.** : 33  
**Date** : 03/02/26

### Lab 7: Error Debugging with AI (Week 4 – Tuesday)

**Topic:** Systematic approaches to finding and fixing bugs using AI

---

#### Task 1 – Runtime Error Due to Invalid Input Type Bug Analysis (AI Explanation)

- `input()` always returns a **string**
- Adding a string and an integer causes a **TypeError**

```

  3]
  ✓ 5s
  ▶ num = int(input("Enter a number: "))
    result = num + 10
    print(result)

  ... Enter a number: 5
    15

  4]
  ✓ 0s
  numbers = [10, 20, 30]
  for i in range(len(numbers)):
    print(numbers[i])

  10
  20
  30

```

## Expected Output – 1

- AI converts user input to an integer
- Runtime error is eliminated

---

## Task 2 – Incorrect Function Return Value Bug Analysis (AI Explanation)

- Function calculates the square but **does not return it**
- Without return, Python returns None

```

  9]
  ✓ 0s
  ▶ def square(n):
    result = n * n

  + Code + Text

  10]
  ✓ 0s
  def square(n):
    result = n * n
    return result

```

## Expected Output – 2

- Function correctly returns the square of the number

---

## Task 3 – IndexError in List Traversal Bug Analysis (AI Explanation)

- range(0, len(numbers)+1) goes **one step too far**
- Causes IndexError: list index out of range

**Buggy Code**

```
2] numbers = [10, 20, 30]
0s for i in range(0, len(numbers)+1):
    print(numbers[i])

... 10
    20
    30

-----
IndexError                                Traceback (most recent call last)
/tmp/ipython-input-2172525831.py in <cell line: 0>()
      1 numbers = [10, 20, 30]
      2 for i in range(0, len(numbers)+1):
----> 3     print(numbers[i])

IndexError: list index out of range
```

Next steps: [Explain error](#)

**AI-Corrected Code**

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

... 10
    20
    30
```

[+ Code](#) [+ Text](#)

## Expected Output – 3

- Loop boundary corrected
- Prevents out-of-range access

---

## Task 4 – Uninitialized Variable Usage Bug Analysis (AI Explanation)

- Variable total is used before assignment
- Causes NameError

**Buggy Code**

```
[5] if True:
0s     pass
    print(total)

... -----
NameError                                Traceback (most recent call last)
/tmp/ipython-input-3608487366.py in <cell line: 0>()
      1 if True:
      2     pass
----> 3     print(total)

NameError: name 'total' is not defined
```

Next steps: [Explain error](#)

```
AI-Corrected Code

[6] total = 0
✓ Os if True:
    pass
    print(total)

... 0

+ Code + Text
```

### Expected Output – 4

- Variable initialized before use
- Program runs safely

---

### Task 5 – Logical Error in Student Grading System Bug Analysis (AI Explanation)

- Logical order of grading conditions is incorrect
- marks  $\geq 80$  wrongly assigns grade C
- else block assigns B incorrectly

```
Buggy Code

[7] marks = 85
✓ Os if marks >= 90:
    grade = "A"
    elif marks >= 80:
        grade = "C"
    else:
        grade = "B"
    print(grade)

... C

AI-Corrected Code
```

```
AI-Corrected Code

[8] marks = 85
✓ Os if marks >= 90:
    grade = "A"
    elif marks >= 80:
        grade = "B"
    else:
        grade = "C"
    print(grade)

... B
```

### Expected Output – 5

- Correct grade is assigned based on marks
  - Logical flow fixed
-

### **Summary: AI-Assisted Debugging Strategies Used**

- ✓ Type conversion for runtime errors
- ✓ Return statement validation
- ✓ Loop boundary correction
- ✓ Variable initialization checks
- ✓ Logical condition reordering