| SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE | | DEPARTMENT OF COMPUTER SCIENCE ENGINEERING | |
|---|---|---|---|
| **Program Name:** B. Tech | | **Assignment Type: Lab** | **Academic Year:**2025-2026 |
| **Course Coordinator Name** | | Dr. Rishabh Mittal | |
| **Instructor(s) Name** | | Mr. S Naresh Kumar<br>Ms. B. Swathi<br>Dr. Sasanko Shekhar Gantayat<br>Mr. Md Sallauddin<br>Dr. Mathivanan<br>Mr. Y Srikanth<br>Ms. N Shilpa<br>Dr. Rishabh Mittal (Coordinator)<br>Dr. R. Prashant Kumar<br>Mr. Ankushavali MD<br>Mr. B Viswanath<br>Ms. Sujitha Reddy<br>Ms. A. Anitha<br>Ms. M.Madhuri<br>Ms. Katherashala Swetha<br>Ms. Velpula sumalatha<br>Mr. Bingi Raju | |
| **Course Code** | 23CS002PC304 | **Course Title** | AI Assisted Coding |
| **Year/Sem** | III/II | **Regulation** | R23 |
| **Date and Day of Assignment** | **Week4 – Wednesday** | **Time(s)** | 23CSBTB01 To 23CSBTB52 |
| **Duration** | 2 Hours | **Applicable to Batches** | All batches |
| **AssignmentNumber:7.3**(Present assignment number)/**24**(Total number of assignments) | | | |
| **NAME : I. Abhinay Powar HALLTICKET N0:2303A51811 BATCH:26** | | | |

| Q.No. | Question | *Expected Time to complete* |
|---|---|---|
| | | |
| 1 | **Lab 7: Error Debugging with AI: Systematic approaches to finding and fixing bugs** | Week4 - Wednesday |
| | **Lab Objectives**<br>• To identify and correct syntax, logic, and runtime errors in Python programs using AI tools | |

• To understand common programming bugs and AI-assisted debugging suggestions
• To evaluate how AI explains, detects, and fixes different types of coding errors
• To build confidence in using AI for structured debugging practices

**Lab Outcomes (LOs)**
After completing this lab, students will be able to:
• Use AI tools to detect and correct syntax, logic, and runtime errors
• Interpret AI-suggested bug fixes and explanations
• Apply systematic debugging strategies using AI-generated insights
• Refactor buggy code using reliable programming patterns

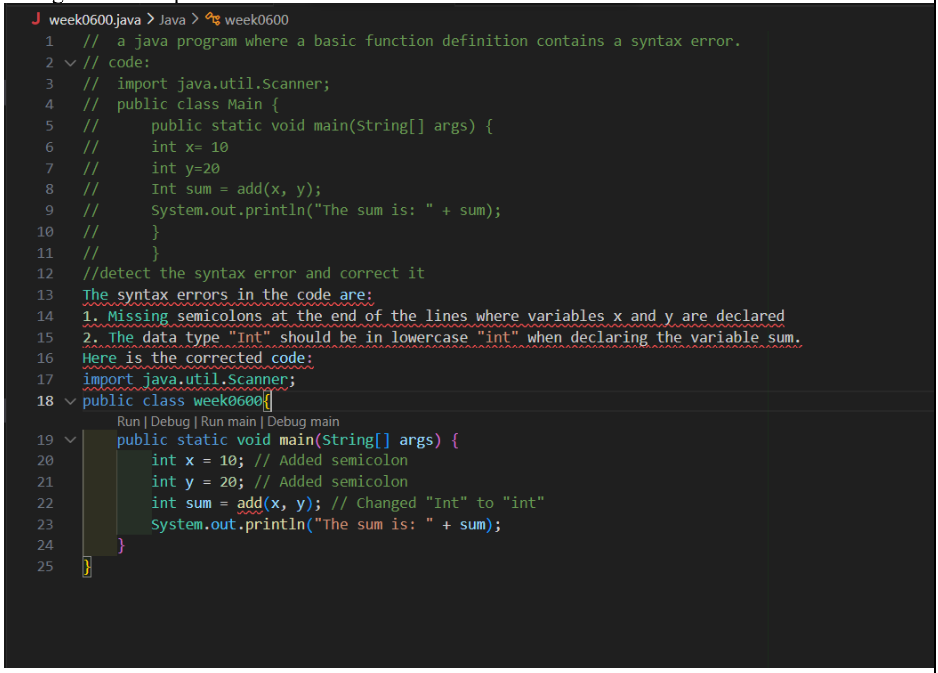
**Task 1: Fixing Syntax Errors**

**Scenario**
You are reviewing a Python program where a basic function definition contains a syntax error.

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

**Requirements**
• Provide a Python function add(a, b) with a **missing colon**
• Use an AI tool to detect the syntax error
• Allow AI to correct the function definition
• Observe how AI explains the syntax issue

**Expected Output**
• Corrected function with proper syntax
• Syntax error resolved successfully
• AI-generated explanation of the fix


week0600.java > Java > week0600

```
//  a java program where a basic function definition contains a syntax error.
// code:
//  import java.util.Scanner;
//  public class Main {
//      public static void main(String[] args) {
//      int x= 10
//      int y=20
//      Int sum = add(x, y);
//      System.out.println("The sum is: " + sum);
//      }
//      }
//detect the syntax error and correct it
The syntax errors in the code are:
1. Missing semicolons at the end of the lines where variables x and y are declared
2. The data type "Int" should be in lowercase "int" when declaring the variable sum.
Here is the corrected code:
import java.util.Scanner;
public class week0600{
    public static void main(String[] args) {
        int x = 10; // Added semicolon
        int y = 20; // Added semicolon
        int sum = add(x, y); // Changed "Int" to "int"
        System.out.println("The sum is: " + sum);
    }
}
```

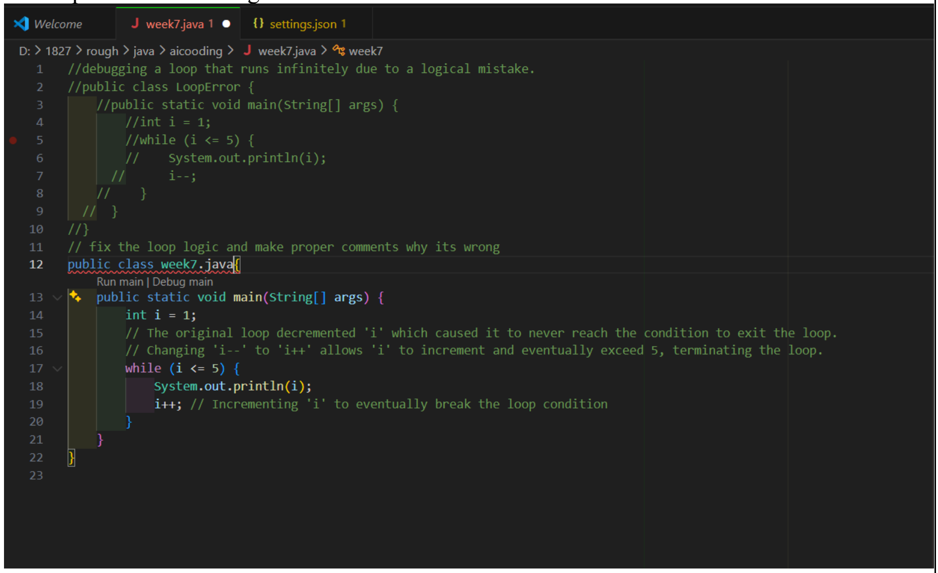
**Task 2: Debugging Logic Errors in Loops**

**Scenario**
You are debugging a loop that runs infinitely due to a logical mistake.

```python
def count_down(n):
    while n >= 0:
        print(n)
        n += 1  # Should be n -= 1
```

**Requirements**
• Provide a loop with an **increment or decrement error**
• Use AI to identify the cause of infinite iteration
• Let AI fix the loop logic
• Analyze the corrected loop behavior

**Expected Output**
• Infinite loop issue resolved
• Correct increment/decrement logic applied
• AI explanation of the logic error



```java
//debugging a loop that runs infinitely due to a logical mistake.
//public class LoopError {
    //public static void main(String[] args) {
        //int i = 1;
        //while (i <= 5) {
        //    System.out.println(i);
       //      i--;
    //    }
  //  }
//}
// fix the loop logic and make proper comments why its wrong
public class week7.java{
    public static void main(String[] args) {
        int i = 1;
        // The original loop decremented 'i' which caused it to never reach the condition to exit the loop.
        // Changing 'i--' to 'i++' allows 'i' to increment and eventually exceed 5, terminating the loop.
        while (i <= 5) {
            System.out.println(i);
            i++; // Incrementing 'i' to eventually break the loop condition
        }
    }
}
```

**Task 3: Handling Runtime Errors (Division by Zero)**

**Scenario**
A Python function crashes during execution due to a division by zero error.

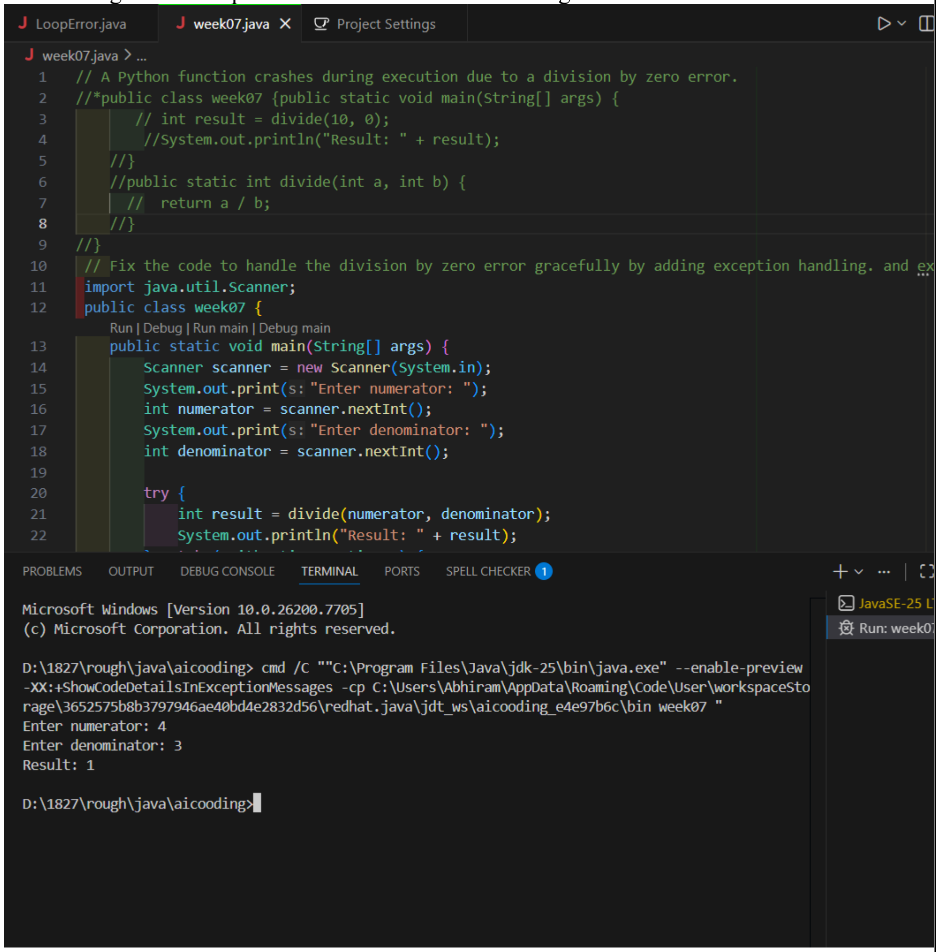
```python
# Debug the following code
def divide(a, b):
    return a / b

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

**Requirements**
• Provide a function that performs division without validation
• Use AI to identify the runtime error
• Let AI add try-except blocks for safe execution
• Review AI's error-handling approach

**Expected Output**
• Function executes safely without crashing
• Division by zero handled using try-except

• Clear AI-generated explanation of runtime error handling



```java
// A Python function crashes during execution due to a division by zero error.
//*public class week07 {public static void main(String[] args) {
        // int result = divide(10, 0);
         //System.out.println("Result: " + result);
    //}
    //public static int divide(int a, int b) {
      //  return a / b;
    //}
//}
 // Fix the code to handle the division by zero error gracefully by adding exception handling. and ex
 import java.util.Scanner;
 public class week07 {
    Run | Debug | Run main | Debug main
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print(s: "Enter numerator: ");
        int numerator = scanner.nextInt();
        System.out.print(s: "Enter denominator: ");
        int denominator = scanner.nextInt();

        try {
            int result = divide(numerator, denominator);
            System.out.println("Result: " + result);
```

```
Microsoft Windows [Version 10.0.26200.7705]
(c) Microsoft Corporation. All rights reserved.

D:\1827\rough\java\aicooding> cmd /C ""C:\Program Files\Java\jdk-25\bin\java.exe" --enable-preview
-XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\Abhiram\AppData\Roaming\Code\User\workspaceSto
rage\3652575b8b3797946ae40bd4e2832d56\redhat.java\jdt_ws\aicooding_e4e97b6c\bin week07 "
Enter numerator: 4
Enter denominator: 3
Result: 1

D:\1827\rough\java\aicooding>
```

**Task 4: Debugging Class Definition Errors**

**Scenario**

You are given a faulty Python class where the constructor is incorrectly defined.

python

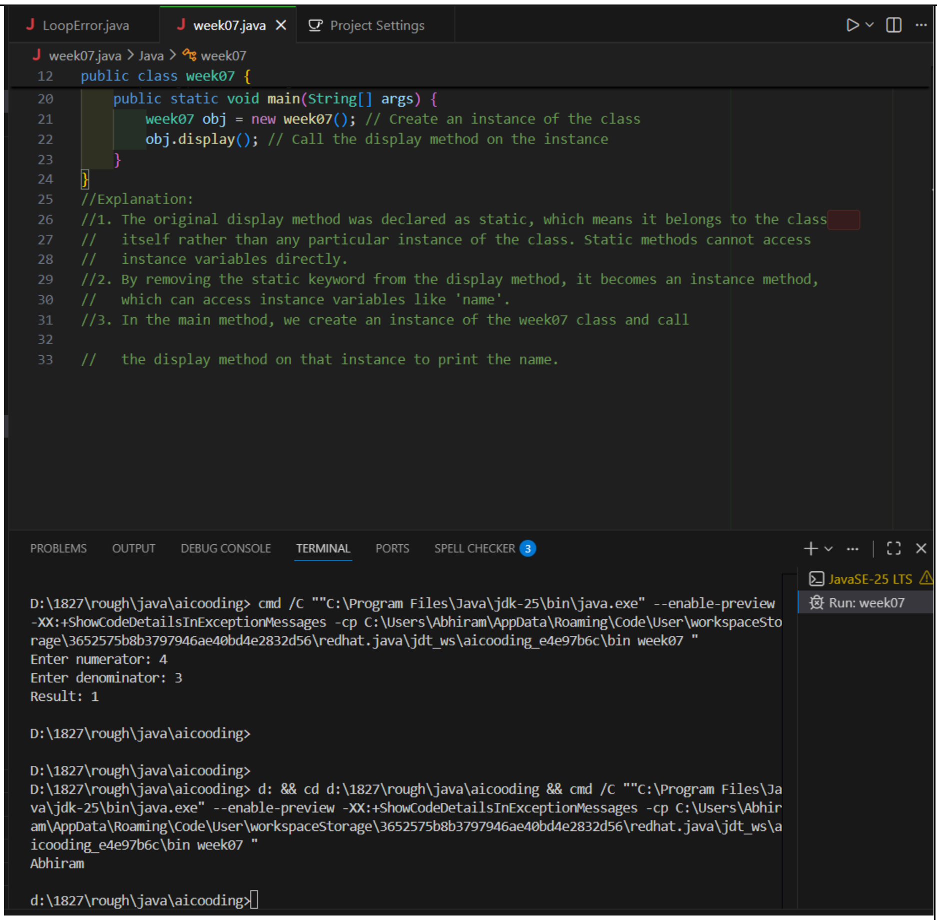
```python
class Rectangle:
    def __init__(length, width):
        self.length = length
        self.width = width
```

**Requirements**

• Provide a class definition with **missing self-parameter**
• Use AI to identify the issue in the __init__() method
• Allow AI to correct the class definition
• Understand why self is required

**Expected Output**

• Corrected __init__() method
• Proper use of self in class definition
• AI explanation of object-oriented error



```java
public class week07 {
    public static void main(String[] args) {
        week07 obj = new week07(); // Create an instance of the class
        obj.display(); // Call the display method on the instance
    }
}
//Explanation:
//1. The original display method was declared as static, which means it belongs to the class
//   itself rather than any particular instance of the class. Static methods cannot access
//   instance variables directly.
//2. By removing the static keyword from the display method, it becomes an instance method,
//   which can access instance variables like 'name'.
//3. In the main method, we create an instance of the week07 class and call

//   the display method on that instance to print the name.
```

```
D:\1827\rough\java\aicooding> cmd /C ""C:\Program Files\Java\jdk-25\bin\java.exe" --enable-preview
-XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\Abhiram\AppData\Roaming\Code\User\workspaceSto
rage\3652575b8b3797946ae40bd4e2832d56\redhat.java\jdt_ws\aicooding_e4e97b6c\bin week07 "
Enter numerator: 4
Enter denominator: 3
Result: 1

D:\1827\rough\java\aicooding>

D:\1827\rough\java\aicooding>
D:\1827\rough\java\aicooding> d: && cd d:\1827\rough\java\aicooding && cmd /C ""C:\Program Files\Ja
va\jdk-25\bin\java.exe" --enable-preview -XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\Abhir
am\AppData\Roaming\Code\User\workspaceStorage\3652575b8b3797946ae40bd4e2832d56\redhat.java\jdt_ws\a
icooding_e4e97b6c\bin week07 "
Abhiram

d:\1827\rough\java\aicooding>
```

## Task 5: Resolving Index Errors in Lists

### Scenario

A program crashes when accessing an invalid index in a list.

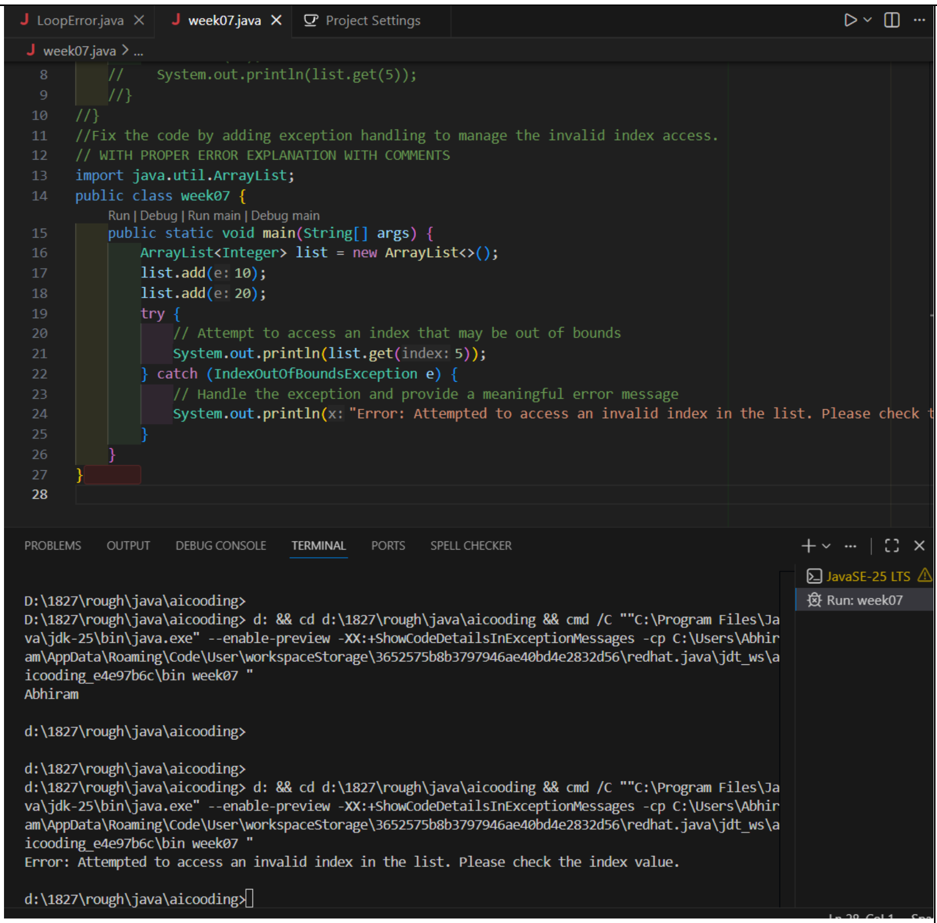
```python
numbers = [1, 2, 3]
print(numbers[5])
```

### Requirements

• Provide code that accesses an **out-of-range list index**
• Use AI to identify the Index Error
• Let AI suggest safe access methods
• Apply bounds checking or exception handling

### Expected Output

• Index error resolved
• Safe list access logic implemented
• AI suggestion using length checks or exception handling



```java
    //    System.out.println(list.get(5));
    //}
//}
//Fix the code by adding exception handling to manage the invalid index access.
// WITH PROPER ERROR EXPLANATION WITH COMMENTS
import java.util.ArrayList;
public class week07 {
    public static void main(String[] args) {
        ArrayList<Integer> list = new ArrayList<>();
        list.add(e: 10);
        list.add(e: 20);
        try {
            // Attempt to access an index that may be out of bounds
            System.out.println(list.get(index: 5));
        } catch (IndexOutOfBoundsException e) {
            // Handle the exception and provide a meaningful error message
            System.out.println(x: "Error: Attempted to access an invalid index in the list. Please check t
        }
    }
}
```

```
D:\1827\rough\java\aicooding>
D:\1827\rough\java\aicooding> d: && cd d:\1827\rough\java\aicooding && cmd /C ""C:\Program Files\Ja
va\jdk-25\bin\java.exe" --enable-preview -XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\Abhir
am\AppData\Roaming\Code\User\workspaceStorage\3652575b8b3797946ae40bd4e2832d56\redhat.java\jdt_ws\a
icooding_e4e97b6c\bin week07 "
Abhiram

d:\1827\rough\java\aicooding>

d:\1827\rough\java\aicooding>
d:\1827\rough\java\aicooding> d: && cd d:\1827\rough\java\aicooding && cmd /C ""C:\Program Files\Ja
va\jdk-25\bin\java.exe" --enable-preview -XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\Abhir
am\AppData\Roaming\Code\User\workspaceStorage\3652575b8b3797946ae40bd4e2832d56\redhat.java\jdt_ws\a
icooding_e4e97b6c\bin week07 "
Error: Attempted to access an invalid index in the list. Please check the index value.

d:\1827\rough\java\aicooding>
```

**Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots**