Modul Praktikum

Pemrograman Berorientasi Objek

Jurusan Ilmu Komputer
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Lampung

2022
Semester Ganjil

Modul 12. Error Handling

Saat mengeksekusi kode Java, kesalahan yang berbeda dapat terjadi: kesalahan pengkodean yang dibuat oleh programmer, kesalahan karena input yang salah, atau hal-hal lain yang tidak terduga.

Ketika terjadi error maka Java akan berhenti menjalankan program dan memberikan pesan error. Istilah teknisnya Java will throw an exception (throw an error).

Capaian Pembelajaran

1. Mahasiswa mampu menangani error di bahasa pemrograman java.
2. Mahasiswa mampu menerapkan error handling di bahasa pemrograman java.
3. Mahasiswa mampu memahami penggunaan ArrayList pada Java

Materi

Java try and catch

Keyword try memungkinkan Anda untuk menentukan blok kode yang akan diuji untuk error saat sedang dieksekusi.

Keyword catch memungkinkan Anda untuk menentukan blok kode yang akan dieksekusi, jika terjadi error pada blok try.

Kata kunci try dan catch digunakan secara berpasangan:

```
try {
  //  Kode yang akan dicoba
}
catch(Exception e) {
  //  Kode yang dijalankan jika terjadi error pada try
}
```

Finally

Pernyataan finally memungkinkan Anda mengeksekusi kode, setelah try...catch, terlepas dari apapun hasilnya, contoh:

```
public class Main {
  public static void main(String[] args) {
    try {
      int[] num = {1, 2, 3};
      System.out.println(num[10]);
    } catch (Exception e) {
      System.out.println("Terjadi error");
    } finally {
      System.out.println("Blok kode finally");
    }
  }
}
```

Output:

```
Terjadi error
Blok kode finally
```

Keyword Throw

Pernyataan throw memungkinkan Anda membuat *custom error*.

Pernyataan throw digunakan bersama dengan tipe exception. Ada banyak tipe exception yang tersedia di Java: ArithmeticException, FileNotFoundException, ArrayIndexOutOfBoundsException, SecurityException, dll. Contoh:

Beri exception jika usia di bawah 18 (cetak "Akses ditolak"). Jika usia 18 tahun atau lebih, cetak "Akses diberikan":

```
public class Main {
  static void checkAge(int age) {
    if (age < 18) {
      throw new ArithmeticException("Akses ditolak");
    } else {
      System.out.println("Akses diberikan");
    }
 }

 public static void main(String[] args) {
   checkAge(15);
 }
}
```

Output:

```
Exception in thread "main" java.lang.ArithmeticException: Akses ditolak
        at Main.checkAge(Main.java:4)
        at Main.main(Main.java:11)
```

Java ArrayList

Class ArrayList adalah array yang dapat diubah ukurannya, yang dapat ditemukan dalam package java.util.

Perbedaan antara array built-in dan ArrayList di Java, ukuran array tidak dapat diubah (jika Anda ingin menambah atau menghapus elemen ke/dari array, Anda harus membuat yang array baru). Sementara elemen dapat ditambahkan dan dihapus dari ArrayList kapan pun Anda mau.

Contoh deklarasi dan menambah data pada ArrayList:

```java
import java.util.ArrayList;

public class Main {
  public static void main(String[] args) {
    ArrayList<String> cars = new ArrayList<String>();
    cars.add("Volvo");
    cars.add("BMW");
    cars.add("Ford");
    cars.add("Mazda");
    System.out.println(cars);
  }
}
```

Output:

```
[Volvo, BMW, Ford, Mazda]
```

Beberapa method yang sering digunakan pada ArrayList diantaranya:

`add(value)`	appends value at end of list
`add(index, value)`	inserts given value just before the given index, shifting subsequent values to the right
`clear()`	removes all elements of the list
`indexOf(value)`	returns first index where given value is found in list (-1 if not found)
`get(index)`	returns the value at given index
`remove(index)`	removes/returns value at given index, shifting subsequent values to the left
`set(index, value)`	replaces value at given index with given value
`size()`	returns the number of elements in list

Langkah Praktikum

Mencoba ArrayList

1. Pada kelas utama di fungsi main. Lakukan import ArrayList.

```
import java.util.ArrayList;
```

2. Buatlah sebuah ArrayList dengan tipe Integer, kemudian tambahkan tiga nilai untuk mengisi ArrayList tersebut. Cetak ArrayList untuk memeriksa isinya.

```
public static void main(String[] args) {

        ArrayList<Integer> nilai = new ArrayList<Integer>();
        nilai.add(70);
        nilai.add(75);
        nilai.add(80);

        System.out.println(nilai);
    }
```

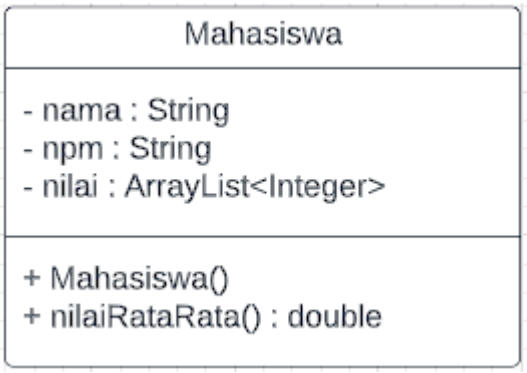
Output:

```
[70, 75, 80]
```

Membuat Class Mahasiswa

1. Buatlah sebuah **class** dengan nama **Mahasiswa**. Sesuaikan isinya dengan class diagram berikut:

Class Diagram



2. Buatlah Constructor, Setter, dan Getter untuk semua atribut nya.
3. Keseluruhan code untuk class ini akan terlihat sebagai berikut:

```
package praktikum6;

import java.util.ArrayList;

public class Mahasiswa {
    private String nama;
    private String npm;
    private ArrayList<Integer> nilai = new ArrayList<Integer>();

    public Mahasiswa(String nama, String npm, ArrayList<Integer> nilai) {
        this.nama = nama;
        this.npm = npm;
        this.nilai = nilai;
    }

    public String getNama() {
        return nama;
    }

    public void setNama(String nama) {
        this.nama = nama;
    }

    public String getNpm() {
        return npm;
    }

    public void setNpm(String npm) {
        this.npm = npm;
    }

    public ArrayList<Integer> getNilai() {
        return nilai;
    }

    public void setNilai(ArrayList<Integer> nilai) {
        this.nilai = nilai;
    }


}
```

4. Tambahkan sebuah method untuk menghitung nilai rata-rata dari ArrayList nilai dengan nama nilaiRataRata. Method ini mengembalikan nilai rata-rata dalam tipe data double.

   Gunakan method get(index) pada ArrayList untuk mengambil setiap elemennya dan size() untuk mengetahui jumlah elemen pada array.

```
public double nilaiRataRata(){
    int sum = 0;
    for (int i = 0; i < this.nilai.size(); i++) {
        sum += this.nilai.get(i);
    }
    return sum/this.nilai.size();
}
```

5. Pada fungsi main di class utama, buatlah sebuah objek Mahasiswa dan isi nilai dengan nilai yang sudah dibuat pada bagian pertama.

```
public static void main(String[] args) {

        ArrayList<Integer> nilai = new ArrayList<Integer>();
        nilai.add(70);
        nilai.add(75);
        nilai.add(80);
        System.out.println(nilai);

        Mahasiswa mhs1 = new Mahasiswa("Naufal", "2017051072", nilai);
    }
```

6. Panggil method nilaiRataRata() dari mhs1 dan cetak hasilnya.

```
System.out.println("Nilai rata-rata : " + mhs1.nilaiRataRata());
```

   Output:

```
[70, 75, 80]
Nilai rata-rata : 75.0
```

7. Sejauh ini tidak terjadi masalah, namun cobalah untuk menghapus baris penambahan nilai berikut dan jalankan lagi programnya.

```
nilai.add(70);
nilai.add(75);
nilai.add(80);
```

Output:

```
[]
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at praktikum6.Mahasiswa.nilaiRataRata(Mahasiswa.java:45)
        at praktikum6.Praktikum6.main(Praktikum6.java:13)
```

8. Muncul error ArithmeticException: / by zero. Error tersebut muncul karena terdapat operasi pembagian dengan 0 pada method nilaiRataRata(). Pembagian dengan 0 terjadi karena array yang diberikan adalah kosong.
9. Untuk mengatasi hal tersebut, kembali ke class Mahasiswa dan tambahkan error handling berikut pada method nilaiRataRata().

```
public double nilaiRataRata(){
    int sum = 0;
    for (int i = 0; i < this.nilai.size(); i++) {
        sum += this.nilai.get(i);
    }

    try {
        return sum/this.nilai.size();
    } catch (Exception e) {
        return 0;
    }
}
```

Kode diatas akan mengembalikan 0 jika array nilai kosong.

Mengubah Setter nilai pada Class Mahasiswa dengan Custom Error

1. Untuk mencegah terjadinya error seperti diatas, Anda juga dapat mengatur setter pada class Mahasiswa agar tidak menerima ArrayList kosong.
2. Ubah setter nilai dengan melakukan cek apakah array yang diterima berukuran 0 atau tidak. Jika berukuran 0, berikan error SecurityException.

```
public void setNilai(ArrayList<Integer> nilai) {
    if (nilai.size() == 0) {
        throw new SecurityException("Array tidak boleh kosong!");
    }else{
        this.nilai = nilai;
    }
}
```

3. Jika anda jalankan lagi programnya masih tidak terjadi perubahan. Itu karena anda belum menggunakan fungsi setNilai().
4. Cobalah memanggil setNilai() pada fungsi main pada kelas utama dengan memberikan array kosong.

```
public static void main(String[] args) {

        ArrayList<Integer> nilai = new ArrayList<Integer>();
        System.out.println(nilai);

        Mahasiswa mhs1 = new Mahasiswa("Naufal", "2017051072", nilai);

        mhs1.setNilai(nilai);

        System.out.println("Nilai rata-rata : " + mhs1.nilaiRataRata());
    }
```

5. Program akan memberikan error sesuai pesan yang dimasukkan.
6. Cobalah sendiri untuk mengubah konstruktor agar menggunakan setter yang telah dibuat.