Object Oriented Programming
CS F213
Amit Dua

BITS Pilani
Pilani Campus

Slides Taken from the slides prepared by Dr. Jennifer

innovate achieve lead

BITS Pilani

Pilani Campus

Exception handling

Uncaught Exceptions

- When the Java run time detects an exception, it constructs an object and throws the exception.
- If there are no exception handlers in the program, the exception is caught by the default exception handler.
- User defined catch using catch keyword.
- Finally keyword.

Example

```
package class1;

public class test  {
public static void main(String args[]) {
int b = 30,c = 0;
System.out.println(b/c);
}
}
```

Console:

```
Exception in thread "main" java.lang.ArithmeticException: / by zero
at class1.test.main(test.java:5)
```

Try and catch

```
public class Testtrycatch2{
  public static void main(String args[]){
   try{
      int data=50/0;
   }catch(ArithmeticException e){System.out.println(e);}
   System.out.println("rest of the code...");
}
}
```

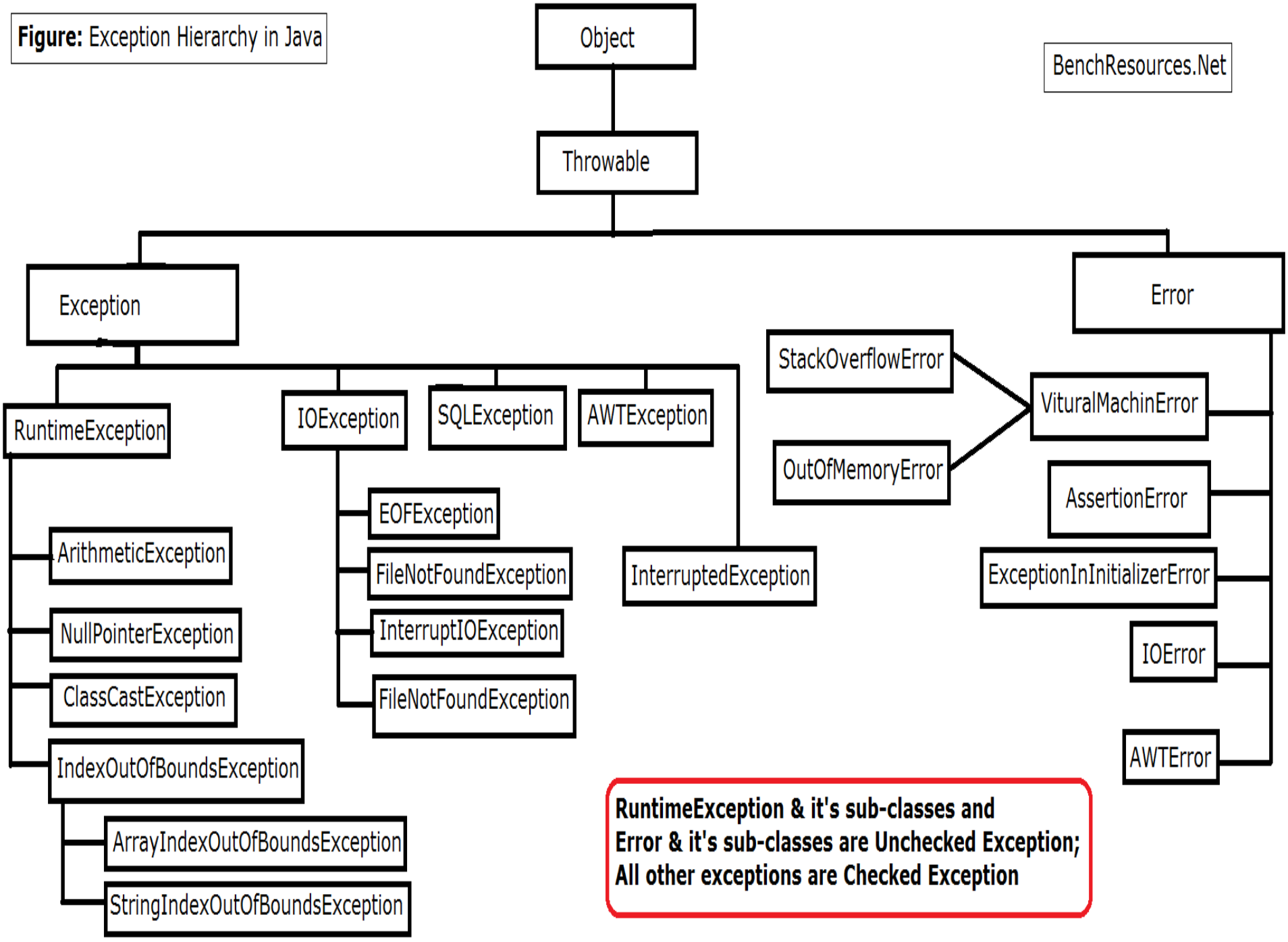
```
Exception in thread main java.lang.ArithmeticException:/ by
    zero
rest of the code...
```

try catch finally

```
class TryCatchFinallyExample {
    public static void main (String[] args){
 int[] arr = new int[4];
try        {
 int i = arr[4];
// this statement will never execute
 // as exception is raised by above
    statement
 System.out.println("Inside try block"); }
catch(ArrayIndexOutOfBoundsException
    ex){
System.out.println("Exception caught in
    catch block"); }
finally
      {
            System.out.printl
n("finally block
executed");
        }

        // rest program will
be executed
        System.out.println("
Outside try-catch-finally
clause");
    }
}
```

Figure: Exception Hierarchy in Java



Key Concepts

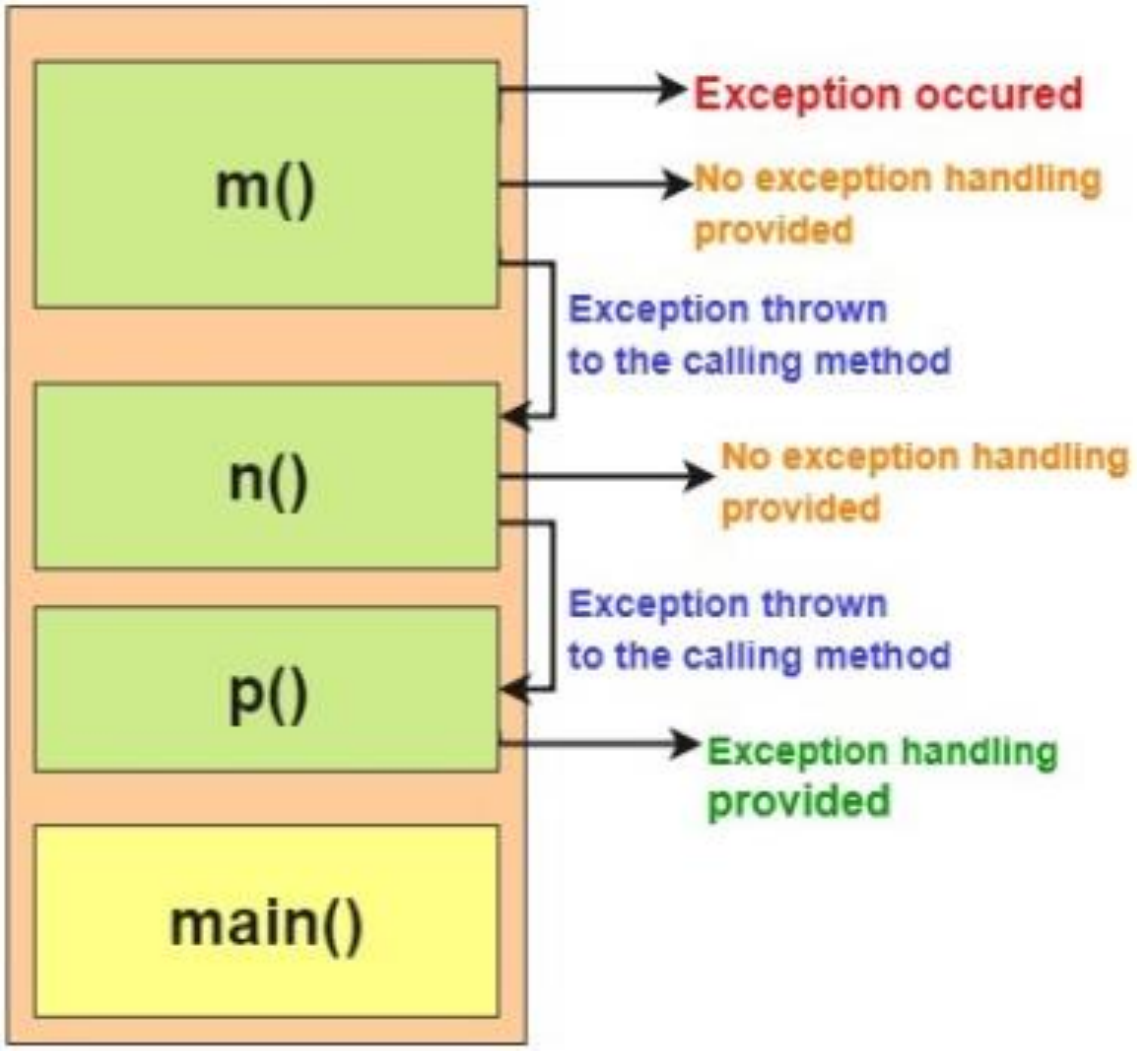
- Displaying an exception
  - Throwable overrides the toString(), and
  - the exception object can be passed to println() and
  - the description of the exception gets printed.

Multiple try-catch

```
public class TestMultipleCatchBlock{
  public static void main(String args[]){
   try{
    int a[]=new int[5];
    a[5]=30/0;     }
   catch(ArithmeticException e){System.out.println("task1 is
    completed");}
   catch(ArrayIndexOutOfBoundsException
    e){System.out.println("task 2 completed");}
   catch(Exception e){System.out.println("common task
    completed");}
     System.out.println("rest of the code...");
}
```

```
class TestExceptionPropagation1{
  void m(){       int data=50/0;    }
  void n(){       m();     }
  void p(){
   try{       n();
   }catch(Exception e){System.out.println("exception
    handled");}    }
  public static void main(String args[]){
   TestExceptionPropagation1 obj=new
    TestExceptionPropagation1();
   obj.p();
   System.out.println("normal flow...");     }   }
```

Memory Stack



Exception -5

- try
- catch
- finally
- throw
- throws

Key Concepts

- ‘throw’ keyword is used to throw an exception explicitly.
- But object thrown should be of type Throwable or subclass of throwable. Primitive types such as int, char … and non throwable classes like String can not be used.
- Many built in run-time exceptions have atleast two constructors, one that takes a string parameter describing the exception and one no argument constructor.
  - e.getMessage() can also be used.

Throw

```
public class TestThrow1{
   static void validate(int age){
     if(age<18)
      throw new ArithmeticException("not valid");
     else
      System.out.println("welcome to vote");
   }
   public static void main(String args[]){
      validate(13);
      System.out.println("rest of the code...");
  }
}
```

Key Concepts

- ‘throws’ keyword is used to specify the callers of the method, when it is capable of causing an exception and it does not handle.

throws

```
import java.io.IOException;
class Testthrows1{    void m()throws IOException{
 throw new IOException("device error");//checked
    exception     }
  void n()throws IOException{       m();     }
  void p(){      try{     n();
   }catch(Exception e){System.out.println("exception
    handled");}     }
  public static void main(String args[]){
   Testthrows1 obj=new Testthrows1();
   obj.p();
   System.out.println("normal flow...");     }   }
```

```
import java.io.IOException;
class Testthrows1{    void m() {
 throw new IOException("device error");//checked
    exception     }
  void n()throws IOException{       m();     }
  void p(){      try{     n();
   }catch(Exception e){System.out.println("exception
    handled");}     }
  public static void main(String args[]){
   Testthrows1 obj=new Testthrows1();
   obj.p();
   System.out.println("normal flow...");     }   }
```

Throw vs throws

No.	throw	throws
1)	Java throw keyword is used to explicitly throw an exception.	Java throws keyword is used to declare an exception.
2)	Throw is followed by an instance.	Throws is followed by class.
3)	Throw is used within the method.	Throws is used with the method signature.
4)	You cannot throw multiple exceptions.	You can declare multiple exceptions e.g. public void method()throws IOException,SQLException.

User Defined Exception

- Helps the user to create their own exception for situations specific to the applications.
- The user defined exception class is derived from the Exception class
- Exception class does not have methods on its own. But it inherits methods provided by the Throwable class.
- Overriding the toString() method helps to provide a description about the exception.

Chained Exceptions

- Allows to associate another exception with an exception.
- The second exception describes the cause of the first exception.
- Chained exception methods supported by Throwable are getCause() and initCause()

Exception - Example

innovate achieve lead

```
class test
{
public static void main(String args[]) {
try{
NumberFormatException e =new NumberFormatException("Outer");
e.initCause(new ArithmeticException("Inner"));
throw e;
}
catch(NumberFormatException e) {
System.out.println(e);
System.out.println(e.getCause());
}
}
}
```

Extra Topic (only if you are interested) Difference between C++ and JAVA

- In C++, all types (including primitive and pointer) can be thrown as exception. But in Java only throwable objects (Throwable objects are instances of any subclass of the Throwable class) can be thrown as exception.
- In C++, there is a special catch called “catch all” that can catch all kind of exceptions. In Java, we can catch Exception object to catch all kind of exceptions.
- In Java, there is a block called finally that is always executed after the try-catch block. This block can be used to do cleanup work.
- In C++, all exceptions are unchecked. In Java, there are two types of exceptions – checked and unchecked.
- In Java, a new keyword *throws* is used to list exceptions that can be thrown by a function. In C++, there is no *throws* keyword, the same keyword *throw* is used for this purpose also.