Names: MANIRAGABA Theoneste

ID :26078

Course:OOP

Group: wensday evening

Assignment: Understanding and Handling OOP/Java Exceptions/10marks

Exceptions to Implement:

Checked Exceptions ::

1. IOException: Simulate a scenario like reading a file from disk that might cause an input/output error. o Example: Attempt to read a non-existent file.

```
import java.io.*;

public class IOException {
   public static void main(String[] args) {
      try {
         // Attempt to read a file that does not exist
         FileReader reader = new FileReader("nonexistentfile.txt");
      } catch (IOException e) {
         // Handle the exception
         System.out.println("IOException caught: " + e.getMessage());
      }
   }
}
```

2. FileNotFoundException: A specific type of IOException that occurs when a file is not found. o Example: Try to open a file that doesn't exist.

```
import java.io.*;

public class FileNotFoundException {
   public static void main(String[] args) {
      try {
         // Specific case of IOException
         FileInputStream file = new FileInputStream("missingfile.txt");
      } catch (FileNotFoundException e) {
         System.out.println("FileNotFoundException caught: " + e.getMessage());
      }
   }
}
```

3. EOFException: Simulate reaching the end of a file unexpectedly. o Example: Attempt to read beyond the file's content.

```
import java.io.*;
public class EOFException{
   public static void main(String[] args) {
      try
 // Simulate EOFException by reading beyond data in a stream
         DataInputStream data = new DataInputStream(new FileInputStream("emptyfile.txt"));
         data.readUTF();
      } catch (EOFException e) {
         System.out.println("EOFException caught: " + e.getMessage());
      } catch (IOException e) {
         System.out.println("IOException caught: " + e.getMessage());
      }
   }
}
```

4. SQLException: Simulate a database error. o Example: Attempt to connect to a non-existent database or execute invalid SQL.

```
import java.sql.*;
public class SQLException {
   public static void main(String[] args) {
      try {
         // Simulate database error
         Connection connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/nonexistentDB", "user", "password");
      } catch (SQLException e) {
         System.out.println("SQLException caught: " + e.getMessage());
      }
```

```
    }
}
```

5. ClassNotFoundException: Demonstrate a scenario where a class file is missing at runtime.

```
public class ClassNotFoundException {
    public static void main(String[] args) {
        try {
            // Simulate missing class error
            Class.forName("com.unknown.NonExistentClass");
        } catch (ClassNotFoundException e) {
            System.out.println("ClassNotFoundException caught: " + e.getMessage());
        }
    }
}
```

Unchecked Exceptions

6. ArithmeticException: Simulate an arithmetic operation that fails.

```
public class ArithmeticException {
    public static void main(String[] args) {
        try {
            int result = 10 / 0;
        } catch (ArithmeticException e) {
            System.out.println("ArithmeticException caught: " + e.getMessage());
        }
    }
}
```

7. NullPointerException: Access a null reference.

```
public class NullPointerException {
    public static void main(String[] args) {
        try {
            String str = null;
            System.out.println(str.length());
        } catch (NullPointerException e) {
            System.out.println("NullPointerException caught: " + e.getMessage());
        }
    }
}
```

8. ArrayIndexOutOfBoundsException: Access an invalid array index.

```
public class ArrayIndexOutOfBoundsException {
    public static void main(String[] args) {
        try {
            int[] numbers = {1, 2, 3};
            System.out.println(numbers[5]);
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("ArrayIndexOutOfBoundsException caught: " + e.getMessage());
        }
    }
}
```

9. ClassCastException: Demonstrate an invalid type cast.

```
public class ClassCastException {
    public static void main(String[] args) {
        try {
            Object obj = new String("test");
            Integer num = (Integer) obj;
        } catch (ClassCastException e) {
```

```
            System.out.println("ClassCastException caught: " + e.getMessage());
        }
    }
}
```

10. IllegalArgumentException: Pass an invalid argument to a method.

```
public class IllegalArgumentException {
    public static void main(String[] args) {
        try {
            Thread.sleep(-1);
        } catch (IllegalArgumentException | InterruptedException e) {
            System.out.println("IllegalArgumentException caught: " + e.getMessage());
        }
    }
}
```

11. NumberFormatException: Attempt to convert a string to a number when the format is invalid.

```
public class NumberFormatException {
    public static void main(String[] args) {
        try {
            int number = Integer.parseInt("invalidNumber");
        } catch (NumberFormatException e) {
            System.out.println("NumberFormatException caught: " + e.getMessage());
        }
    }
}
```