Lab 7: Error Debugging with AI: Systematic approaches to finding and fixing bugs

2303A51760

Batch-28

Assignment-7.3

Task 1: Fixing Syntax Errors

Function contains syntax error (missing colon).

Buggy Code

```
def add(a, b)
    return a + b
```

AI Prompt Used

"Find the syntax error in this Python function and correct it."

Corrected Code

```
def add(a, b):
    return a + b
print(add(5, 3))
```

Explanation

- Python requires a **colon (:)** after function declaration.
- Without colon → SyntaxError occurs.
- AI detected missing colon and corrected it.

Expected Output

8

Task 2: Debugging Logic Errors in Loops

Loop runs infinitely because increment is missing.

Buggy Code

```
i = 1
while i <= 5:
    print(i)
```

AI Prompt Used

"Why is this loop infinite? Fix the logic error."

Corrected Code

```
i = 1
while i <= 5:
    print(i)
    i += 1
```

Explanation

- Value of i never changes in buggy code.
- Condition i <= 5 remains true forever.
- AI added increment i += 1.

Expected Output

1

2

3

4

5

Task 3: Handling Runtime Errors (Division by Zero)

Program crashes when divisor is zero.

Buggy Code

```
def divide(a, b):
    return a / b

print(divide(10, 0))
```

AI Prompt Used

"Identify runtime error and add safe exception handling."

Corrected Code

```
def divide(a, b):
    try:
        return a / b
    except ZeroDivisionError:
        return "Cannot divide by zero"

print(divide(10, 0))
```

Explanation

- Division by zero raises **ZeroDivisionError**.
- AI added try-except block.
- Prevents program crash.

Expected Output

Cannot divide by zero

Task 4: Debugging Class Definition Errors

Constructor missing self parameter.

Buggy Code

```
class Student:
    def __init__(name, age):
        name = name
        age = age
```

AI Prompt Used

"Fix constructor error and explain why self is required."

Corrected Code

```
class Student:
    def __init__(self, name, age):
        self.name = name
        self.age = age

s1 = Student("Sai", 19)
```

```
print(s1.name, s1.age)
```

Explanation

- self represents current object instance.
- Without self, attributes are not stored inside object.
- AI corrected constructor and variable assignment.

Expected Output

Sai 19

Task 5: Resolving Index Errors in Lists

Program accesses invalid list index.

Buggy Code

```
numbers = [10, 20, 30]
print(numbers[5])
```

AI Prompt Used

"Identify index error and suggest safe list access."

Corrected Code (Method 1: Length Check)

```
numbers = [10, 20, 30]

index = 5
if index < len(numbers):
    print(numbers[index])
```

```
else:
    print("Index out of range")
```

Corrected Code (Method 2: Exception Handling)

```
numbers = [10, 20, 30]


try:
    print(numbers[5])
except IndexError:
    print("Index out of range")
```

Explanation

- Accessing index greater than list size causes **IndexError**.
- AI suggested:
    - Bounds checking using len()
    - OR try-except handling

Expected Output

Index out of range