BAB 9
EXCEPTION

A. POKOK BAHASAN

⊕ Exception handling

B. TUJUAN PEMBELAJARAN

- ✓ Memahami mengenai exception.
- ✓ Memahami tipe exception yaitu Checked Exception dan Unchecked Exception.
- ✓ Menangani exception menggunakan try, catch, finally, throw dan throws

B. DASAR TEORI

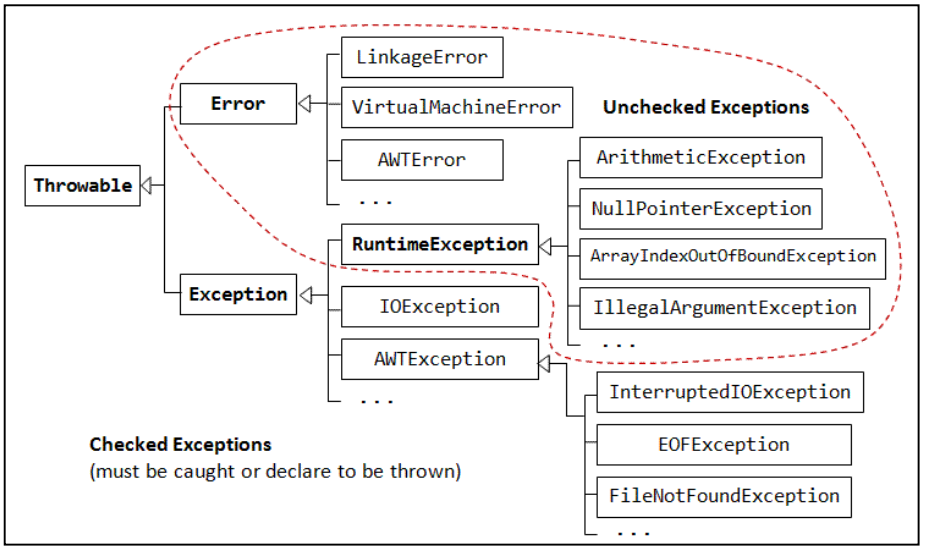
Exception adalah suatu kondisi abnormal yang terjadi pada saat menjalankan program. Teknik yang dipakai dalam Java untuk menangani kondisi yang tidak biasa saat menjalankan operasi normal dalam program dinamakan exception handling. Karena dalam java segala sesuatu merupakan objek, maka exception juga direpresentasikan dalam sebuah objek yang menjelaskan tentang exception tersebut. Contoh exception adalah pembagian bilangan dengan 0, pengisian elemen array diluar ukuran array, kegagalan koneksi database, file yang akan dibuka tidak ada, dan mengakses objek yang belum diinisialisasi. Terdapat dua penanganan exception yaitu:

a. Menangani sendiri exception tersebut.
b. Meneruskannya ke luar dengan cara membuat objek tentang exception tersebut dan melemparkannya (throw) keluar agar ditangani oleh kode yang memanggil method(method yang didalamnya terdapat exception) tersebut.

Ada lima keyword yang digunakan oleh Java untuk menangani exception yaitu try, catch, finally, throw dan throws.

- **Tipe-Tipe Exception**

  Pada exception, superclass tertinggi adalah class Throwable, tetapi kita hampir tidak pernah menggunakan class ini secara langsung. Dibawah class Throwable terdapat dua subclass yaitu class Error dan class Exception.



- **Penggunaan Blok try-catch**

  Untuk menangani exception dalam program, dengan meletakkan kode program yang memungkinkan terjadinya exception didalam blok try, diikuti dengan blok catch yang menentukan jenis exception yang ingin ditangani. Contoh :

```
public class Percobaan2 {
public static void main(String[] args)
{ int a[] = new int[5];
try{
a[5] = 100 ;
}catch(ArrayIndexOutOfBoundsException e){
System.out.println("Indeks Array melebihi
batas");
}
System.out.println("Setelah blok try-catch"); }
}
```

Output :

```
Terjadi exception karena Indeks Array melebihi batas Setelah blok try-catch
```

Dapat terjadi kode yang terdapat dalam blok try mengakibatkan lebih dari satu exception. Dalam hal ini, kita dapat menuliskan lebih dari satu blok catch. Contoh :

```
public class Percobaan5 {
public static void main(String[] args) {
int bil=10;
String b[] = {"a","b","c"};
try{
System.out.println(bil/0);
System.out.println(b[3]);
}catch(ArithmeticException e){
System.out.println("Error Aritmetik");
}catch(ArrayIndexOutOfBoundsException e){
System.out.println("Error  Kapasitas  Array  Melebihi
Batas");
}catch(Exception e){
System.out.println("Terdapat Error");
}
}
}
```

- **Menggunakan Keyword ”finally”**

Terdapat kode yang harus dijalankan walaupun terjadi atau tidak terjadi exception, misalkan kita membuka file, hal ini memungkinkan terjadinya exception misal file tidak ada, file tidak bisa dibuka, selanjutnya yang harus dilakukan adalah menutup file tersebut.

```
public class Percobaan2 {
public static void main(String[] args) {
int a[] = new int[5];
try{
a[5] = 100 ;
}catch(ArrayIndexOutOfBoundsException e){
System.out.println("Terjadi  exception  karena  Indeks
Array melebihi batas");
}finally{
System.out.println("Selalu Dijalankan");
} System.out.println("Setelah blok try-catch");
}
}
```

- **Menggunakan Keyword ”throw ” dan ”throws”**

  Secara eksplisit, kita dapat melempar (throw) exception dari program menggunakan keyword throw. Jika exception tersebut adalah checked exception, maka pada method harus ditambahkan throws. Jika exception tersebut adalah unchecked exception, maka pada method tidak perlu ditambahkan throws.

```
public class Percobaan6 {
public static void method1() throws
FileNotFoundException{
throw new FileNotFoundException("File Tidak Ada");
} public static void main(String[] args) {
try {
method1();
} catch (FileNotFoundException ex) {
System.out.println(ex.getMessage());}
} }
```

C. PERCOBAAN

- **Percobaan 1**

  Memahami cara menangkap Exception dengan tipe ArrayIndexOutOfBoundsException

```
public class Percobaan2 {
public static void main(String[] args) {
int a[] = new int[5];
try{
a[5] = 100 ;
} catch(ArrayIndexOutOfBoundsException e){
System.out.println("Indeks Array melebihi batas");
}
}
}
```

- **Percobaan 2**

  Jalankan percobaan 2, bagaimana output program? Perbaiki dengan Percobaan22 untuk menangani exception.

```
public class Percobaan2 {
public static void main(String[] args) {
int bil = 10 ;
System.out.println(bil/0);
}
}
```

```
public class Percobaan22 {
public static void main(String[] args) {
int bil = 10 ; try{
System.out.println(bil/0;
}catch(ArithmeticException e){
System.out.println("Tidak boleh membagi
bilangan dengan 0");
}
}
}
```

- **Percobaan 3**

  Memahami try bertingkat.

```
public class Percobaan4 {
public static void main(String[] args)
{ int bil = 10 ; try{
System.out.println(bil/0)
}catch(ArithmeticException e){
System.out.println("Terjadi exception karena
tidak boleh membagi bilangan dengan 0");
}catch(Exception e){
System.out.println("Terdapat Error");
}
}
}
```

- **Percobaan 4**

  Penggunaan finally

```
public class ExcepTest{
public static void main(String args[]){
int a[] = new int[2];
try{
System.out.println("Access element three :" + a[3]);
}catch(ArrayIndexOutOfBoundsException e){
System.out.println("Exception thrown :" + e);
}
finally{ a[0] = 6;
System.out.println("First element value: "
+a[0]); System.out.println("The finally statement
is executed");
}
}
}
```

D. LATIHAN

- **Latihan 1**
  Method yang melempar checked exception

```
import java.io.FileNotFoundException;
public class percobaan7 { public
static void method1() throws
FileNotFoundException{
throw new FileNotFoundException("File Tidak Ada");
}
public static void main(String[] args) {
try {method1();}catch (FileNotFoundException ex) {
System.out.println(ex.getMessage());
}}}
```

- **Latihan 2**
  Method yang melempar unchecked exception

```
public class percobaan {
public static void main (String args[]) {
Scanner sc = new Scanner(System.in);
try {
System.out.print ("Masukan Angka : ");
int num = sc.nextInt();
if (num>10) throw new Exception();
System.out.println("Angka kurang dari atau sama dengan
10");}
catch (Exception s) {
System.out.println("Angka lebih dari 10");}
System.out.println("Selesai");}}
```

- **Latihan 3**
  Menggunakan konsep Inheritance untuk membuat superclass dan subclass exception. Program menangani exception dengan menangkap subclass exception dengan superclass.

```
import javax.swing.*;
class ExceptionA extends Exception {}
class ExceptionB extends ExceptionA {}
class ExceptionC extends ExceptionB {}
public class Demo {
public static void main( String args[] )
{try {
throw new ExceptionC();}
catch( ExceptionA a ) {
JOptionPane.showMessageDialog( null,
a.toString(), "Exception",
JOptionPane.INFORMATION_MESSAGE ); }
try {
throw new ExceptionB(); } catch(
ExceptionA b ) {
JOptionPane.showMessageDialog(
null, b.toString(), "Exception",
JOptionPane.INFORMATION_MESSAGE );}
System.exit( 0 );}}
```