Test 6
SET A
Debugging

1. import java.util.ArrayList;

import java.util.List;

public class Test {

public static void main(String[] args) {

List<Object> list = new ArrayList<>();

list.add("Hello");

list.add(123);

for (Object obj : list) {

String str = (String) obj;

System.out.println(str);

}

}

}

 Issue: ClassCastException when trying to cast an Integer to String.

 Solution: Use instanceof check before casting.

Solution:

```
import java.util.ArrayList;

import java.util.List;

public class Test {

    public static void main(String[] args) {

        List<Object> list = new ArrayList<>();

        list.add("Hello");

        list.add(123);
```

```
        for (Object obj : list) {
            if (obj instanceof String) {
                String str = (String) obj;
                System.out.println(str);
            } else if (obj instanceof Integer) {
                int num = (int) obj;
                System.out.println(num);
            }
        }
    }
}
```

Output:

```
java -cp /tmp/TGA1WyA8mh/Test
Hello
123

=== Code Execution Successful ===
```

2 . Why is my ArithmeticException not being caught in the divide method?

 Potential Issue: Ensure the divide method is correctly wrapped in a try-catch block.

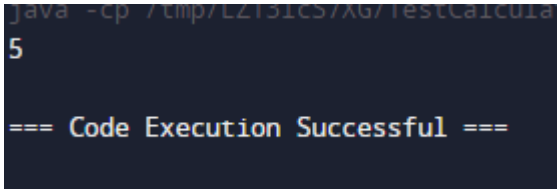
```
public class Calculator {
public int divide(int a, int b) {
return a / b;
}
}

public class TestCalculator {
public static void main(String[] args) {
```

```
Calculator calc = new Calculator();

try {

int result = calc.divide(10, 0);

System.out.println(result);

} catch (ArithmeticException e) {

System.out.println(&amp;quot;Cannot divide by zero.&amp;quot;);

}

}


}
```

Solution:

```
class calculator {

   public int divide(int a, int b) {

      return a / b;

   }

}

public class TestCalculator {

   public static void main(String[] args) {

      calculator calc = new calculator();

      try {

         int result = calc.divide(10, 2);

         System.out.println(result);

      } catch (ArithmeticException e) {

         System.out.println("Cannot divide by zero.");

      }

}

}
```

Output:



3. Why is my array index out of bounds when trying to access an element?

 Potential Issue: Check that the index is within the valid range (0 to array length - 1).

```
public class ArrayIndexOutOfBounds {

public static void main(String[] args) {

int[] numbers = {1, 2, 3};

try {

System.out.println(numbers[3]); // Index 3 is out of bounds

} catch (ArrayIndexOutOfBoundsException e) {

System.out.println(&amp;quot;Caught ArrayIndexOutOfBoundsException.&amp;quot;);

}

}

}
```

Solution:

```
public class ArrayIndexOutOfBounds {
    public static void main(String[] args) {
        int[] numbers = {1, 2, 3};
        try {
            System.out.println(numbers[2]); // Accessing index 2, which is valid
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Caught ArrayIndexOutOfBoundsException.");
        }
    }
}
```

Output:

```
java -cp /tmp/VZpCtDyacG/ArrayIndexOutOfBounds
3

=== Code Execution Successful ===
```