DEPARTMENT OF COMPUTER APPLICATIONS

I YEAR II SEM

Java Programming

UNIT I – Java Fundamentals

Exception Handling

Introduction- Exception handling

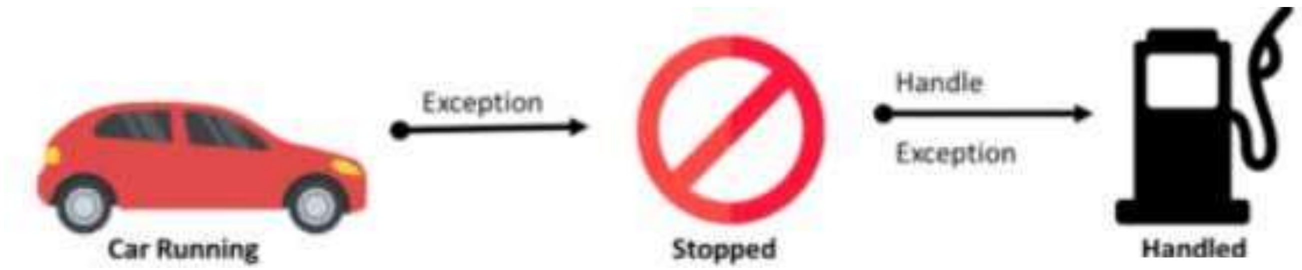
Bug

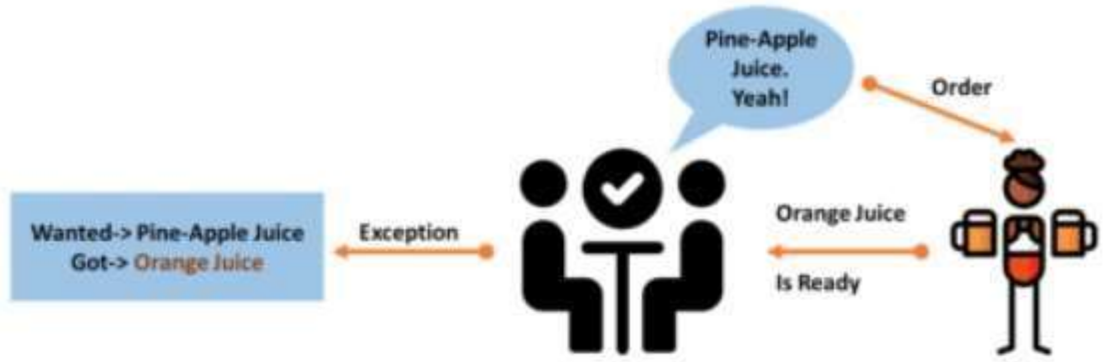
Debug

Exception Handling

Exception Handling

1. An Exception is an event that occurs during the execution of a program and it interrupts the normal flow of program executions.
2. Exception Handling is a mechanism to handle runtime errors such as ClassNotFoundException, IOException, SQLException, RemoteException, etc.





Some common problems which may cause exception

1. Creating array object with negative size.
2. Accessing index of array which is not available
3. Dividing an integer value with zero.
4. Invoking instance members with null reference.
5. Recursive method invocation without conditional check.

Exception handling - Types

1. Synchronous Exception – Errors such as “**Out-of-range index**” and “**Over-flow**” belong to the synchronous type exception.

2.Asynchronous Exception – The errors that are caused by events beyond the control of the program that is called Asynchronous Exception.

Error handling code that performs the following tasks:

1. Find the exception
2. Throw the exception
3. Catch the exception
4. Handle the exception

Exception Handling Mechanism

Exception Handling mechanism is basically built upon three keywords namely:

1. Try

2. Throw

3. Catch

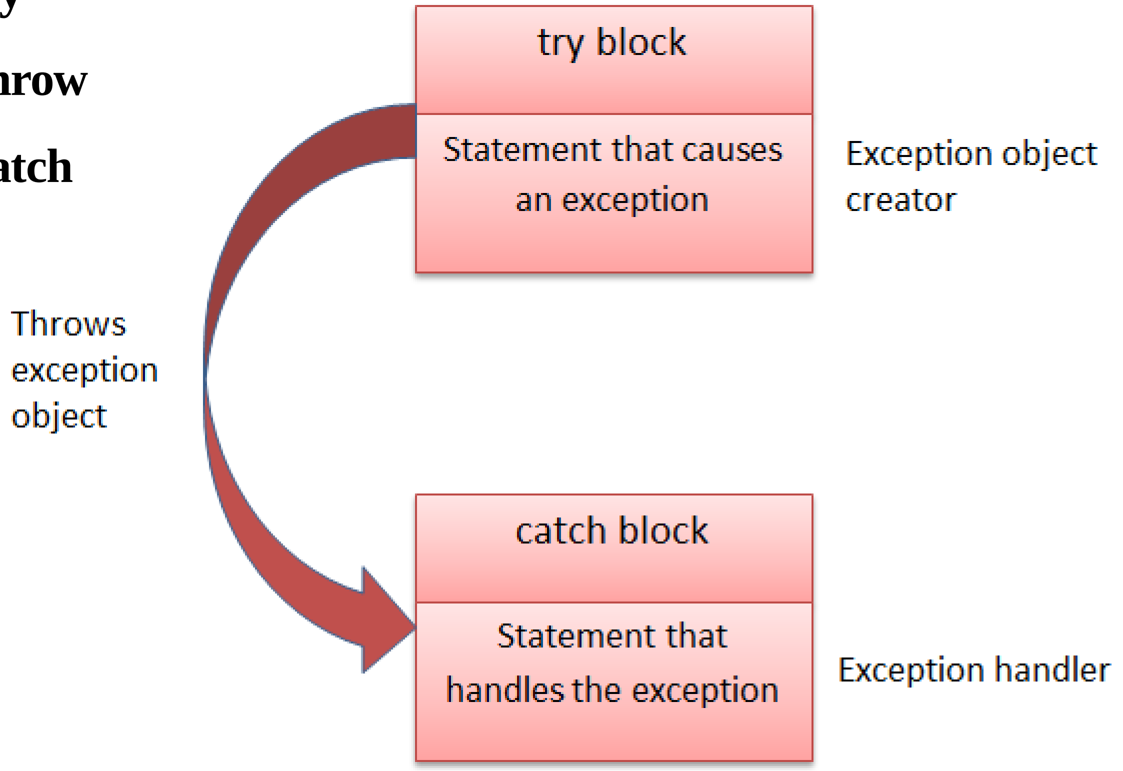
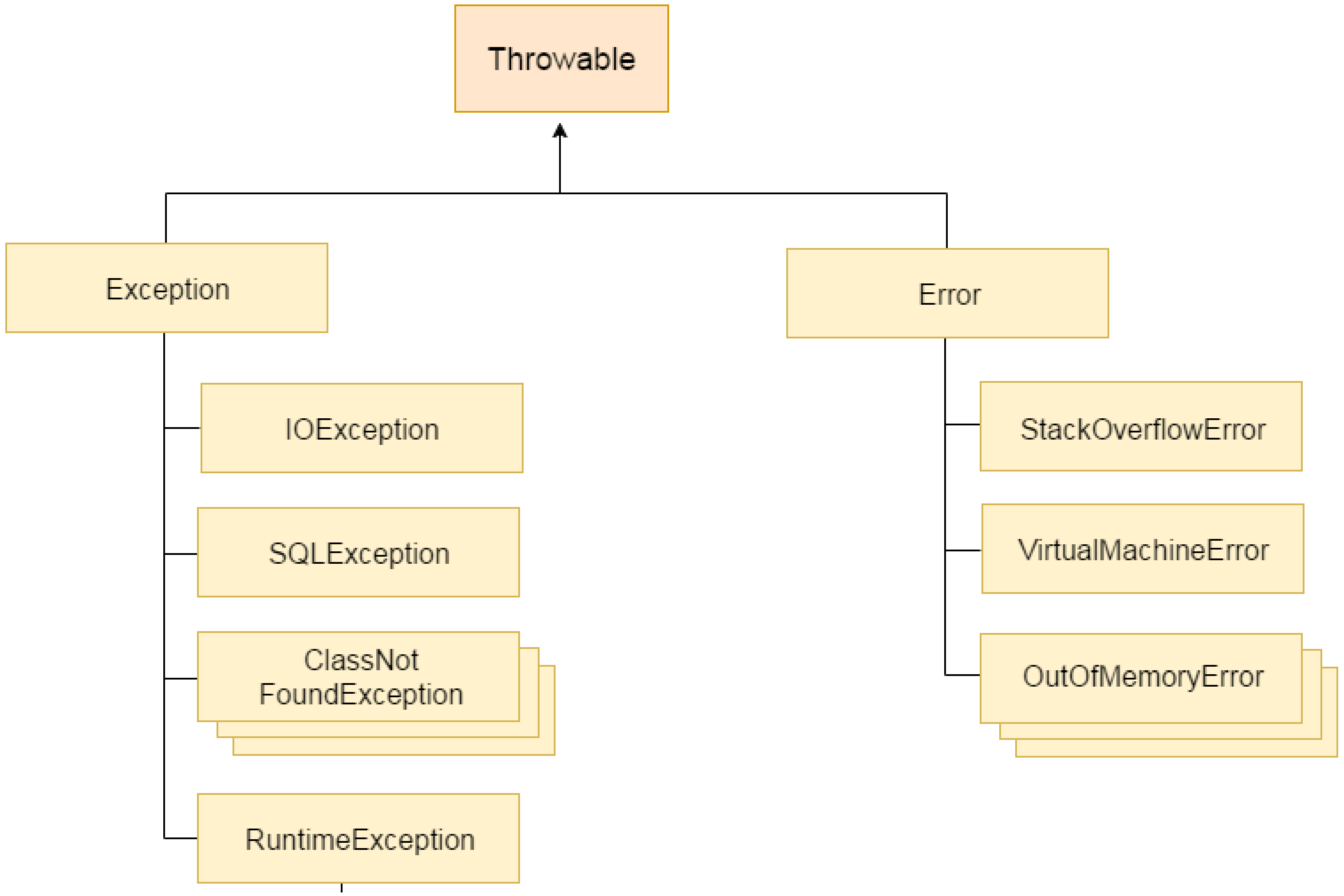


Fig: Exception Handling Mechanism

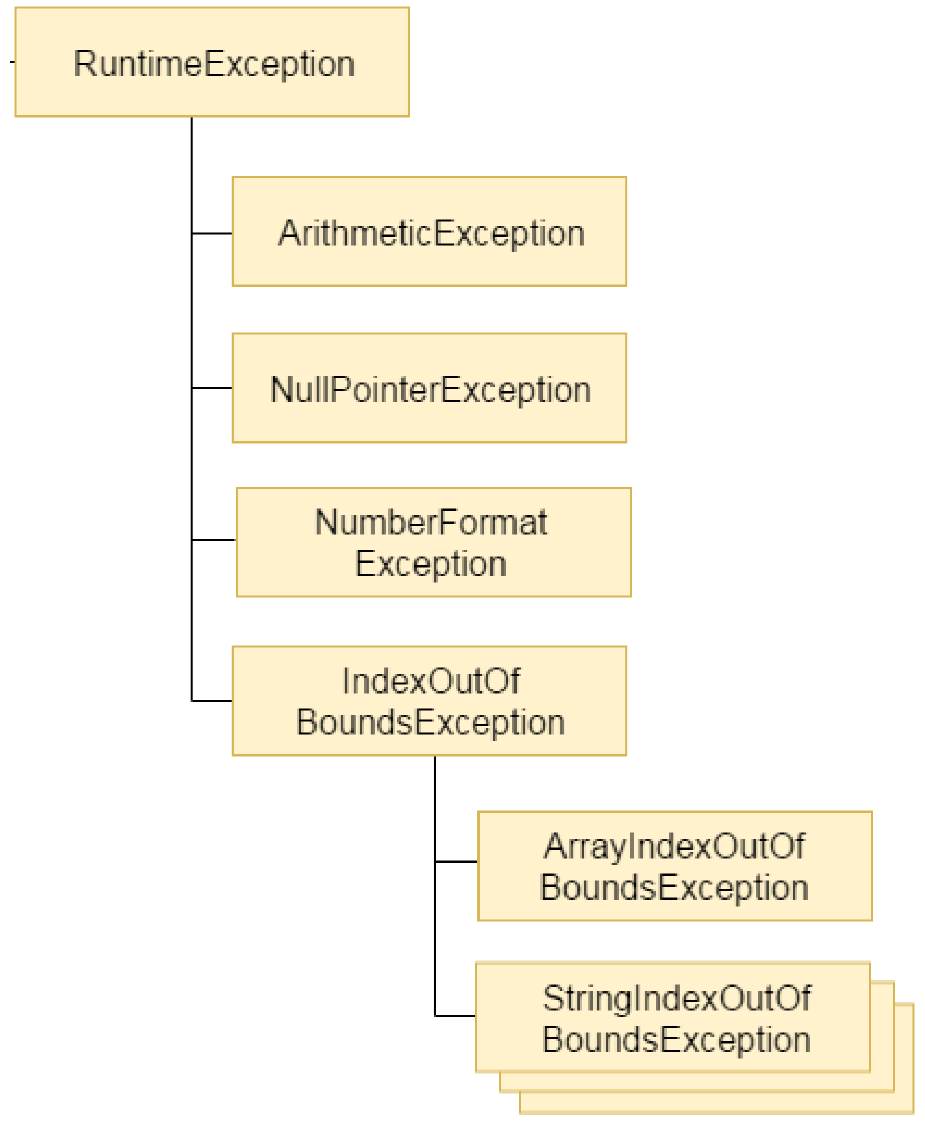
Syntax of Exception Handling:

```
.......
.......
try
{
statement; //  generates an exception
throw exception; //  throws an exception
.........
}
catch(Exception-type e)
{
statement; //  processes the exception
}
........
........
```

Hierarchy of Java Exception classes



Hierarchy of Java Exception classes


RuntimeException
ArithmeticException
NullPointerException
NumberFormat Exception
IndexOutOf BoundsException
ArrayIndexOutOf BoundsException
StringIndexOutOf BoundsException

Types of Java Exceptions

Three types of exceptions:

1. Checked Exception
2. Unchecked Exception
3. Error

Unchecked Exception
Checked Exception
Error
Exception Handling

Difference between Checked and Unchecked Exceptions

1) Checked Exception

The classes which directly inherit Throwable class except RuntimeException and Error are known as checked exceptions e.g. IOException, SQLException etc. Checked exceptions are checked at compile-time.

2) Unchecked Exception

The classes which inherit RuntimeException are known as unchecked exceptions e.g. ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException etc. Unchecked exceptions are not checked at compile-time, but they are checked at runtime.

3) Error

Error is irrecoverable e.g. OutOfMemoryError, VirtualMachineError, AssertionError etc.

Java Exception Handling Example

```
public class JavaExceptionExample{
  public static void main(String args[]){
  try{
      //code that may raise exception
      int data=100/0;
   }catch(ArithmeticException e){System.out.println(e);}
   //rest code of the program
   System.out.println("rest of the code...");
  }
}
```

Output:

```
Exception in thread main java.lang.ArithmeticException:/ by
zero
rest of the code...
```

Common Scenarios of Java Exceptions

A scenario where ArithmeticException occurs

If we divide any number by zero, there occurs an ArithmeticException.

```
int a=50/0;//ArithmeticException
```

A scenario where NullPointerException occurs

If we have a null value in any variable, performing any operation on the variable throws a NullPointerException.

```
String s=null;
System.out.println(s.length());//NullPointerException
```

A scenario where NumberFormatException occurs

The wrong formatting of any value may occur NumberFormatException. Suppose I have a string variable that has characters, converting this variable into digit will occur NumberFormatException.

```
String s="abc";
int i=Integer.parseInt(s);//NumberFormatException
```

A scenario where ArrayIndexOutOfBoundsException occurs

If you are inserting any value in the wrong index, it would result in ArrayIndexOutOfBoundsException as shown below:

```
int a[]=new int[5];
a[10]=50; //ArrayIndexOutOfBoundsException
```

Reference

1. Herbert Schildt " The Complete Reference Java 2, 8th edition , Tata McGraw Hill, 2011
2. Ralph Bravaco, Shai Simonson, "Java Programming: From the Ground up Tata McGraw Hill, 2012
3. https://www.javatpoint.com/try-catch-block