CSA0992 - Programming in Java

192011080
G.K. Gagana priya

Assignment - 4

① Exception Handling :-

The Exception Handling in Java is one of the powerful mechanism to handle the runtime errors so that the normal flow of the Application can be maintained.

Such as classNotFoundException, IOException, SQLException, RemoteException etc

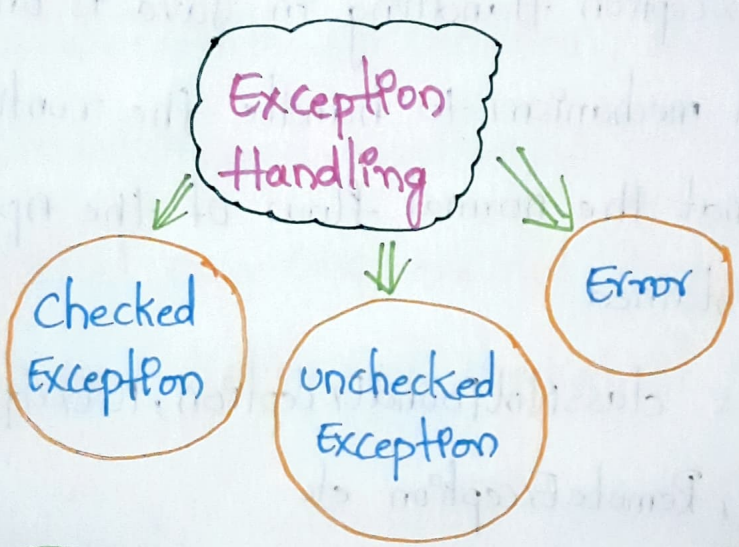
Advantage of Exception Handling

The core advantage of Exception handling is to maintain the normal flow of the application An exception normally disrupts the normal flow of the application.

The Java.lang.Throwable class is the root class of Java Exception hierarchy inherited by two classes.

Types of Java Exceptions:

1. checked Exception
2. unchecked Exception
3. Error



Checked Exception :-

The classes that directly inherit. The throwable class except RuntimeException and Error are known as 'checked exceptions.

for Example: IOException, SQLException etc.

checked Exceptions are checked at compile time

Unchecked Exception:-

The classes that inherit the RuntimeException are known as unchecked exceptions.

for Example: ArithmeticException, NullPointerException ArrayINdexoutofBoundsException, etc.

unchecked exceptions are not checked at compile time, but they are checked at runtime

Error:-

Error is Irrecoverable

Some Example of errors are outOfMemoryError, VirtualMachineError, AssertionError etc.

Java Exception Keywords

Java Exception keywords are 5 types:

* try
* catch
* finally

* throw

* throws.

Example on using Handling Arithmetic Exception:

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

Example on using finally :-

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
public class Example {
  public static void main(String[] args) {
    BufferedReader reader = null;
    try {
      reader = new BufferedReader(new FileReader
                ("Example.txt"));
      String line;
      while ((line = reader.readline()) != null)
{
  System.out.println(line);
 }
 } catch(IOException e) {
 System.out.println("An IOException occured :" + e.get
    message());
```

```
    }
    finally {
    try {
      if (reader! =null) {
          reader. close();
        }
      } catch (IOException e) {
      system.out. println ("Error closing the reader:"
        +e.getmessage());
       }
    }
  }
}
```