Exception Handling

- An event, which occurs during the execution of a program,

that disrupts the normal flow of the program's instructions.

- **Checked Exception**
  - ✓ Checked exceptions are called compile-time exceptions because these exceptions are checked at compile-time by the compiler.

- ClassNotFoundException
- IOException
- SQLException
- FileNotFoundException
- NoSuchMethodException

➢ ClassNotFoundException

```
public class Main {
   final static String string= "Hi";
   public static void main(String argvs[]) throws ClassNotFoundException {
      Class.forName(string);
      System.out.println("Welcome to aspire");
   }
}
```

➢ FileNotFoundException

```
public class Main {
   public static void main(String[] args){
      try
      {
         File file = new File("input.txt");
         Scanner scanner = new Scanner(file);
      } catch (FileNotFoundException e) {
         System.out.println("Error: File not found!");
```

```
        }
    }
}
```

```
Error: File not found!
```

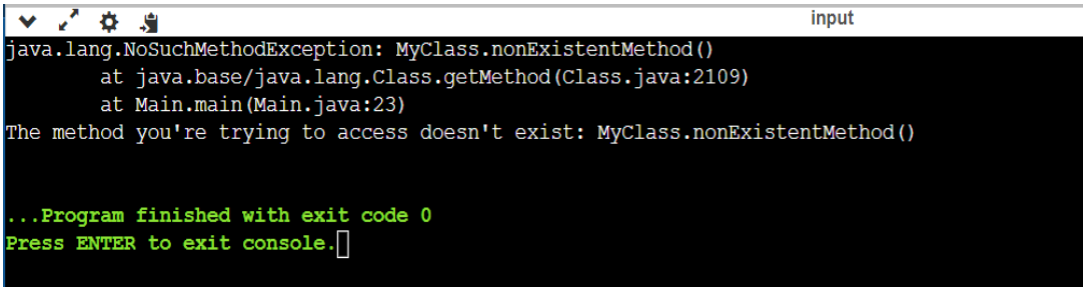
- IOException

```
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
public class Main{
      public static void main(String[] args) throws FileNotFoundException {
          BufferedReader br = new BufferedReader(new FileReader("hello.txt"));
          String line = br.readLine();
          IOExceptionJava(16777384) String java.io.BufferedReader.readLine() throws
       IOException
      }
}
```

- NoSuchMethodException

```
import java.lang.reflect.Method;
public class Main{
public static void main(String[] args) {
      MyClass obj = new MyClass();
      try {
          Method method = obj.getClass().getMethod("nonExistentMethod");
          method.invoke(obj);
      } catch (NoSuchMethodException e) {
          e.printStackTrace();
          System.err.println("The method you're trying to access doesn't exist: " +
e.getMessage());
      } catch (Exception e) {
          e.printStackTrace();
      }
   }
}

class MyClass {
   public void myMethod() {
      System.out.println("This is my method!");
   }
}
```



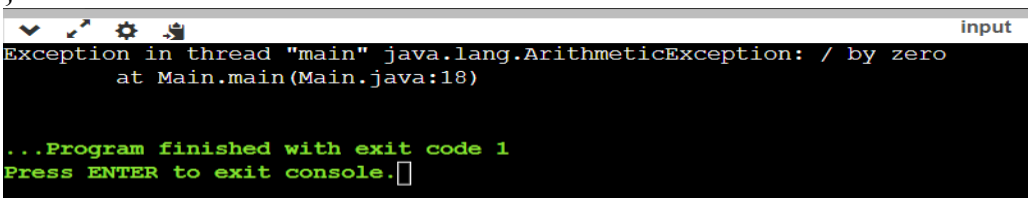
- **Unchecked Exception**

✓ The classes that inherit the RuntimeException are known as unchecked exceptions
✓ Unchecked exceptions are not checked at compile-time, but they are checked at runtime.

- ArithmeticException
- NullPointerException
- ArrayIndexOutOfBoundException

➢ Arithemetic Exception

```
public class Main {
   public static void main(String args[]) {
      int number=10,number1=0,result;
      result=number/number1;
      System.out.println(result);
      System.out.println("End of the program");
   }
}
```

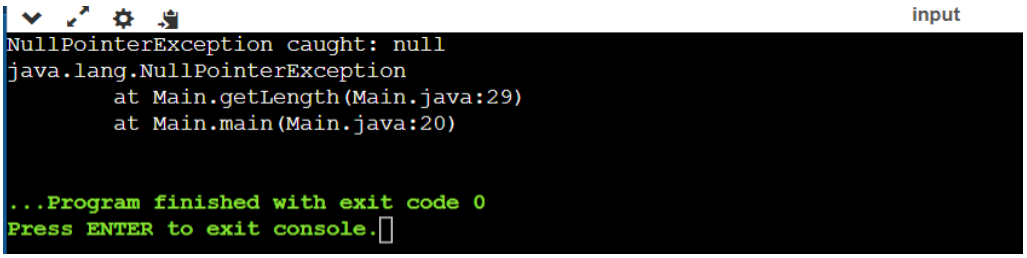


➢ NullPointerException

```
public class Main {
   public static void main(String[] args) {
      String string = null;
      try {
         int length = getLength(string);
         System.out.println("Length of the string: " + length);
      } catch (NullPointerException e) {
         System.out.println("NullPointerException caught: " + e.getMessage());
         e.printStackTrace();
```

```
        }
    }

    public static int getLength(String s) {
        return s.length();
    }
}
```



- ArrayIndexOutOfBoundsException

```
public class Main {
    public static void main(String[] args) {
        int[] student_id = {101, 102, 103, 104, 105};
        try {
            int element = student_id[105];
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("ArrayIndexOutOfBoundsException: Invalid index
accessed.");
        }
    }
}
```