Nama : Kelvianto Pratama Harum

NIM : 200210500016

MK : Pemrograman Lanjut

Dosen : Muhammad Fajar B, S.Pd., M.Cs.

PERTEMUAN VI

FILE AND EXCEPTION HANDLING

A. Praktikum 6.1 – ContohOutput.java

Source code:

```java
package com.example.Pertemuan6;
//Praktikum 6.1
public class P1_ContohOutput {
    public static void main(String[] args) {
        int varInt = 45;
        double varDouble = 7.98845d;
        float varFloat = 6.97f;
        char varChar = 'X';
        String varString = "ini sebuah String";

        System.out.printf("%d\n", varInt);
        System.out.printf("%e\n", varDouble);
        System.out.printf("%f\n", varFloat);
        System.out.printf("%.2f\n", varFloat);
        System.out.printf("%c\n", varChar);
        System.out.printf("%.3s\n", varString);
        System.out.printf("%s\n", varString);

        System.out.println("======================");
        System.out.print("ini adalah ");
        System.out.print("contoh penggunaan print");

        System.out.println();
        System.out.println("======================");
        System.out.println("ini adalah ");
        System.out.println("contoh penggunaan print");
    }
}
```

Output:

```
> Task :app:run
45
7.988450e+00
6.970000
6.97
X
ini
ini sebuah String
======================
ini adalah contoh penggunaan print
======================
ini adalah
contoh penggunaan print

BUILD SUCCESSFUL in 2s
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada praktikum ini dibuat sebuah class dengan nama **P1_ContohOutput** dengan sebuah **main** method.

Pada bagian **main** method, dideklarasikan sejumlah variabel, yaitu:

- **varInt** dengan tipe data **integer** dan nilai **45**
- **varDouble** dengan tipe data **double** dan nilai **7.98845d**
- **varFloat** dengan tipe data **float** dan nilai **6.97f**
- **varChar** dengan tipe data **char** dan nilai **'X'**
- **varString** dengan tipe data **String** dan nilai **"ini sebuah String"**

Kemudian variabel tersebut dioutput ke console dengan **printf** yaitu method yang menampilkan data sesuai dengan format string yang diberikan:

- Variabel **varInt** dioutput dengan format **'%d'** yaitu format yang digunakan untuk menampilkan data **integer (decimal, base-10)**, sehingga data yang ditampilkan pada console adalah **45** seperti pada output.
- Variabel **varDouble** dioutput dengan format **'%e'** yaitu format yang digunakan untuk menampilkan data **float atau double dengan notasi scientific**, sehingga data yang ditampilkan pada console adalah **7.988450e+00** (ini sama seperti **7.988450 x 10^0)** seperti pada output.
- Variabel **varFloat** dioutput dengan format **'%f'** yaitu format yang digunakan untuk menampilkan data **float**, sehingga data yang ditampilkan pada console adalah **6.970000** seperti pada output.
- Variabel **varFloat** dioutput dengan format **'%.2f'** yaitu format yang digunakan untuk menampilkan data **float dengan 2 angka setelah koma**, sehingga data yang ditampilkan pada console adalah **6.97** seperti pada output.
- Variabel **varChar** dioutput dengan format **'%c'** yaitu format yang digunakan untuk menampilkan data **char (atau character)**, sehingga data yang ditampilkan pada console adalah **'X'** seperti pada output.
- Variabel **varString** dioutput dengan format **'%.3s'** yaitu format yang digunakan untuk menampilkan data **String sebanyak 3 karakter dari awal**, sehingga data yang ditampilkan pada console adalah **ini** seperti pada output.
- Variabel **varString** dioutput dengan format **'%s'** yaitu format yang digunakan untuk menampilkan data **String**, sehingga data yang ditampilkan pada console adalah **"ini sebuah String"** seperti pada output.

Kemudian dioutput string berikut:

- **“==========”** menggunakan method **println** yang artinya String yang diberikan akan diberikan **newline (\n) atau baris baru** pada akhir String.
- **“ini adalah “** menggunakan method **print** yang artinya String akan dioutput apa adanya tanpa modifikasi
- **“contoh penggunaan print”** menggunakan method **print** yang artinya String akan dioutput apa adanya tanpa modifikasi
- Sebuah baris kosong menggunakan **println**
- **“==========”** menggunakan method **println** yang artinya String yang diberikan akan diberikan **newline (\n) atau baris baru** pada akhir String.
- **“ini adalah “** menggunakan method **println** yang artinya String yang diberikan akan diberikan **newline (\n) atau baris baru** pada akhir String.
- **“contoh penggunaan print”** menggunakan method **println** yang artinya String yang diberikan akan diberikan **newline (\n) atau baris baru** pada akhir String.
-

B. Praktikum 6.2 – ContohInput.java

Source code:

```java
package com.example.Pertemuan6;
//Praktikum 6.2
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class P2_ContohInput {
    public static void main(String args[]){
        String nama = "";
        BufferedReader input = new BufferedReader(new InputStreamReader(System.in));
        System.out.println("Nama : ");
        try {
            nama = input.readLine();
        } catch(IOException e){
            e.printStackTrace();
        }
        System.out.println("Nama Anda : " + nama);
    }
}
```

Output:

```
> Task :app:run
Nama :
Kelvin Pratama
Nama Anda : Kelvin Pratama

BUILD SUCCESSFUL in 6s
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada praktikum ini dilakukan import **method** bawaan java seperti:

- **java.io.BufferedReader** dan **java.io.InputStreamReader** yang digunakan untuk membaca input keyboard user melalui console.
- **java.io.IOException** yang digunakan untuk menghandle exception terkait **input dan output** pada program.

Kemudian dibuat sebuah class dengan nama **P2_ContohInput** dengan sebuah method **main**.

Pada **main** method, dilakukan eksekusi sebagai berikut:

- Variabel **nama** dideklarasikan dengan tipe data **String** dan diinisialisasi dengan string kosong
- Kemudian dibuat sebuah object **input** dari class **BufferedReader** dan **InputStreamReader** yang digunakan sebagai interface antara input keyboard user dan program.
- Output sebuah string **"Nama: "** dengan method **println** yang artinya string akan diikuti dengan baris baru.
- Lalu, program akan mencoba mengambil **input** dari user lalu dimasukkan dalam variabel **nama**
- Jika program mendapatkan eksepsi **IOException**, eksepsi tersebut ditangkap ke dalam object **e**, lalu hasil tracing stack callnya ditampilkan dalam console.
- Lalu, program akan menampilkan string **"Nama Anda: "** diikuti dengan nilai variabel **nama**

C. Praktikum 6.3 – ContohInputScanner.java

Source code:

```
1   package com.example.Pertemuan6;
2   //Praktikum 6.3
3   import java.util.Scanner;
4
5   public class P3_ContohInputScanner {
6       public static void main(String args[]){
7           Scanner input = new Scanner(System.in);
8           System.out.println("Nama : ");
9           String nama = input.nextLine();
10          System.out.println("Nama Anda : " + nama);
11      }
```

Output:

```
> Task :app:run
Nama :
Kelvin Pratama
Nama Anda : Kelvin Pratama

BUILD SUCCESSFUL in 6s
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada praktikum ini dilakukan import **method** bawaan java seperti:

- **java.util.Scanner** yang digunakan untuk membaca input keyboard user melalui console.

Kemudian dibuat sebuah class dengan nama **P3_ContohInputScanner** dengan sebuah method **main**.

Pada **main** method, dilakukan eksekusi sebagai berikut:

- Dibuat sebuah object interface bernama **input** yang merupakan object dari class **java.util.Scanner**. Object ini digunakan untuk membaca input keyboard user melalui console.
- Ditampilkan string **"Nama: "** dengan method **println** yang artinya string diikuti dengan baris baru
- Lalu, program akan membaca input user pada object **input** menggunakan method **nextLine()** lalu nilainya disimpan dalam variabel **nama**
- Lalu, program akan menampilkan string **"Nama Anda: "** diikuti dengan nilai variabel **nama**

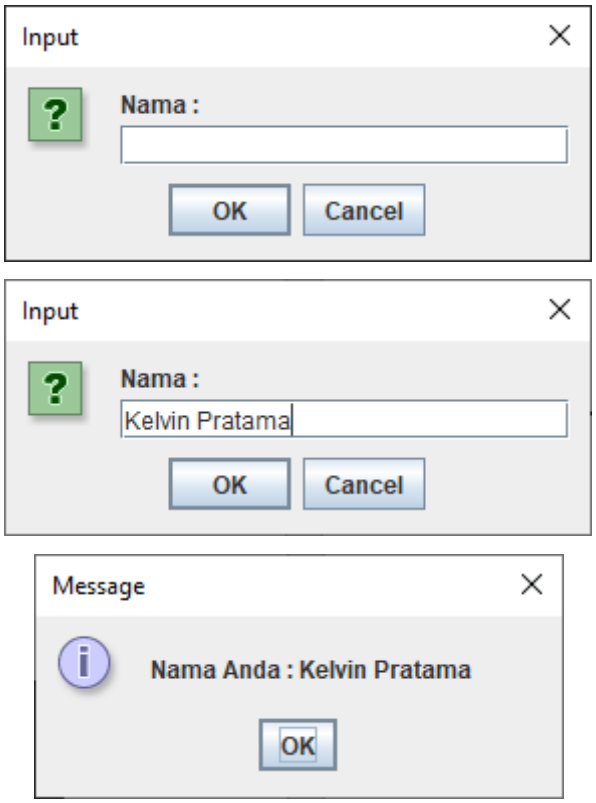
D. Praktikum 6.4 – ContohInputJOptionPane.java

Source code:

```java
package com.example.Pertemuan6;
//Praktikum 6.4
import javax.swing.JOptionPane;

public class P4_ContohInputJOptionPane {
    public static void main(String args[]){
        String nama = JOptionPane.showInputDialog(null, "Nama : ");
        JOptionPane.showMessageDialog(null, "Nama Anda : " + nama);
    }
}
```

Output:



Penjelasan:

Pada praktikum ini dilakukan import **method** bawaan java seperti:

- **javax.swing.JOptionPane** yang digunakan untuk komponen GUI bawaan Java

Kemudian dibuat sebuah class dengan nama **P4_ContohInputJOptionPane** dengan sebuah method **main**.

Pada **main** method, dilakukan eksekusi sebagai berikut:

- Sebuah panel GUI Input Dialog dari **JOptionPane** dibuat yang digunakan untuk menampilkan sebuah kolom textbox dengan placeholder **"Nama: "**
- Input dari Panel tersebut disimpan dalam sebuah variabel dengan nama **'nama'** dan tipe data **String**.
- Kemudian, sebuah panel message dialog ditampilkan berisikan string **"Nama Anda: "** diikuti dengan isi variabel **nama**.

E. Praktikum 6.5 – ContohFile.java

Source code:

```java
package com.example.Pertemuan6;
//Praktikum 6.5
public class P5_ContohFile {
    public static void main(String args[]){
        java.io.File file = new java.io.File(("java.txt"));
        try{
            if (file.createNewFile())
                System.out.println("File berhasil dibuat");
            else
                System.out.println("File Gagal di Buat");
        } catch (Exception e){
            System.out.println("Error");
        }
        System.out.println("Apakah file ada? " + file.exists());
        System.out.println("Apakah file bisa dibaca? " + file.canRead());
        System.out.println("Apakah file bisa ditulis? " + file.canWrite());
        System.out.println("Apakah berupa direktori? " + file.isDirectory());
        System.out.println("Apakah berupa file? " + file.isFile());
        System.out.println("Apa namanya? " + file.getName());
        System.out.println("Dimana lokasinya? " + file.getPath());
        System.out.println("Dimana lokasi lengkapnya? " + file.getAbsolutePath());
    }
}
```

Output:

```
> Task :app:run
File Gagal di Buat
Apakah file ada? true
Apakah file bisa dibaca? true
Apakah file bisa ditulis? true
Apakah berupa direktori? false
Apakah berupa file? true
Apa namanya? java.txt
Dimana lokasinya? java.txt
Dimana lokasi lengkapnya? D:\A KULIAH\Semester 8.2\Pemrograman Lanjut\P6 - File and Exception Handling\Project Files\Pertemuan6\app\java.txt

BUILD SUCCESSFUL in 2s
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada praktikum ini, dibuat sebuah class dengan nama **P5_ContohFile** dengan sebuah method **main**.

Pada **main** method, dilakukan eksekusi sebagai berikut:

- Dibuat sebuah object dengan nama **file** yang merupakan object dari **java.io.File** untuk file **java.txt**
- Kemudian, program akan mencoba membuat file dengan nama tersebut menggunakan method **createNewFile()**. Jika berhasil, maka akan ditampilkan **“File berhasil dibuat”** pada console, jika tidak maka akan ditampilkan **“File Gagal di Buat”** pada console.
- Dan jika percobaan tersebut gagal (menghasilkan eksepsi **Exception**) maka akan ditangkap dalam object **e**. Lalu, program akan menampilkan **“Error”** pada console
- Lalu, program akan mengecek apakah file tersebut ada atau tidak menggunakan method **exists()** yang kemudian hasilnya ditampilkan bersama dengan string **“Apakah file ada? ”**

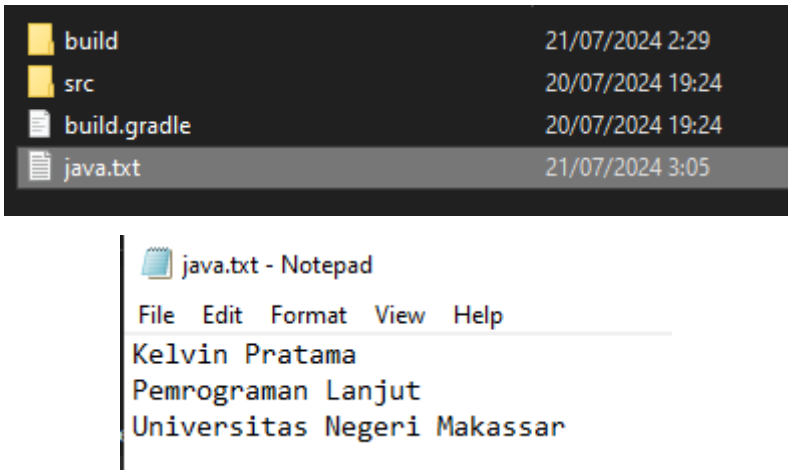
- Lalu, program akan mengecek apakah file tersebut bisa dibaca menggunakan method **canRead()** yang kemudian hasilnya ditampilkan bersama dengan string **“Apakah file bisa dibaca? ”**
- Lalu, program akan mengecek apakah file tersebut bisa ditulis menggunakan method **canWrite()** yang kemudian hasilnya ditampilkan bersama dengan string **“Apakah file bisa ditulis? ”**
- Lalu, program akan mengecek apakah file tersebut merupakan direktori menggunakan method **isDirectory()** yang kemudian hasilnya ditampilkan bersama dengan string **“Apakah berupa direktori? ”**
- Lalu, program akan mengecek apakah file tersebut merupakan sebuah file menggunakan method **isFile()** yang kemudian hasilnya ditampilkan bersama dengan string **“Apakah berupa file? ”**
- Lalu, program akan mengecek nama file tersebut menggunakan method **getName()** yang kemudian hasilnya ditampilkan bersama dengan string **“Apa namanya? ”**
- Lalu, program akan mengecek lokasi relatif file tersebut menggunakan method **getPath()** yang kemudian hasilnya ditampilkan bersama dengan string **“Dimana lokasinya? ”**
- Lalu, program akan mengecek lokasi absolut (lengkap) file tersebut menggunakan method **getAbsolutePath()** yang kemudian hasilnya ditampilkan bersama dengan string **“Dimana lokasi lengkapnya? ”**

F. Praktikum 6.6 – TulisFile.java

Source code:

```
1   package com.example.Pertemuan6;
2   //Praktikum 6.6
3   import java.io.File;
4   import java.io.FileNotFoundException;
5
6   public class P6_TulisFile {
7       public static void main(String args[]){
8           File file = new File("java.txt");
9           try{
10              java.io.PrintWriter output = new java.io.PrintWriter(file);
11              output.println("Kelvin Pratama");
12              output.println("Pemrograman Lanjut");
13              output.println("Universitas Negeri Makassar");
14              output.close();
15          } catch (FileNotFoundException e){
16              e.printStackTrace();
17          }
18      }
19  }
```

Output:



Penjelasan:

Pada praktikum ini dilakukan import **method** bawaan java seperti:

- **java.io.File** yang digunakan untuk mengakses sebuah file.
- **java.ioFileNotFoundException** yang digunakan untuk eksepsi jika file tidak ditemukan dari class **java.io.File**

Kemudian dibuat sebuah class dengan nama **P6_TulisFile** dengan sebuah method **main**.

Pada **main** method, dilakukan eksekusi sebagai berikut:

- Dibuat sebuah object **file** untuk mengakses file **java.txt** menggunakan class **java.io.File**
- Kemudian, program mencoba untuk membuat sebuah interface writer **output** pada object **file** menggunakan class **java.io.PrintWriter**.
- Lalu, program mencoba menulis **“Kelvin Pratama”** pada object **output**
- Lalu, program mencoba menulis **“Pemrograman Lanjut”** pada object **output**
- Lalu, program mencoba menulis **“Universitas Negeri Makassar”** pada object **output**
- Lalu, program menutup file tersebut untuk dibaca menggunakan method **close()** pada object **output**.
- Jika program terjadi eksepsi **FileNotFoundException (file tidak ditemukan)** akan ditangkap pada object **e**. Lalu ditampilkan call stack terakhirnya hingga terjadi eksepsi menggunakan method **printStackTrace()**

G. Praktikum 6.7 – BacaFile.java

Source code:

```java
package com.example.Pertemuan6;
//Praktikum 6.7
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

public class P7_BacaFile {
    public static void main(String args[]){
        File file = new File(("java.txt"));
        try {
            Scanner input = new Scanner(file);
            input.useDelimiter("\n");
            while(input.hasNext()){
                String nama = input.next();
                String mk = input.next();
                String pt = input.next();
                System.out.println("Nama: " + nama);
                System.out.println("Mata Kuliah: " + mk);
                System.out.println("Perguruan Tinggi: " + pt);
            }
            input.close();
        } catch (FileNotFoundException e){
            e.printStackTrace();
        }
    }
}
```

Output:

```
> Task :app:run
Nama: Kelvin Pratama
Mata Kuliah: Pemrograman Lanjut
Perguruan Tinggi: Universitas Negeri Makassar

BUILD SUCCESSFUL in 604ms
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada praktikum ini dilakukan import **method** bawaan java seperti:

- **java.io.File** yang digunakan untuk mengakses sebuah file.
- **java.ioFileNotFoundException** yang digunakan untuk eksepsi jika file tidak ditemukan dari class **java.io.File**
- **java.util.Scanner** yang digunakan untuk membaca input dari sebuah interaksi eksternal, contohnya file atau user input.

Kemudian dibuat sebuah class dengan nama **P7_TulisFile** dengan sebuah method **main**.

Pada **main** method, dilakukan eksekusi sebagai berikut:

- Dibuat sebuah object **file** untuk mengakses file **java.txt** menggunakan class **java.io.File**

- Kemudian, program akan mencoba membuat sebuah object **input** untuk membaca file dari object **file** menggunakan class **java.util.Scanner**.
- Lalu, didefinisikan pemisah antar data dengan pemisah **"\n"** atau baris baru.
- Lalu, perulangan dengan kondisi **selama object input masih memiliki data belum dibaca**.
- Kemudian, program akan membaca object **input** untuk data berikutnya (**next()**) yang kemudian disimpan dalam variabel **nama** dengan tipe data **String**.
- Lalu, program akan membaca object **input** untuk data berikutnya (**next()**) yang kemudian disimpan dalam variabel **mk** dengan tipe data **String**.
- Lalu, program akan membaca object **input** untuk data berikutnya (**next()**) yang kemudian disimpan dalam variabel **pt** dengan tipe data **String**.
- Lalu, program akan menampilkan string **"Nama: "** dengan isi variabel **nama**
- Lalu, program akan menampilkan string **"Mata Kuliah: "** dengan isi variabel **mk**
- Lalu, program akan menampilkan string **"Perguruan Tinggi: "** dengan isi variabel **pt**
- Lalu, program akan mengakhir pembacaan file dengan method **close** pada object **input**.
- Jika program terjadi eksepsi **FileNotFoundException (file tidak ditemukan)** akan ditangkap pada object **e**. Lalu ditampilkan call stack terakhirnya hingga terjadi eksepsi menggunakan method **printStackTrace()**

H. Praktikum 6.8 – ContohEksepsi1.java

Source code:

```
1  package com.example.Pertemuan6;
2  //Praktikum 6.8
3  public class P8_ContohEksepsi1 {
4      public static void main(String args[]){
5          int[] arrayInteger = new int[5];
6          arrayInteger[7] = 9; //SALAH, karena tidak terdapat index ke-7
7      }
8  }
```

Output:

```
> Task :app:run FAILED
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 7 out of bounds for length 5
        at com.example.Pertemuan6.P8_ContohEksepsi1.main(P8_ContohEksepsi1.java:6)

FAILURE: Build failed with an exception.
```

Penjelasan:

Pada praktikum ini, dibuat sebuah class dengan nama **P8_ContohEksepsi1** dengan sebuah method **main**.

Pada **main** method, dilakukan eksekusi sebagai berikut:

- Dibuat sebuah array tipe data **integer** dengan nama **arrayInteger** dengan member **{1,2,3,4,5}**
- Kemudian, program akan mencoba mengakses array dengan indeks **7** (data ke-8, zero-based index) untuk diisi dengan nilai **9**. Eksekusi ini akan gagal dan akan membangkitkan eksepsi **ArrayIndexOutOfBoundsException** seperti yang terlihat pada output console.

I. Praktikum 6.9 – ContohEksepsi2.java

Source code:

```
1   package com.example.Pertemuan6;
2   //Praktikum 6.9
3   public class P9_ContohEksepsi2 {
4       public static void main(String[] args){
5           int pembilang = 7;
6           int penyebut = 0;
7           try {
8               int hasil = pembilang / penyebut;
9               System.out.println("Hasil = " + hasil);
10          } catch (Exception e){
11              System.out.println("KESALAHAN: " + "Terdapat pembagian dengan nol");
12          }
13          System.out.println("Statement setelah blok trycatch");
14      }
15  }
```

Output:

```
> Task :app:run
KESALAHAN: Terdapat pembagian dengan nol
Statement setelah blok trycatch

BUILD SUCCESSFUL in 717ms
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada praktikum ini, dibuat sebuah class dengan nama **P9_ContohEksepsi2** dengan sebuah method **main**.

Pada **main** method, dilakukan eksekusi sebagai berikut:

- Dideklarasikan variabel **pembilang** dengan tipe data **integer** dan diinisialisasi dengan nilai **7**.
- Dideklarasikan variabel **penyebut** dengan tipe data **integer** dan diinisialisasi dengan nilai **0**.

- Kemudian, program mencoba untuk menghitung nilai variabel **pembilang** dibagi dengan variabel **penyebut** lalu hasilnya disimpan dalam variabel **hasil** dengan tipe data integer
- Lalu, program akan menampilkan string **"Hasil = "** dengan nilai variabel **hasil**
- Jika program membangkitkan eksepsi **Exception** maka akan ditangkap pada object **e**, lalu akan ditampilkan string **"KESALAHAN: Terdapat pembagian dengan nol"** pada console
- Lalu setelah itu program akan menampilkan string **"Statement setelah blok try-catch"**

J. Praktikum 6.10 – ContohMultiEksepsi.java

Source code:

```
1   package com.example.Pertemuan6;
2   //Praktikum 6.10
3   public class P10_ContohMultiEksepsi {
4       public static void cobaEksepsi(int pembilang, int penyebut){
5           try{
6               int hasil = pembilang / penyebut;
7               System.out.println("Hasil bagi: " + hasil);
8               int[] Arr = {1,2,3,4,5};
9               Arr[10] = 23;
10          } catch(ArithmeticException eksepsi){
11              System.out.println("Terdapat pembagian dengan 0");
12          } catch (ArrayIndexOutOfBoundsException eksepsi2){
13              System.out.println("Indeks di luar rentang");
14          }
15      }
16      public static void main(String args[]){
17          cobaEksepsi(4,0);
18          System.out.println();
19          cobaEksepsi(12,4);
20      }
21  }
```

Output:

```
> Task :app:run
Terdapat pembagian dengan 0

Hasil bagi: 3
Indeks di luar rentang

BUILD SUCCESSFUL in 2s
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada praktikum ini, dibuat sebuah class dengan nama **P9_ContohEksepsi2** dengan sebuah method **main** dan method **cobaEksepsi**.

Pada method **cobaEksepsi** diperlukan 2 parameter untuk pemanggilannya, yaitu variabel **pembilan** dan **penyebut** dengan tipe data **integer**. Dalam method ini dilakukan eksekusi sebagai berikut:

- Program mencoba melakukan eksekusi menggunakan **try-catch**
- Dalam blok **try**, program menghitung nilai dari operasi matematika nilai variabel **pembilang** dibagi dengan **penyebut** lalu hasilnya disimpan dalam variabel **hasil** dengan tipe data **integer**. Jika terdapat operasi yang tidak valid, seperti pembagian dengan nilai **0**, program akan membangkitkan eksepsi **ArithmeticException**.
- Lalu, program akan menampilkan string **"Hasil bagi: "** dengan nilai variabel **hasil**.
- Kemudian, program membuat array dari tipe data **integer** dengan nama **Arr** dan member **{1,2,3,4,5}**
- Kemudian, program mencoba mengakses indeks ke-**10** (**data ke-11, zero-based index) array Arr** untuk diisi dengan nilai **23**. Eksekusi ini akan membangkitkan eksepsi **ArrayIndexOutOfBoundsException**, karena tidak ada index ke-10 pada array tersebut.
- Jika program membangkitkan eksepsi **ArithmeticException** maka akan ditangkap pada object **e**. Lalu, akan ditampilkan string **"Terdapat pembagian dengan 0"** pada console
- Jika program membangkitkan eksepsi **ArrayIndexOutOfBoundsException** maka akan ditangkap pada object **e**. Lalu, akan ditampilkan string **"Indeks diluar rentang"** pada console

Pada **main** method, dilakukan eksekusi sebagai berikut:

- Pemanggilan method **cobaEksepsi** dengan argumen nilai **4** dan **0**.
- Pemanggilan method **cobaEksepsi** dengan argumen nilai **12** dan **4**.

K. Praktikum 6.11 – Mahasiswa.java

Source code:

```java
package com.example.Pertemuan6;
//Praktikum 6.11
class DemoThrow {
    private String nim;
    private String nama;

    public void setNIM(String inputNIM){
        try{
            if(inputNIM == null){
                throw new NullPointerException();
            }
            nim = inputNIM;
        }catch (NullPointerException npe){
            System.out.println("KESALAHAN: NIM tidak boleh null");
        }
    }
    public String getNIM(){
        return nim;
    }
    public void setNama(String inputNama){
        try{
            if(inputNama == null){
                throw new NullPointerException();
            }
            nama = inputNama;
        }catch(NullPointerException npe){
            System.out.println("KESALAHAN: Nama mahasiswa tidak boleh null");
        }
    }
    public String getNama(){
        return nama;
    }
}
public class P11_Mahasiswa{
    public static void main(String[] args){
        DemoThrow mhs = new DemoThrow();
        mhs.setNIM(null);
        mhs.setNama("Kelvin");
        System.out.println("\nNIM : " + mhs.getNIM());
        System.out.println("Nama: "+ mhs.getNama());
    }
}
```

Output:

```
> Task :app:run
KESALAHAN: NIM tidak boleh null

NIM : null
Nama: Kelvin

BUILD SUCCESSFUL in 2s
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada praktikum ini, dideklarasikan 2 class yaitu **DemoThrow** dan **Mahasiswa**.

Class **DemoThrow** dideklarasikan dengan modifier **default**.

Pada class **DemoThrow**, dideklarasikan 2 field dengan modifier **private** dengan tipe data **String**, yaitu **nim** dan **nama**.

Kemudian, dideklarasikan method setter untuk field **nim** dengan nama **setNIM** dengan parameter pemanggilan variabel **inputNIM**. Eksekusi didalam method ini adalah sebagai berikut:

- Program mencoba untuk memeriksa jika **inputNIM** adalah **null**, jika iya maka program akan membangkitkan eksepsi **NullPointerException**.
- Program memasukkan nilai **inputNIM** dalam field **nim**.
- Jika program membangkitkan eksepsi **NullPointerException** maka akan ditangkap kedalam object **e**. Lalu akan ditampilkan **"KESALAHAN: NIM tidak boleh null"** pada console.

Lalu, dideklarasikan method getter untuk field **nim** dengan nama **getNIM**. Method ini akan mengembalikan nilai field **nim** pada pemanggil method.

Kemudian, dideklarasikan method setter untuk field **nama** dengan nama **setNama** dengan parameter pemanggilan variabel **inputNama**. Eksekusi didalam method ini adalah sebagai berikut:

- Program mencoba untuk memeriksa jika **inputNama** adalah **null**, jika iya maka program akan membangkitkan eksepsi **NullPointerException**.
- Program memasukkan nilai **inputNama** dalam field **nama**.
- Jika program membangkitkan eksepsi **NullPointerException** maka akan ditangkap kedalam object **e**. Lalu akan ditampilkan **"KESALAHAN: Nama Mahasiswa tidak boleh null"** pada console.
- Lalu, dideklarasikan method getter untuk field **nama** dengan nama **getNama**. Method ini akan mengembalikan nilai field **nama** pada pemanggil method.

Kemudian, dibuat sebuah class dengan nama **P11_Mahasiswa** dengan sebuah method **main**.

Pada **main** method, dilakukan eksekusi sebagai berikut:

- Dibuat sebuah object **mhs** dari class **DemoThrow**
- Dipanggil method **setNIM** dengan argumen **null** pada object **mhs**
- Dipanggil method **setNama** dengan argumen **"Kelvin"** pada object **mhs**
- Kemudian, program menampilkan string **"NIM: "** dengan nilai balik pemanggilan method **getNIM()** pada object **mhs**

- Kemudian, program menampilkan string **"Nama: "** dengan nilai balik pemanggilan method **getNama()** pada object **mhs**

L. Prakikum 6.12 – DemoThrows.java

Source code:

```
1   package com.example.Pertemuan6;
2   //Praktikum 6.12
3   public class P12_DemoThrows {
4       public static void cobaEksepsi() throws IllegalAccessException{
5           throw new IllegalAccessException("KESALAHAN: Illegal Access");
6       }
7       public static void main(String[] args){
8           try{
9               cobaEksepsi();
10          }catch(Exception e){
11              System.out.println("Eksepsi ditangkap disini....");
12              System.out.println(e.getMessage());
13          }
14          System.out.println("Statement setelah blok try-catch");
15      }
16  }
```

Output:

```
> Task :app:run
Eksepsi ditangkap disini....
KESALAHAN: Illegal Access
Statement setelah blok try-catch

BUILD SUCCESSFUL in 1s
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada praktikum ini, dibuat sebuah class dengan nama **P12_DemoThrows** dengan sebuah method **main** dan method **cobaEksepsi**.

Pada method **cobaEksepsi**, akan dibangkitkan eksepsi **IllegalAccessException** dengan pesan **"KESALAHAN: Illegal Access"**

Pada **main** method, dilakukan eksekusi sebagai berikut:

- Program mencoba melakukan eksekusi method **cobaEksepsi()**
- Jika program membangkitkan **Exception** maka akan ditangkap pada object **e**. Lalu program akan menampilkan string **"Eksepsi ditangkap disini…"** dan pesan dari eksepsi tersebut.
- Lalu program akan menampilkan string **"Statement setelah blok try-catch"** setelah blok **try-catch**

M. Praktikum 6.13 – DemoFinally.java

Source code:

```java
package com.example.Pertemuan6;
//Praktikum 6.13
public class P13_DemoFinally {
    public static void cobaEksepsi(int pembilang, int penyebut){
        try{
            int hasil = pembilang / penyebut;
            System.out.println("Hasil bagi: " + hasil);
            int[] Arr = {1,2,3,4,5};
            Arr[10] = 23;
        }catch(ArithmeticException eksepsil){
            System.out.println("Terdapat pembagian dengan 0");
        }catch(ArrayIndexOutOfBoundsException eksepsi2){
            System.out.println("Indeks di luar rentang");
        }finally {
            System.out.println("Ini adalah statement dalam blok finally");
        }
    }
    public static void main(String[] args){
        cobaEksepsi(4,0);
        System.out.println();
        cobaEksepsi(12,3);
    }
}
```

Output:

```
> Task :app:run
Terdapat pembagian dengan 0
Ini adalah statement dalam blok finally

Hasil bagi: 4
Indeks di luar rentang
Ini adalah statement dalam blok finally

BUILD SUCCESSFUL in 2s
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada praktikum ini, dibuat sebuah class dengan nama **P13_DemoFinally** dengan sebuah method **main** dan method **cobaEksepsi**.

Pada method **cobaEksepsi** diperlukan 2 parameter untuk pemanggilannya, yaitu variabel **pembilang** dan **penyebut** dengan tipe data **integer**. Dalam method ini dilakukan eksekusi sebagai berikut:

- Program mencoba melakukan eksekusi menggunakan **try-catch**
- Dalam blok **try**, program menghitung nilai dari operasi matematika nilai variabel **pembilang** dibagi dengan **penyebut** lalu hasilnya disimpan dalam variabel **hasil** dengan tipe data **integer**. Jika terdapat operasi yang tidak valid, seperti pembagian dengan nilai **0**, program akan membangkitkan eksepsi **ArithmeticException**.
- Lalu, program akan menampilkan string **“Hasil bagi: ”** dengan nilai variabel **hasil**.

- Kemudian, program membuat array dari tipe data **integer** dengan nama **Arr** dan member **{1,2,3,4,5}**
- Kemudian, program mencoba mengakses indeks ke-**10** (**data ke-11, zero-based index) array Arr** untuk diisi dengan nilai **23**. Eksekusi ini akan membangkitkan eksepsi **ArrayIndexOutOfBoundsException**, karena tidak ada index ke-10 pada array tersebut.
- Jika program membangkitkan eksepsi **ArithmeticException** maka akan ditangkap pada object **e**. Lalu, akan ditampilkan string **"Terdapat pembagian dengan 0"** pada console
- Jika program membangkitkan eksepsi **ArrayIndexOutOfBoundsException** maka akan ditangkap pada object **e**. Lalu, akan ditampilkan string **"Indeks diluar rentang"** pada console
- Lalu, program akan menjalankan perintah pada blok **finally** terlepas dari adanya eksepsi yang dibangkitkan atau tidak. Program akan menampilkan string **"Ini adalah statement dalam blok finally"**

Pada **main** method, dilakukan eksekusi sebagai berikut:

- Pemanggilan method **cobaEksepsi** dengan argumen nilai **4** dan **0**.
- Pemanggilan method **cobaEksepsi** dengan argumen nilai **12** dan **4**.

N. Tantangan

Buatlah sebuah program penanganan eksepsi untuk inputan data yang tidak sesuai dengan format penulisan seperti berikut:

1. Nomor handphone : harus berupa nomor
2. Nomor KTP : harus berjumlah 16 digit angka
3. Alamat email : harus sesuai dengan format penulisan email yang baku

Source code:

```
package com.example.Pertemuan6;

import java.io.*;
import java.util.regex.*;

class Biodata{
    private String nomor_hp, nomor_ktp, email;
    public void setNomorHP(String nomor_hp){
        try{
            long nomor = Long.parseLong(nomor_hp);
            this.nomor_hp = nomor_hp;
        } catch(NumberFormatException e){
            System.out.println("KESALAHAN: Bukan format nomor! (String: " + nomor_hp + ")");
        }
    }
    public void setNomorKTP(String nomor_ktp){
        try{
            if(nomor_ktp.length() != 16){
                throw new Exception("Panjang nomor KTP harus 16 digit! (String: " + nomor_ktp + ")");
            }
            this.nomor_ktp = nomor_ktp;
        }catch(Exception e){
            System.out.println("KESALAHAN: " + e.getMessage());
        }
    }
    public void setEmail(String email){
        try{
            if(!adalahEmailValid(email)){
                throw new Exception("Bukan format email! (String: " + email + ")");
            }
            this.email = email;
        } catch(Exception e){
            System.out.println("KESALAHAN: " + e.getMessage());
        }
    }
    public String getNomorHP(){
        return this.nomor_hp;
    }
    public String getNomorKTP(){
        return this.nomor_ktp;
    }
    public String getEmail(){
        return this.email;
    }
    public static boolean adalahEmailValid(String email){
        String emailregex = "(?:[a-z0-9!#$%&'*+/=?^_`{|}~-]+(?:\\.[a-z0-9!#$%&'*+/=?^_`
        Pattern pat = Pattern.compile(emailregex);
        return pat.matcher(email).matches();
    }
}
```

```java
public class Tantangan {
    public static void main(String[] args){
        Biodata kelvin = new Biodata();
        BufferedReader input = new BufferedReader(new InputStreamReader(System.in));
        try{
            System.out.print("Masukkan nomor HP: ");
            String nomor_hp = input.readLine();

            System.out.print("Masukkan nomor KTP: ");
            String nomor_ktp = input.readLine();

            System.out.print("Masukkan email: ");
            String email = input.readLine();

            kelvin.setNomorHP(nomor_hp);
            kelvin.setNomorKTP(nomor_ktp);
            kelvin.setEmail(email);
        }catch (IOException e){
            System.out.println("KESALAHAN: "+ e.getMessage());
        }

        System.out.println("\nNomor HP: " + kelvin.getNomorHP());
        System.out.println("Nomor KTP: " + kelvin.getNomorKTP());
        System.out.println("E-mail: " + kelvin.getEmail());
    }
}
```

Output:

```
> Task :app:run
Masukkan nomor HP: 085155263857
Masukkan nomor KTP: 1234567890123456
Masukkan email: kelvin@kelvin-pratama.my.id

Nomor HP: 085155263857
Nomor KTP: 1234567890123456
E-mail: kelvin@kelvin-pratama.my.id

BUILD SUCCESSFUL in 19s
2 actionable tasks: 1 executed, 1 up-to-date
```

```
> Task :app:run
Masukkan nomor HP: bukannomor
Masukkan nomor KTP: 1234567890
Masukkan email: bukanemail.com
KESALAHAN: Bukan format nomor! (String: bukannomor)
KESALAHAN: Panjang nomor KTP harus 16 digit! (String: 1234567890)
KESALAHAN: Bukan format email! (String: bukanemail.com)

Nomor HP: null
Nomor KTP: null
E-mail: null

BUILD SUCCESSFUL in 13s
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada praktikum ini dilakukan import **method** bawaan java seperti:

- **java.io.*** yang digunakan untuk mengimport seluruh class dalam **java.io**.
- **java.util.regex.*** yang digunakan untuk mengimport seluruh class dalam **java.util.regex**

Kemudian dibuat sebuah class dengan modifier **default** dengan nama **Biodata**.

Pada class **Biodata**, dilakukan eksekusi sebagai berikut:

- Dideklarasikan **field** dengan modifier **private** dan tipe data **String** dengan nama **nomor_hp, nomor_ktp,** dan **email**.

- Kemudian dideklarasikan method setter untuk field **nomor_hp** dengan nama **setNomorHP** dan parameter pemanggilan variabel **nomor_hp** bertipe data **String**.
- Dalam method ini, program akan mencoba mengkonversi nilai **nomor_hp** ke tipe data **Long** dan nilainya disimpan pada dummy variabel **nomor**.
- Lalu, program akan menyimpan nilai argumen **nomor_hp** pada **field nomor_hp**.
- Jika program membangkitkan eksepsi **NumberFormatException**, akan ditampilkan **"KESALAHAN: Bukan format nomor! (String: "** diikuti nilai argumen **nomor_hp**.
- Kemudian dideklarasikan method setter untuk field **nomor_ktp** dengan nama **setNomorKTP** dan parameter pemanggilan variabel **nomor_ktp** bertipe data **String**.
- Dalam method ini, program akan memeriksa jika panjang argumen **nomor_ktp** tidak sama dengan **16**. Jika iya, maka program akan membangkitkan eksepsi **Exception** dengan pesan **"Panjang nomor KTP harus 16 digit! (String: "** diikuti dengan nilai argumen **nomor_ktp**
- Lalu, program akan menyimpan nilai argumen **nomor_ktp** pada **field nomor_ktp**.
- Jika program membangkitkan eksepsi **Exception**, akan ditampilkan **"KESALAHAN: "** diikuti pesan eksepsi.
- Kemudian dideklarasikan method setter untuk field **email** dengan nama **setEmail** dan parameter pemanggilan variabel **email** bertipe data **String**.
- Dalam method ini, program akan memeriksa nilai balik pemanggilan method **adalahEmailValid** dengan argumen nilai variabel **email** adalah **false**. Jika iya, maka program akan membangkitkan eksepsi **Exception** dengan pesan **"Bukan format email (String: "** diikuti dengan nilai argumen **nomor_ktp**
- Lalu, program akan menyimpan nilai argumen **email** pada **field email**.
- Jika program membangkitkan eksepsi **Exception**, akan ditampilkan **"KESALAHAN: "** diikuti pesan eksepsi.
- Kemudian, dideklarasikan method getter untuk field **nomor_hp** dengan nama **getNomorHP()**
- Kemudian, dideklarasikan method getter untuk field **nomor_ktp** dengan nama **getNomorKTP()**
- Kemudian, dideklarasikan method getter untuk field **email** dengan nama **getEmail()**

- Kemudian, dideklarasikan method **adalahEmailValid** dengan tipe data balik **boolean** dan parameter pemanggilan **email** dengan tipe data **String**
- Dalam method ini, dilakukan eksekusi pengecekan jika string yang diberikan dalam pemanggilan sesuai dengan format **regular expression** email yang dideklarasikan pada string **emailregex**. Jika iya, program akan mengembalikan nilai **true**. Jika tidak, maka program akan mengembalikan nilai **false**.

Kemudian dibuat sebuah class dengan nama **Tantangan** dengan sebuah method **main**. Pada **main** method, dilakukan eksekusi sebagai berikut:

- Dibuat sebuah object dengan nama **kelvin** dari class **Biodata**.
- Dibuat sebuah object interface **input** untuk membaca user-input dari console.
- Kemudian, program akan mencoba menjalankan blok **try-catch**
- Dalam blok **try**, program menampilkan string **"Masukkan nomor HP: "**
- Lalu, program akan mengambil nilai input dari console lalu memasukkannya pada variabel **nomor_hp** dengan tipe data **String**.
- Lalu, program menampilkan string **"Masukkan nomor KTP: "**
- Lalu, program akan mengambil nilai input dari console lalu memasukkannya pada variabel **nomor_ktp** dengan tipe data **String**.
- Lalu, program menampilkan string **"Masukkan email: "**
- Lalu, program akan mengambil nilai input dari console lalu memasukkannya pada variabel **email** dengan tipe data **String**.
- Lalu, dipanggil method **setNomorHP** dengan argumen variabel **nomor_hp** pada object **kelvin**.
- Lalu, dipanggil method **setNomorKTP** dengan argumen variabel **nomor_ktp** pada object **kelvin**.
- Lalu, dipanggil method **setEmail** dengan argumen variabel **email** pada object **kelvin**.
- Jika program membangkitkan eksepsi **IOException** pada blok **try**, maka blok **catch** akan dieksekusi. Program akan menampilkan string **"KESALAHAN: "** dengan pesan eksepsi yang diberikan.
- Lalu, program akan menampilkan **"Nomor HP: "** diikuti dengan nilai balik pemanggilan method **getNomorHP()** pada object **kelvin**
- Lalu, program akan menampilkan **"Nomor KTP: "** diikuti dengan nilai balik pemanggilan method **getNomorKTP()** pada object **kelvin**

- Lalu, program akan menampilkan **“E-mail: ”** diikuti dengan nilai balik pemanggilan method **getEmail()** pada object **kelvin**