LAPORAN PRAKTIKUM PEMROGRAMAN BERBASIS OBJEK
MODUL 8 EXCEPTION, I/O, OPERASI FILE

Oleh :
Alif Alpian Sahrul Muharom
(20102007)

Dosen:
Agus Priyanto, M.Kom

PROGRAM STUDI S1 TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

PURWOKERTO

2022

I. TUJUAN

a. Mengerti prinsip polimorfisme dalam bahasa C++ dan Java
b. Mengerti tentang prinsip polimorfisme dan pemakaiannya dalam membentuk suatu kelas.

II. TOOL

1. Apache NetBeans IDE 13
2. Java SE Development Kit 18

III. DASAR TEORI

IV. GUIDED

1. CobaThrow.java

```
package main.java.com.Alpian.Pertemuan_9.Guided;
/**
 * @author
 * Alif Alpian Sahrul Muharom
 * 20102007
 * IF-08-0
 */
public class CobaThrow {
    public static void methodLain() {
        try {
            throw new ArrayIndexOutOfBoundsException(1);
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Penanganan exception "
                    + "dalam method methodLain()");
            throw e;
        }
    }

    public static void main(String[] args) {
        try {
            methodLain();
        } catch (Exception e) {
            System.out.println("Penanganan exception "
                    + "dalam method main()");
        }
    }
}
```

Penjelasan

Program di atas mencontohkan bagaimana penggunaan dasar dari try and catch method pada method-method pada suatu kelas dan fungsi main.

2. TryCatch1.java

```
package main.java.com.Alpian.Pertemuan_9.Guided;
/**
 * @author
 * Alif Alpian Sahrul Muharom
 * 20102007
 * IF-08-0
 */
public class TryCatch1 {
    public static void main(String[] args) {
```

```
        int[] array = new int[10];
        try {
            System.out.println(array[11]);
            System.out.println("Ini adalah baris "
                    + "yang akan dijalankan exception");
        } catch (Exception e) {
            System.out.println("Terjadi sebuah array "
                    + "index out of bound");
        }
        System.out.println("Keluar dari catch");
    }
}
```

Penjelasan

Program di atas merupakan contoh penggunaan langsung bagaimana exception handling bekerja. Pada pemanggilan array, dipanggil array ke-11. Yang dimana hal tersebut akan berujung error dikarenakan array yang tersedia hanya sampai array ke-10. Maka, program akan menjalankan catch dan menampilkan pesan yang telah disediakan.

3. TryCatch2.java

```
package main.java.com.Alpian.Pertemuan_9.Guided;
/**
 * @author
 * Alif Alpian Sahrul Muharom
 * 20102007
 * IF-08-0
 */
public class TryCatch2 {
    public static void main(String[] args) {
        try {
            int x = 0;
            int y = 10/x;
            int[] array = {10,11};
            y = array[x];
            System.out.println("Tidak terjadi Exception");
        } catch (ArithmeticException e) {
            System.out.println("Tidak dapat dibagi 0");
        }
        catch (ArrayIndexOutOfBoundsException e)  {
            System.out.println("Indeks di luar kapasitas array");
        }
        System.out.println("Setelah blok try catch");
    }
}
```

Penjelasan

Program di atas merupakan contoh penggunaan langsung bagaimana exception handling bekerja. Pada pemanggilan array, dipanggil array ke-11. Yang dimana hal tersebut akan berujung error dikarenakan array yang tersedia hanya sampai array ke-10. Maka, program akan menjalankan catch dan menampilkan pesan yang telah disediakan.

4. demoFinally.java

```
package main.java.com.Alpian.Pertemuan_9.Guided;
/**

 * @author
 * Alif Alpian Sahrul Muharom
 * 20102007
 * IF-08-0
 */
public class demoFinally {
    public static void main(String[] args) {
        int x = 3;
        int[] array = {10, 11, 12};
        try {
            System.out.println(array[x]);
            System.out.println("Tidak terjadi kesalahan exeption");
        } catch (ArrayIndexOutOfBoundsException e) {
            //TODO: handle exception
            System.out.println("Terjadi exeption");
            //System.out.println(array[x-4]);
        }
        finally{
            // finally berarti baris ini akan tetap dijalankan
meskipun
            //terjadi kesalahan atau tidak terjadi kesalahan.
            System.out.println("Program selesai dijalankan");
        }
    }
}
```

Penjelasan

Finally merupakan pelengkap dari exception handling. Finally akan tetap dijalankan walaupun program tidak error. Syntax ini jarang digunakan.

5. DemoStream1.java

```
package main.java.com.Alpian.Pertemuan_9.Guided;
/**
 * @author
 * Alif Alpian Sahrul Muharom
```

```
 * 20102007
 * IF-08-0
 */
public class demoStream1 {
    public static void main(String[] args) {
        byte[] data = new byte[10];
        System.out.print("Masukkan data : ");
        try {
            System.in.read(data);
        } catch (IOException e) {
            System.out.println("Terjadi Exception");
        }
        System.out.print("yang anda ketik adalah : ");
        for (int i = 0; i < data.length; i++) {
            System.out.print((char)data[i]);
        }
    }
}
```

Penjelasan

Karena merupakan handler exception I/O, maka library *java.io.IOException* digunakan. Karena merupakan String, maka program dapat membaca dan menampilkan ulang apa yang user inputkan.

6. DemoStream2.java

```
package main.java.com.Alpian.Pertemuan_9.Guided;
/**
 * @author
 * Alif Alpian Sahrul Muharom
 * 20102007
 * IF-08-0
 */
public class demoStream2 {
    public static void main(String[] args) {
        byte[] data = new byte[10];
        int panjang = 0;
        System.out.print("Masukkan data : ");
        try {
            panjang = System.in.read(data);
            System.out.print("yang anda ketik : ");
            System.out.write(data);
            System.out.print("Panjang karakter : " + panjang);
            System.out.print("\nindex ke -1 sebanyak 3 : ");
            System.out.write(data,1,3);
```

```
            System.out.print("\n");

        } catch (IOException e) {
            System.out.println("Terjadi exception");
        }
    }
}
```

Penjelasan

Program tersebut meminta user untuk menginputkan sesuatu dan program akan mengeluarkan output berupa apa yang user ketik, panjang karakternya, dan menampilkan index ke 1 sebanyak 3.

7. DemoStream3.java

```
package main.java.com.Alpian.Pertemuan_9.Guided;
/**
 * @author
 * Alif Alpian Sahrul Muharom
 * 20102007
 * IF-08-0
 */
public class demoStream3 {
    public static void main(String[] args) throws IOException {
        char data;
        String str = "";
        BufferedReader buff = new BufferedReader(new
        InputStreamReader(System.in));

        System.out.println("Ketik: ");
        data = (char) buff.read();
        while (data != '\n') {
            str += data;
            data = (char) buff.read();
        }
        System.out.println("Yang diketik: " + str);
        System.out.println("Program Selesai");
    }
}
```

Penjelasan

Throws IOException Adalah suatu Method yang Membaca Input Data String. Jika tidak ada error, maka program akan jalan sampai selesai.

8. DemoStream4.java

```
package main.java.com.Alpian.Pertemuan_9.Guided;
/**
```

```
 * @author
 * Alif Alpian Sahrul Muharom
 * 20102007
 * IF-08-0
 */
public class demoStream4 {
    public static void main(String[] args) {
        byte data;
        String namaFile = "demo.txt";
        FileOutputStream fout = null;
        try {
            fout = new FileOutputStream(namaFile, true);
            System.out.print("Ketik : ");
            data = (byte)System.in.read();
            while (data!=(byte)'\n') {
                fout.write(data);
                data = (byte)System.in.read();
            }
        } catch (FileNotFoundException e) {
            System.out.println("File"+namaFile+"Tidak dapat dibuat");
        } catch (IOException e){
            System.out.println("Terjadi Exception");
        }

        finally {
            if (fout!=null){
                try {
                    fout.close();
                } catch (IOException e) {
                    System.out.println("Terjadi Exception");
                }
            }
        }
    }
}
```

Penjelasan

Program di atas akan membuat sebuah file txt bernama demo dan akan menampilkan apa yang user tulis ke dalam file txt tersebut. jika file tersebut error, maka fungsi try dan catch akan dijalankan dan menampilkan pesan error. Finally pun akan ikut dijalankan.

9. BarangEx.java

```
package main.java.com.Alpian.Pertemuan_9.Guided;
/**
```

```
 * @author
 * Alif Alpian Sahrul Muharom
 * 20102007
 * IF-08-0
 */
public class BarangEx implements Externalizable {
    private String nama;
    private int jumlah;

    public BarangEx() { }

    public BarangEx(String nm, int jml) {
        nama = nm;
        jumlah = jml;
    }

    public void writeExternal(ObjectOutput out) throws IOException {
        out.writeObject(nama);
        out.writeInt(jumlah);
    }

    public void readExternal(ObjectInput in) throws IOException,
ClassNotFoundException {
        this.nama = (String) in.readObject();
        this.jumlah = in.readInt();
    }

    public String toString() {
        return "data barang: " + nama + "\n" + "jumlah barang: " +
jumlah;
    }

    public static void simpanObject(BarangEx brg) throws IOException
{
        FileOutputStream fos = new FileOutputStream("dtEx.txt");
        ObjectOutputStream oos = new ObjectOutputStream(fos);
        oos.writeObject(brg);
        oos.flush();
    }

    public static BarangEx bacaObject()
            throws ClassNotFoundException, IOException {
        FileInputStream fis = new FileInputStream("dtEx.txt");
        ObjectInputStream ois = new ObjectInputStream(fis);
```

```
        return (BarangEx) ois.readObject();
    }

    public static void main(String[] args)
            throws ClassNotFoundException, IOException {
        BarangEx awal = new BarangEx("sepatu", 2);
        simpanObject(awal);
        System.out.println(bacaObject());
    }
}
```

10. BarangSer.java

```
package main.java.com.Alpian.Pertemuan_9.Guided;
/**
 * @author
 * Alif Alpian Sahrul Muharom
 * 20102007
 * IF-08-0
 */
class BarangSer implements Serializable {
    private String nama;
    private int jumlah;

    public BarangSer(String nm, int jmlh){
        nama = nm;
        jmlh = jumlah;
    }

    public void tampil(){
        System.out.println("Nama Barang : " + nama);
        System.out.println("Jumlah Barang : " + jumlah);
    }

    public void simpanObject(BarangSer ob){
        try {
            FileOutputStream fos = new
FileOutputStream("dataBarang.txt");
            ObjectOutputStream oos = new ObjectOutputStream(fos);
            oos.writeObject(ob);
            oos.flush();
        } catch (IOException ioe) {
            System.out.println("Error"+ioe);
        }
    }
```

```
    public void bacaObject(BarangSer obb){
        try {
            FileInputStream fis = new
FileInputStream("dataBarang.txt");
            ObjectInputStream ois = new ObjectInputStream(fis);

            while ((obb = (BarangSer)ois.readObject())!=null) {
                obb.tampil();
            }
        } catch (IOException ioe) {
            System.exit(1);
        } catch (Exception e){
            System.exit(1);
        }
    }

    public static void main(String[] args) {
        BarangSer a1 = new BarangSer("Motor", 5);
        a1.simpanObject(a1);
        a1.bacaObject(a1);
    }
}
```

Penjelasan

Perbedaan utama antara Serializable dan Externalizable

- Penanda antarmuka: Serializable adalah antarmuka penanda tanpa metode apa pun. Antarmuka Externalizable berisi dua metode: writeExternal() dan readExternal().
- Proses serialisasi: Proses serialisasi default akan dimulai untuk kelas yang mengimplementasikan antarmuka Serializable. Programer mendefinisikan proses serialisasi akan dimulai untuk kelas yang mengimplementasikan antarmuka Externalizable.
- Pemeliharaan: Perubahan yang tidak kompatibel dapat memutus serialisasi.
- Mundur Kompatibilitas dan Kontrol: Jika Anda harus mendukung beberapa versi, Anda dapat memiliki kontrol penuh dengan antarmuka Externalizable. Anda dapat mendukung berbagai versi objek Anda. Jika Anda menerapkan Externalizable, Anda bertanggung jawab untuk membuat serial super kelas
- 5. public No-arg constructor: Serializable menggunakan refleksi untuk membangun objek dan tidak memerlukan konstruktor arg. Tapi Externalizable menuntut konstruktor tanpa argumen publik.

V. KESIMPULAN

Kesalahan pada program yang akan dibuat bisa berasal dari 3 hal, yaitu kesalahan **hardware**, **software**, maupun **keduanya**. Jadi, diperlukan exception handling agar program tidak langsung crash atau berhenti secara paksa. Exception Handling dapat dilakukan menggunakan *keyword* try-catch, dimana pada artikel ini akan dijelaskan bagaimana melakukannya dalam Bahasa pemrograman Java. Throw dan finally juga dapat digunakan tergantung kondisi yang dibutuhkan oleh programmer.