LAPORAN PRAKTIKUM
PEMROGRAMAN BERBASIS OBJEK
PERTEMUAN KE- 12

Nama : Riyan Risky Widya S
Nim : 185410103
Kelas : TI-2

STMIK AKAKOM YOGYAKARTA

A. TUJUAN

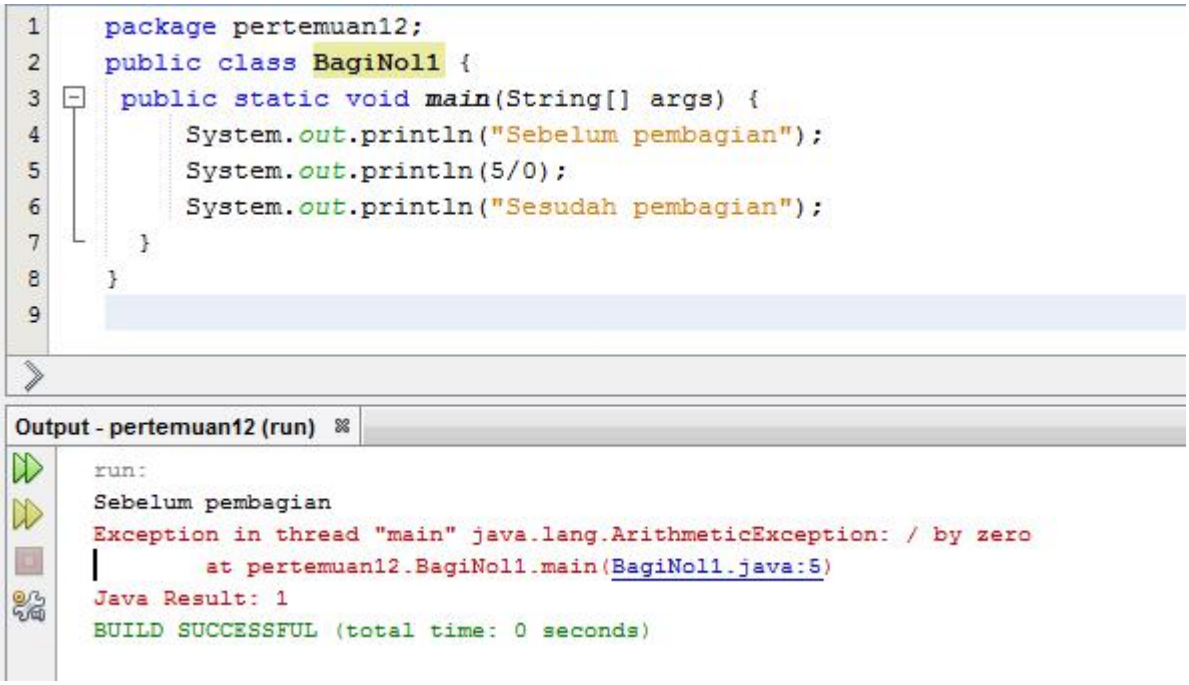
Dapat menggunakan perintah yang menangani Exception pada program,menggunakan throws untuk melempar eksepsi dan membuat Execption sendiri.

B. DASAR TEORI

Dalam pembuatan program seringkali dijumpai error atau kesalahan. Oleh karena itu, diperlukan suatu mekanisme yang membantu menangani error atau kesalahan yang terjadi, baik saat pembuatan maupun implementasi program. Exception adalah event yang terjadi ketika program menemui kesalahan pada saat instruksi program dijalankan. Banyak hal yang dapat menimbulkan event ini, misalnya crash, harddisk rusak dengan tiba-tiba, sehingga program program tidak bisa mengakses file-file tertentu. RunTimeException biasanya disebabkan oleh kesalahan program atau pada desain program. Misalnya NullPointerException yang disebabkan oleh proses inisialisasi program yang tidak sempurna dan arrayIndexOutOfBoundsException yang disebabkan akses array yang melebihi kapasitas array yang ada. Eksekusi program akan dihentikan segera setelah kalimat throw, kalimat-kalimat setelah kalimat throw tidak dieksekusi. Java akan melakukan inspeksi blok try terdekat untuk menemukan catch yang cocok dengan dengan tipe exception yang dilempar.

C. PEMBAHASAN

Praktik 1

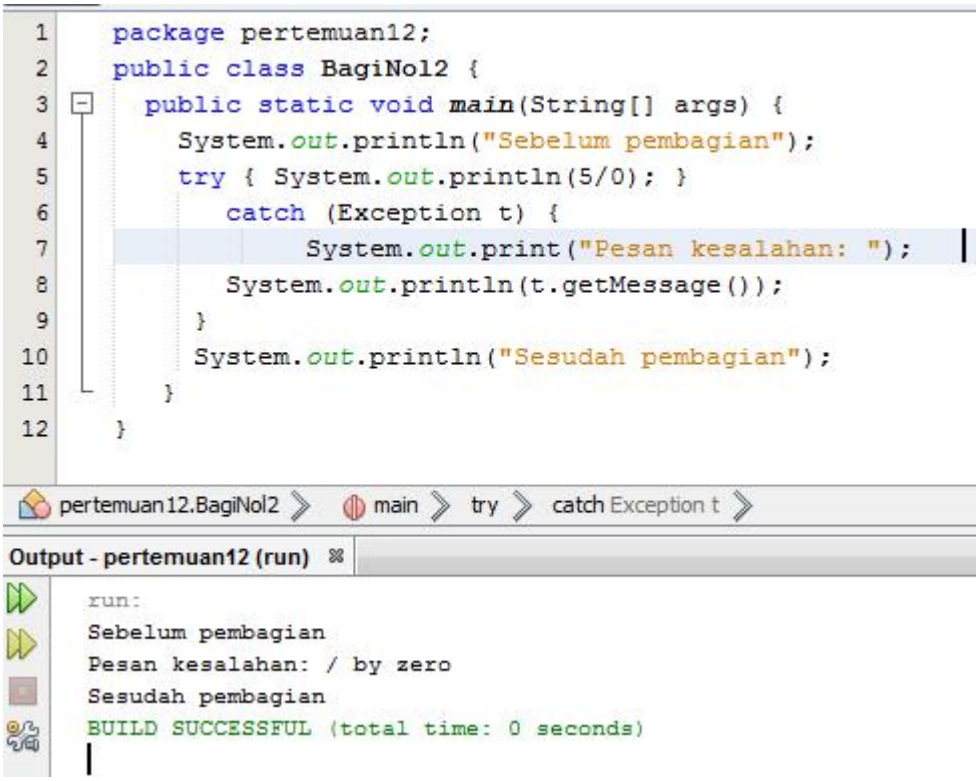


```
1  package pertemuan12;
2  public class BagiNol1 {
3   public static void main(String[] args) {
4        System.out.println("Sebelum pembagian");
5        System.out.println(5/0);
6        System.out.println("Sesudah pembagian");
7    }
8  }
9
```

```
Output - pertemuan12 (run)
run:
Sebelum pembagian
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at pertemuan12.BagiNol1.main(BagiNol1.java:5)
Java Result: 1
BUILD SUCCESSFUL (total time: 0 seconds)
```

Baris ke-6 tidak akan dieksekusi karena ada kesalahan pembagian dengan bilangan nol pada baris ke-6. sehingga output atau keluaranya hanya sebelum pembagian dan kalimat sesudah pembagian tidak ditampilkan. Jika error terjadi karena kesalahan aritmatika maka output keluaran errornya adalah java.lang.ArithmeticException: / by zero.

Praktik 2



```
1  package pertemuan12;
2  public class BagiNol2 {
3    public static void main(String[] args) {
4      System.out.println("Sebelum pembagian");
5      try { System.out.println(5/0); }
6        catch (Exception t) {
7            System.out.print("Pesan kesalahan: ");
8        System.out.println(t.getMessage());
9      }
10     System.out.println("Sesudah pembagian");
11   }
12 }
```

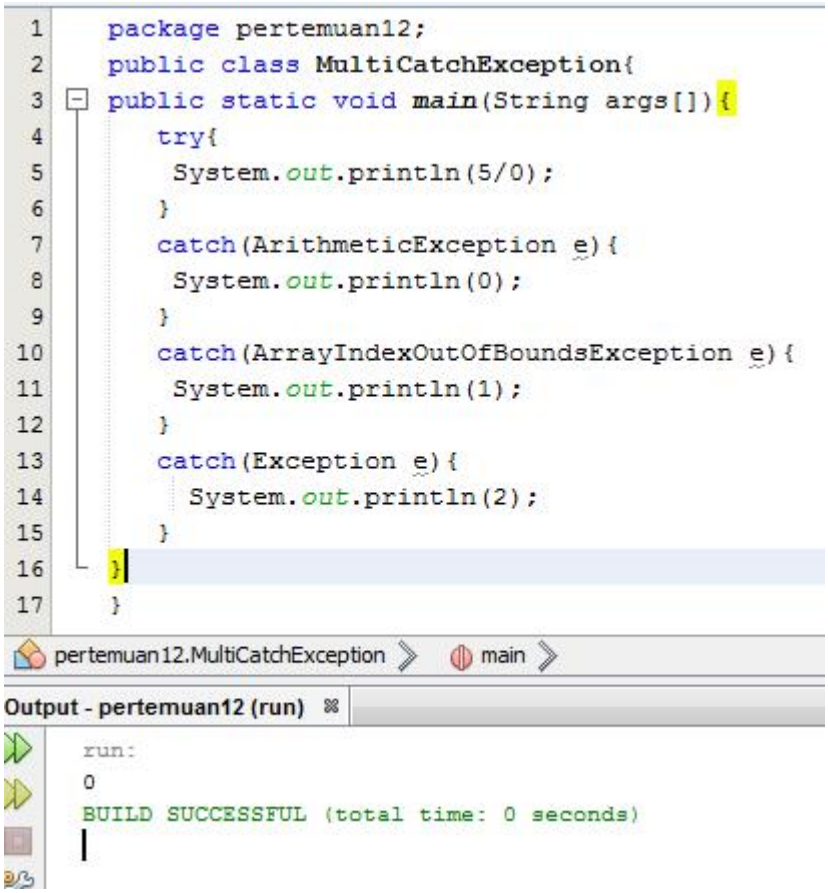
pertemuan12.BagiNol2 > main > try > catch Exception t

Output - pertemuan12 (run)

```
run:
Sebelum pembagian
Pesan kesalahan: / by zero
Sesudah pembagian
BUILD SUCCESSFUL (total time: 0 seconds)
```

Walaupun seperti praktik 1 tadi namun kali ini pada baris kode yang terjadi kesalahan diberikan catch and try maka baris program berikutnya akan di tetap dieksekusi dan sebagai gantinya yang error tadi adalah dengan menambah catchexception t lalu menampilkan pesan kesalahanya yaitu / by zero.

Praktik 3



```
1  package pertemuan12;
2  public class MultiCatchException{
3  public static void main(String args[]){
4     try{
5      System.out.println(5/0);
6     }
7     catch(ArithmeticException e){
8      System.out.println(0);
9     }
10    catch(ArrayIndexOutOfBoundsException e){
11     System.out.println(1);
12    }
13    catch(Exception e){
14      System.out.println(2);
15    }
16 }
17 }
```

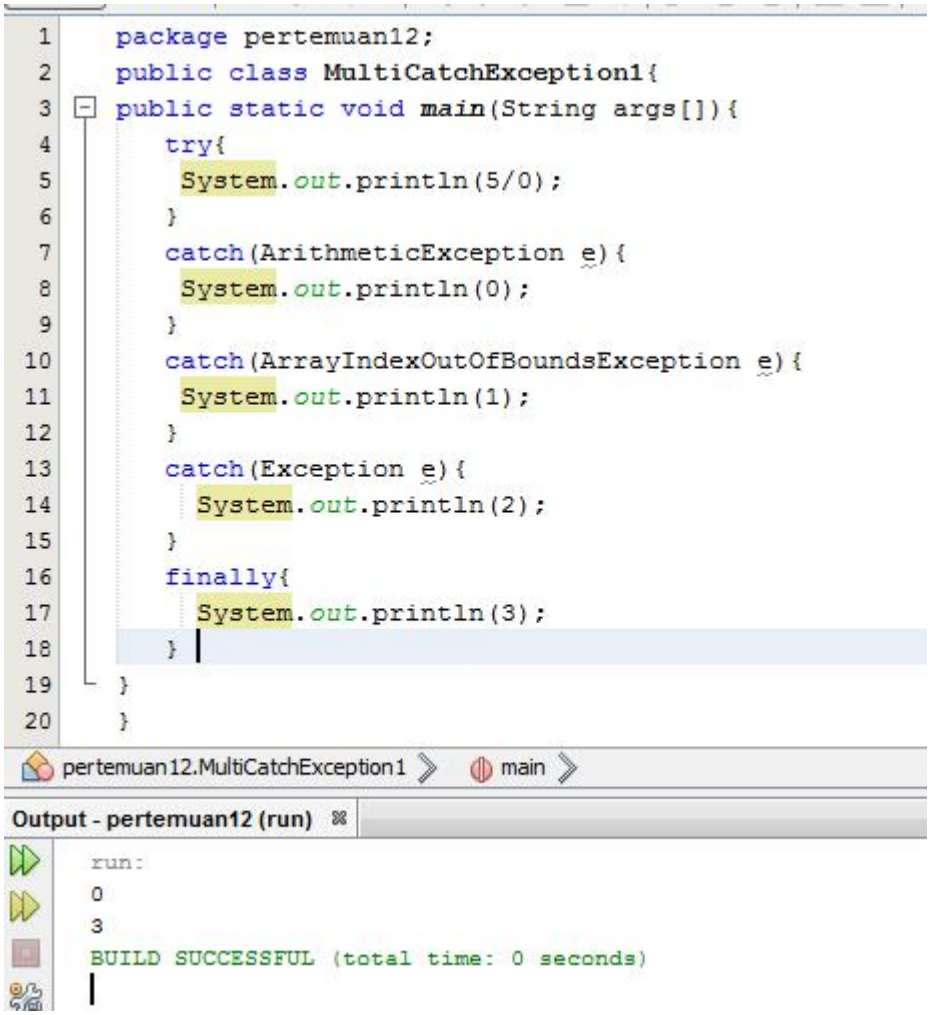
pertemuan12.MultiCatchException > main >

Output - pertemuan12 (run)

```
run:
0
BUILD SUCCESSFUL (total time: 0 seconds)
```

Yaitu pada baris ke 5 terjadi kesalahan karena bilangan yang dibagi dengan angka 0 itu tidak bisa di jalankan sehingga output adalah angka nol yang artinya pada blok try terdapat kesalahan arithmetic maka kesalahan tersebut ditangkap oleh blok catch dari kelas ArithmeticException.

Praktik 4



```
1  package pertemuan12;
2  public class MultiCatchException1{
3  public static void main(String args[]){
4      try{
5       System.out.println(5/0);
6      }
7      catch(ArithmeticException e){
8       System.out.println(0);
9      }
10     catch(ArrayIndexOutOfBoundsException e){
11      System.out.println(1);
12     }
13     catch(Exception e){
14       System.out.println(2);
15     }
16     finally{
17       System.out.println(3);
18     }
19 }
20 }
```

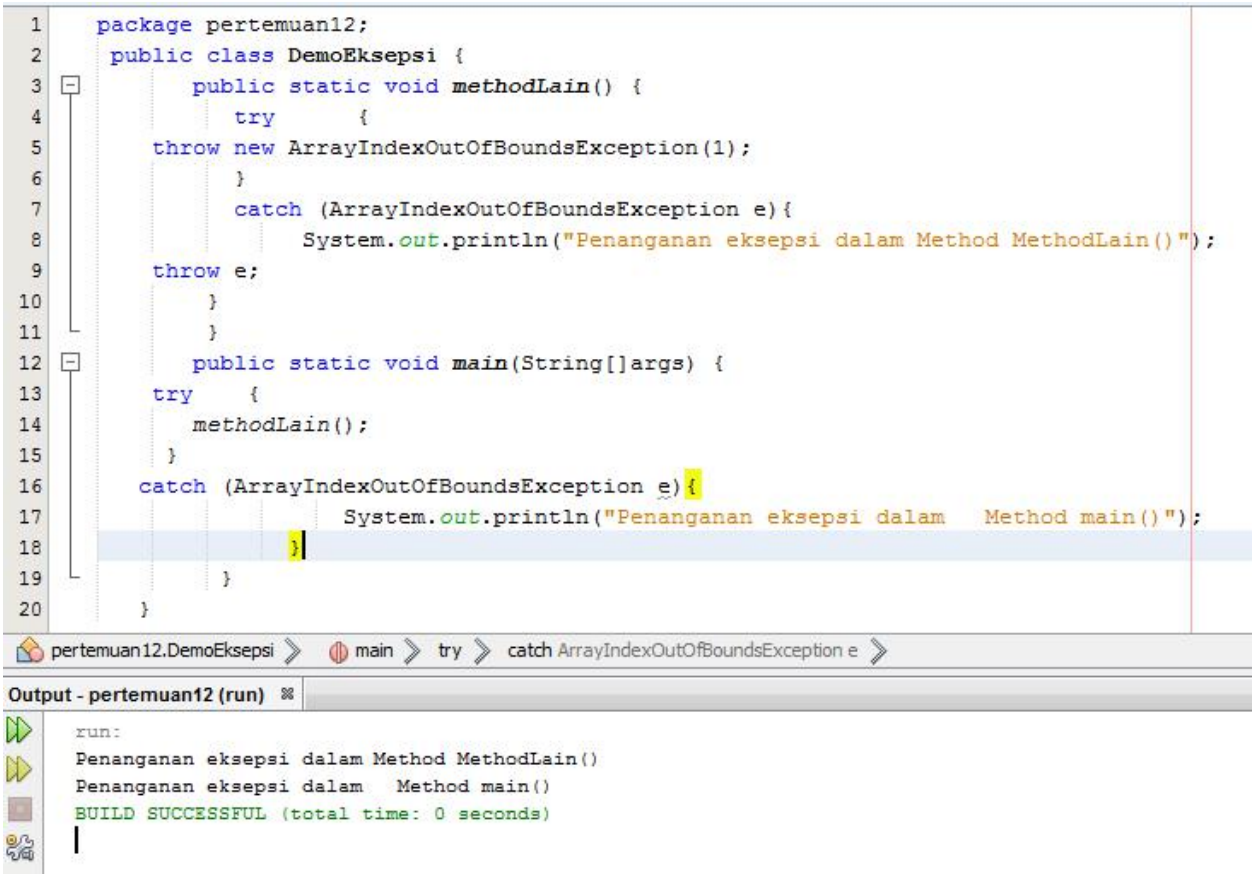
pertemuan12.MultiCatchException1 > main >

Output - pertemuan12 (run)

```
run:
0
3
BUILD SUCCESSFUL (total time: 0 seconds)
```

Pada baris ke 5 terjadi kesalahan dimana bilangan di bagi 0 maka dengan catch di tangkapp oleh arithmetiexception sehingga agian blok finally adalah bagian blok yang selalu dikerjakan. Output program adalah angka nol dan tiga, artinya setelah salah satu blok catch dieksekusi maka alur program selanjutnya adalah mengeksekusi bagian blok finally.

Praktik 5



```
package pertemuan12;
 public class DemoEksepsi {
       public static void methodLain() {
          try      {
    throw new ArrayIndexOutOfBoundsException(1);
          }
          catch (ArrayIndexOutOfBoundsException e){
              System.out.println("Penanganan eksepsi dalam Method MethodLain()");
    throw e;
        }
        }
       public static void main(String[]args) {
    try    {
       methodLain();
     }
   catch (ArrayIndexOutOfBoundsException e){
                  System.out.println("Penanganan eksepsi dalam   Method main()");
                }
         }
    }
```

pertemuan12.DemoEksepsi > main > try > catch ArrayIndexOutOfBoundsException e

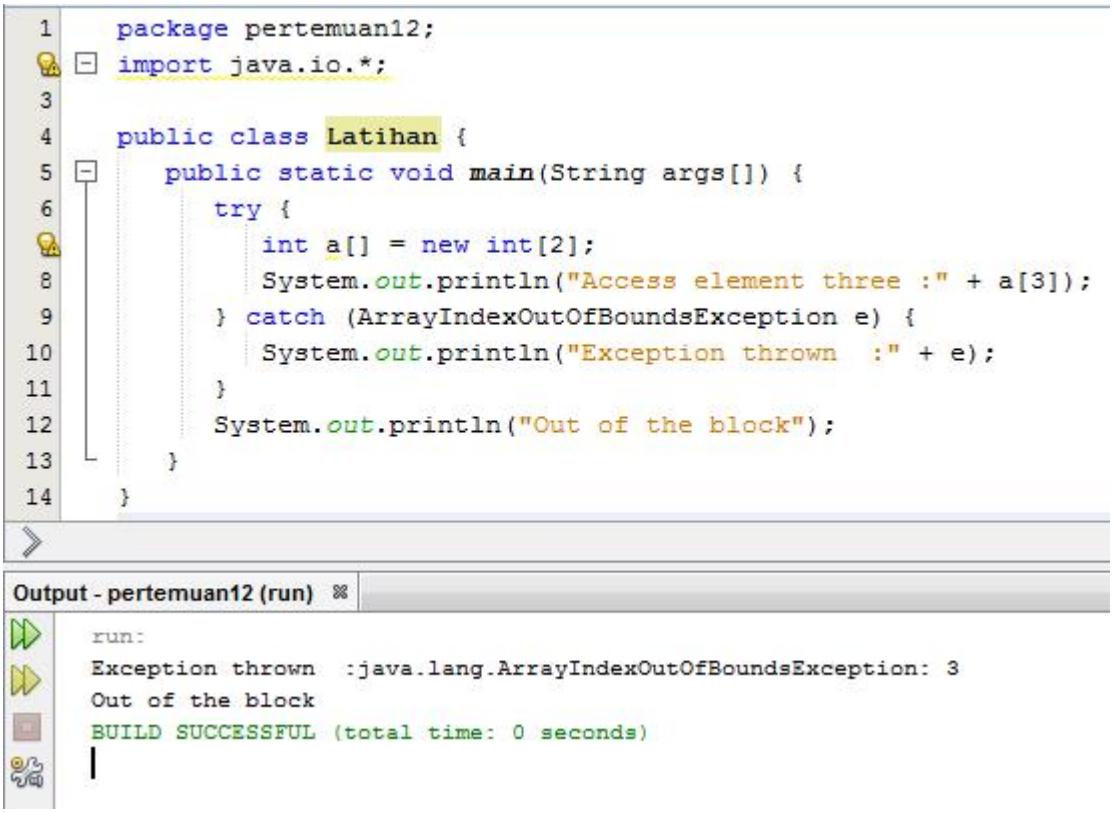
Output - pertemuan12 (run)

```
run:
Penanganan eksepsi dalam Method MethodLain()
Penanganan eksepsi dalam   Method main()
BUILD SUCCESSFUL (total time: 0 seconds)
```

Pada class DemoEksepsi dibuat method dengan nama methodLain yang didalamya terdapat try catch. Pada try membuat new Arrayindexoutofboundsexception(1) sehingga catch dijalankan dan pada baris berikutnya adalah menampilkan tulisan penanganan eksepsi dalam method methodLain(). laly pada method main membuat try lagi dengan memanggil method methodLain() lalu pada catch dijalankan tulisan penanganan eksepsi dalam method main().

Latihan

1 .Buat 2 contoh program menggunakan class exception selain yang telah dipraktikkan.



```
1   package pertemuan12;
2   import java.io.*;
3
4   public class Latihan {
5       public static void main(String args[]) {
6           try {
7               int a[] = new int[2];
8               System.out.println("Access element three :" + a[3]);
9           } catch (ArrayIndexOutOfBoundsException e) {
10              System.out.println("Exception thrown  :" + e);
11          }
12          System.out.println("Out of the block");
13      }
14  }
```

```
Output - pertemuan12 (run)
run:
Exception thrown  :java.lang.ArrayIndexOutOfBoundsException: 3
Out of the block
BUILD SUCCESSFUL (total time: 0 seconds)
```

Class exception yang pertama digunakan adalah arrayIndexOutOfBoundException. Class ini menangani dimana jika kita memanggil array lebih dari data yang dapat disimpan oleh array tersebut. Sehingga kode blok programnya akan seperti itu dan outputnya seperti pada diatas.

D. TUGAS

1. Jelaskan perbedaan kelas error dengan kelas exception.

- Error berhubungan dengan lingkungan di mana aplikasi berjalan sementara itu Exception berhubungan dengan aplikasi itu sendiri.

- Compiler tidak memiliki pengetahuan mengenai *unchecked* exception, termasuk Error dan semua subclass dari *RunTimeException* karena mereka tidak muncul pada saat runtime program. Compiler hanya memiliki pengetahuan untuk *checked* exception saja. Oleh karena itu, compiler akan tetap memaksa programmer untuk menyertakan blok try-catch bila ada pernyataan-pernyataan yang mungkin saja bisa melemparkan *checked* exceptions.

- Kategori Exception pada Java dibagi meenjadi dua, yaitu *checked* dan *unchecked*. Sementara itu, semua Error termasuk dari kategori *unchecked* saja.

- Error tidak akan bisa ditangani oeh blok try-catch. Meskipun anda mencoba untuk menanganinya menggunakan blok try-catch tersebut, namun aplikasi anda tidak akan pulih ketika error tersebut terjadi. Kebalikannya, Exception dapat ditangani oleh blok try-catch sehingga dapat membuat program tetap berjalan norman jika exception ini muncul.

- Memulihkan Error adalah hal yang sangat tidak mungkin. Satu-satunya cara yang mungkin dilakukan adalah mengehentikan program. Di sisi lainnya, Exception dapat dipulihkan dengan menggunakan blok try-catch atau melemparkan exception kembali kepada *caller*.

2. sebutkan kegunaan dari exception handling.

Keuntungannya dalah memungkinkan sebuah method untuk melemparkan *exception* pada pemanggilnya atau *caller* dan memungkinkan *caller* untuk menangani *exception* tersebut.

Dapat dibayangkan bila tanpa kapabilitas tersebut method yang dipanggil itu sendiri harus menangani *exception* atau menghentikan program. Seringkali method yang dipanggil tidak mengetahui apa yang harus dilakukan ketika terjadi error. Ini adalah kasus yang tipikal untuk method-method library. Library method dapat mendeteksi error namun hanya *caller* yang dapat mengetahui apa yang harus dilakukan ketika error muncul.

E. KESIMPULAN

Pada pemrograman java terdapat sebuah cara untuk menangani error saat program dijalankan dan mengarahkan ke sebuah blok perogram yang dimaui. Ini disebut error handling.