

**LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK**

**MODUL 7
“EXEPTION HANDLING”**



Disusun oleh
Bunga Laelatul Muna
NIM : 21102010

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2023**

BAB 1 – TUJUAN PRAKTIKUM

Tujuan dari praktikum instalasi Java (IntelliJ IDEA) adalah untuk memberikan pengetahuan dan keterampilan kepada mahasiswa tentang Exception Handling

- Mahasiswa dapat memahami tentang konsep utama pbo yaitu Exception Handlin
- Mahasiswa dapat mengerti berbagai jenis Exception Handling

BAB II- DASAR TEORI

Apa itu Exception ? Mudahnya, Exception adalah cara program memberitahu Anda apa yang salah dengan kode Anda hingga menyebabkan program error. Dalam perjalanan Anda mempelajari bahasa Java, Anda pasti sering menghadapi yang namanya error, entah yang disebabkan karena kesalahan penulisan syntax, lupa melakukan import, atau kekeliruan dalam logika program yang anda tulis. Beberapa contoh Exception yang mungkin ditampilkan oleh compiler adalah `ArrayIndexOutOfBoundsException`, `InputMismatchException`, dan lain sebagainya.

Dapatkah exception dicegah ? Tentu saja bisa, dengan yaitu dengan menulis kode dengan benar. Akan tetapi Walau begitu, ada sebuah cara untuk mengabaikan sebuah exception agar program tetap berjalan walau terjadi error, yaitu dengan menggunakan yang namanya Try-Catch! Adakah alasan baik mengabaikan sebuah exception? Tentu ada! Bayangkan Anda membuat sebuah program besar, tetapi terjadi kekeliruan yang sangat sepele dan sebenarnya tidak begitu fatal error-nya. Normalnya, program Anda akan keluar jika ini terjadi. Namun jika Anda menggunakan Try-Catch, program akan tetap berjalan walau ada sebuah error. Tidak hanya itu, Anda juga memiliki kontrol penuh apa yang harus dilakukan program jika sebuah error terjadi.

BAB III – GUIDED

Guided 1 ArrayException Handling

- Source Code

```
package com.Bunga_Laelatul_Muna.PB0.Pertemuan7.Guided;  
//Bunga Laelatul Muna  
//21102010  
  
public class ArrayException {  
    public static void main(String[] args) {  
        try {  
            int[] numbers = {1, 2, 3};  
            System.out.println(numbers[10]);  
            // Program yang benar  
        } catch (ArrayIndexOutOfBoundsException e) {  
            System.out.println("Terjadi kesalahan: " + e.getMessage());  
            // Program yang salah  
        }  
    }  
}
```

- SS Output

```
Terjadi kesalahan: Index 10 out of bounds for length 3  
  
Process finished with exit code 0
```

- Penjelasan

- ArrayIndexOutOfBoundsException merupakan salah satu jenis eksepsi yang terjadi ketika mencoba mengakses indeks yang tidak valid pada sebuah array
- Ketika menggunakan ini dan disertai catch maka program yang salah akan tetap bisa di run namun menampilkan pesan “Terjadi Kesalahan”.

Guided 2 MultiException

- **Source Code**

```
package com.Bunga_Laelatul_Muna.PB0.Pertemuan7.Guided;
import java.util.InputMismatchException;
import java.util.Scanner;

public class multiException {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        try{
            System.out.println("Masukan Bilangan: ");
            int bilangan = input.nextInt();

            System.out.println("Masukan pembagi: ");
            int pembagi = input.nextInt();
            int hasil = bilangan/pembagi;

            System.out.println(bilangan + " / " + pembagi + " = " + hasil + " (dibulatkan)");
        }
        catch (ArithmeticException | InputMismatchException e){
            System.out.println("Error: Tidak dapat memproses");
        }
        System.out.println("Proses selesai");
    }
}
```

- **SS Output**

```
"C:\Program Files\Java\jdk-20\bin\java.exe" "-ja
Masukan Bilangan:
4
Masukan pembagi:
0
Error: Tidak dapat memproses
Proses selesai

Process finished with exit code 0
```

- **Penjelasan**

- Pada awal program, kita mengimpor dua kelas, yaitu `InputMismatchException` dan `Scanner`, yang diperlukan untuk mendapatkan input dari pengguna dan menangani kesalahan input.
- Selanjutnya, kita mendeklarasikan sebuah objek `Scanner` dengan nama `input` untuk membaca input dari pengguna.
- Di dalam blok `try`, kita meminta pengguna untuk memasukkan dua bilangan: `bilangan` dan `pembagi`.

- Kemudian, kita melakukan operasi pembagian dengan membagi bilangan dengan pembagi dan menyimpan hasilnya di variabel hasil.
- Setelah itu, kita mencetak hasil pembagian dengan pesan yang sesuai.
- Jika terjadi kesalahan dalam blok try, baik dalam operasi pembagian maupun saat membaca input, eksepsi yang cocok akan ditangkap oleh blok catch.
- Pada blok catch, kita menggunakan multiple catch dengan menggunakan "|" (pipe) untuk menangani lebih dari satu jenis eksepsi. Dalam contoh ini, kita menangkap ArithmeticException dan InputMismatchException.
- Jika terjadi salah satu dari eksepsi yang ditangkap, kita mencetak pesan kesalahan yang sesuai.
- Setelah blok try-catch, kita mencetak pesan "Proses selesai" yang akan ditampilkan setelah eksekusi program

BAB IV – UNGUIDED

UnGuided FileNotFoundException

- Source Code

```
package com.Bunga_Laelatul_Muna.PB0.Pertemuan7.Tugas;
//Bunga Laelatul Muna
//21102010
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

public class FileNotFoundExep21102010 {
    public static void main(String[] args) {
        try {
            File file = new File( pathname: "Ada.txt"); // Mencoba membuka file yang tidak ada
            Scanner scanner = new Scanner(file);

            System.out.println("File ditemukan. Program berjalan dengan benar.");
        } catch (FileNotFoundException e) {
            System.out.println("Terjadi FileNotFoundException: " + e.getMessage());
            System.out.println("File tidak ditemukan. Program mengalami kesalahan.");
        }
    }
}
```

- **SS Output**

```
"C:\Program Files\Java\jdk-20\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA\bin\idea_rt.jar" -Dfile.encoding=UTF-8
Terjadi FileNotFoundException: Ada.txt (The system cannot find the file specified)
File tidak ditemukan. Program mengalami kesalahan.

Process finished with exit code 0
```

- **Penjelasan**

- FileNotFoundException adalah jenis exception yang dilempar saat sebuah program mencoba membuka atau mengakses sebuah file yang tidak ditemukan di sistem file.
- Dalam contoh di atas, kita mencoba membuka file dengan nama "Ada.txt" menggunakan objek File dan Scanner. Jika file tidak ditemukan, FileNotFoundException akan dilempar dan ditangkap oleh blok catch. Dalam blok catch, kita mencetak pesan "File tidak ditemukan" kepada pengguna.

UnGuided Null Pointer Exception

- **Source Code**

```
package com.Bunga_Laelatul_Muna.PB0.Pertemuan7.Tugas;
//Bunga Laelatul Muna
//21102010

public class NullPointerExep21102010 {
    public static void main(String[] args) {
        String str = null;
        try {
            int length = str.length(); // Akses method pada objek null
            System.out.println("Program berjalan dengan benar.");
        } catch (NullPointerException e) {
            System.out.println("Terjadi NullPointerException: " + e.getMessage());
            System.out.println("Program mengalami kesalahan.");
        }
    }
}
```

- **SS Output**

```
"C:\Program Files\Java\jdk-20\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA\bin\idea_rt.jar" -Dfile.encoding=UTF-8
Terjadi NullPointerException: Cannot invoke "String.length()" because "str" is null
Program mengalami kesalahan.

Process finished with exit code 0
```

- **Penjelasan**

- Null Pointer Exception (NullPointerException) adalah jenis exception yang dilempar saat sebuah program mencoba menggunakan atau mengakses objek yang memiliki nilai null. Dalam bahasa Java, null adalah nilai khusus yang menandakan bahwa sebuah objek tidak memiliki referensi yang valid atau tidak menunjuk ke objek apapun.
- Di program tsb kita mendeklarasikan variable Str bernilai Null, kemudian kita mencoba mengakses method pada objek Null nya. Kemudian kita catch atau menangkap sebuah pesan, dimana Ketika nilai Null valid / terdapat kesalahan maka program akan tetap berjalan namun akan menampilkan pesan di catch yakni “Terjadi Kesalahan”.

UnGuided Illegal Argument Exception

- **Source Code**

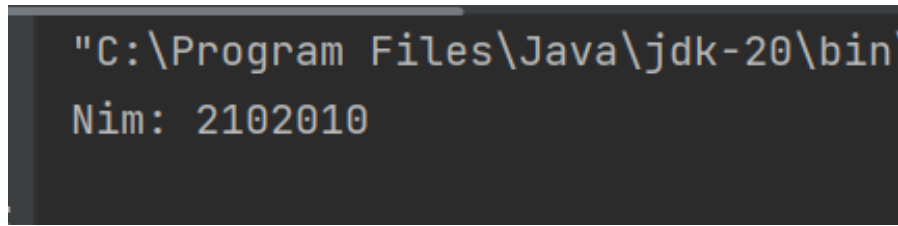
```
package com.Bunga_Laelatul_Muna.PB0.Pertemuan7.Tugas;
public class IllegalArg21102010 {
    public static void main(String[] args) {
        try {
            int nim = -2102010;
            validateNim(nim);
            // Program yang benar
        } catch (IllegalArgumentException e) {
            System.out.println("Terjadi kesalahan: " + e.getMessage());
            // Program yang salah
        }
    }

    public static void validateNim(int nim) {
        if (nim < 0) {
            throw new IllegalArgumentException("Nim tidak valid: " + nim);
        }
        System.out.println("Nim: " + nim);
    }
}
```

- **SS Output** (Ketika nilai nim = -21102010) atau **SALAH**

```
"C:\Program Files\Java\jdk-20\bin\java.exe" "-javaagent:C:\P
Terjadi kesalahan: Nim tidak valid: -2102010
```

Ketika nilai nim = 21102010 (**BENAR**)



```
"C:\Program Files\Java\jdk-20\bin\  
Nim: 2102010
```

- **Penjelasan**

- `IllegalArgumentException` adalah jenis exception yang dilempar saat sebuah program menerima argumen yang tidak valid atau tidak sesuai dengan yang diharapkan oleh suatu metode atau konstruktor. `IllegalArgumentException` termasuk dalam kategori checked exception, yang berarti kita harus menanganinya secara eksplisit dalam kode kita.
- Di program tersebut kita mencoba menginisialisasi variabel `nim` dengan nilai `-21102010`. Kemudian dibawahnya kita ada memanggil sebuah fungsi `ValidateNim` dengan parameter `nim`.
- Fungsi `ValidateNim` berisi pengkondisian bahwa Ketika nilai `nim` kurang dari 0 maka akan dikirim pesan bahwa "Nim tidak valid +nim". Dan di dalam blok `catch` pun akan dikirim pesan 'Terjadi Kesalahan'.
- Namun jika program benar di dalam fungsi `ValidateNim` sudah di deklarasikan output "Nim: + nim".

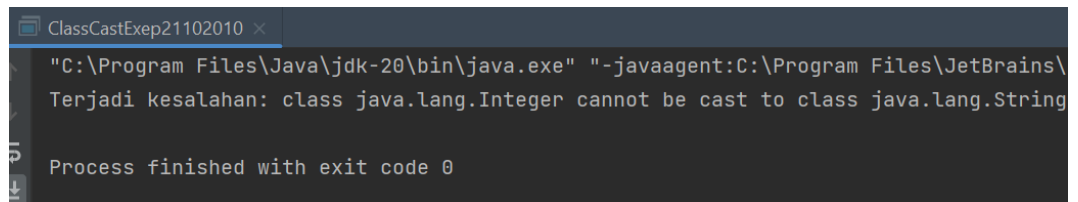
UnGuided Class Cast Exception

- **Source Code**



```
package com.Bunga_Laelatul_Muna.PB0.Pertemuan7.Tugas;  
  
public class ClassCastExep21102010 {  
    public static void main(String[] args) {  
        try {  
            Object nim = new Integer( value: 21102010);  
            String str = (String) nim; //Melakukan penugasan yg tidak valid, inisialisasi awal int tp di suruh str  
            System.out.println(str);  
            // Program yang benar  
        } catch (ClassCastException e) {  
            System.out.println("Terjadi kesalahan: " + e.getMessage());  
            // Program yang salah  
        }  
    }  
}
```


- **SS Output**



```
ClassCastExep21102010 x
"C:\Program Files\Java\jdk-20\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\
Terjadi kesalahan: class java.lang.Integer cannot be cast to class java.lang.String

Process finished with exit code 0
```

- **Penjelasan**

- ClassCastException adalah jenis exception yang dilempar saat mencoba melakukan penugasan (casting) objek ke tipe yang tidak sesuai atau tidak kompatibel. Ketika tipe objek tidak dapat dikonversi ke tipe yang diinginkan, maka ClassCastException akan dilempar.
- Program diatas di inisialisais Object nim bernilai 21102010 yang artinya type data int. Kemudian dibawahnya di perintahkan lagi variable str dengan tipe data String bernilai Object nim yang bertipe data int. Hal ini akan membuat program eror Ketika di run, karena melakukan penugasan yang tidak valid.
- Setelah di scan program tidak valid karena output yang akan di hasilkan ada print(str), program akan langsung mengarahkan ke blok catch
- Dimana blok catch akan tetap menjalankan program yang eror namun dengan memberikan tambahan pesan bahwa 'Terjadi Kesalahan'.

A. REFERENSI

1. Java Documentation
2. Modul 7