LAPORAN PRAKTIKUM STRUKTUR DATA

Modul ke : 4
Judul Praktikum : Object Comparable & Comparator
Hari dan Tanggal Pelaksanaan : Jumat, 22 April 2022
Tempat Pelaksanaan : Lab Desain
Dosen Pengampu Praktikum : Khoirul Umam, S.Pd, M.Kom

Nama Mahasiswa Pelaksana : Andri Wijaksono
NIM Pelaksana : 362155401206

A. Tugas Pendahuluan

Tuliskan hasil pengerjaan Tugas Pendahuluan pada bagian ini.

Buatlah resume mengenai interface Comparable dan Comparator.
Jawab:

1. **Interface Comparable**

   Interface Comparable berasal dari package java.lang. Interface ini memiliki sebuah method bernama compareTo() yang wajib di-override oleh class yang mengimplementasikan interface tersebut. Method compareTo() menerima satu buah parameter bertipe Object dan mengembalikan nilai (return value) bertipe integer seperti yang digambarkan pada gambar beriIut:

   Public int compareTo(Object0)

   Return value dari method compareTo() :
   - 0 jika dua object yang dibandingkan sama,
   - Bilangan positif, jika object 1 lebih besar dibandingkan dengan object 2,
   - Bilangan negatif, jika object 1 lebih kecil dibandingkan dengan object 2
   -

2. **Interface Comparator**

   Pengimplementasian interface Comparable pada suatu class hanya akan menetapkan properti default dari class tersebut yang digunakan sebagai dasar perbandingan. Dengan demikian setiap objek dari class terkait hanya dapat dibandingkan berdasarkan properti default tersebut. Lalu bagaimana jika ternyata perbandingan yang dibutuhkan harus menggunakan properti lainnya yang juga dimiliki oleh class tersebut? Pada kasus ini interface Comparator menyediakan solusinya.

   Untuk menerapkan perbandingan antarobjek yang didasarkan pada suatu properti tertentu, maka harus dibuat terebih dahulu sebuah class yang mendefinisikan perbandingan tersebut. Class yang nantinya disebut sebagai comparator class tersebut harus mengimplementasikan interface Comparator dan meng-override method compare(). Method ini serupa dengan method compareTo() pada interface Comparable, namun menerima dua buah parameter bertipe Object yang akan dibandingkan. Gambaran dari method tersebut ditunjukkan oleh gambar berikut:

   - **public int compare(Object o1, Object o2)**

B. Kegiatan Praktikum

Cantumkan apa saja yang dilakukan pada latihan-latihan praktikum, *source code* yang dipakai, *screen shot* hasil eksekusi kode, dan jawaban dari pertanyaan-pertanyaan yang muncul pada tiap kegiatan latihan.

Latihan 1: Penggunaan Java native sorting method untuk data string

1. Buat sebuah file bernama **ArrayString.java** kemudian tuliskan kode berikut:

```
import java.util.Arrays;
public class ArrayString {
    public static void main(String args[]) {
        String animals[] = new String[6];
            animals[0] = "snake";
            animals[1] = "kangaroo";
            animals[2] = "wombat";
            animals[3] = "bird";
            System.out.println("\nSEBELUM DISORTING");
            for (int i = 0; i < 4; i++) {
            System.out.println("animal " + i + " : " + animals[i]);
            }
            Arrays.sort(animals, 0, 4);
            System.out.println("\nSETELAH DISORTING");
            for (int i = 0; i < 4; i++) {
                System.out.println("animal " + i + " : " + animals[i]);
                }
    }
}
```

2. Buat lagi sebuah file bernama **ArrayListString.java** kemudian tuliskan kode berikut ke dalamnya:

```
import java.util.ArrayList;
import java.util.Collections;
public class ArrayListString {
    public static void main(String args[]) {
        ArrayList<String> animals = new ArrayList<String>();
            animals.add("snake");
            animals.add("kangaroo");
            animals.add("wombat");
            animals.add("bird");
        System.out.println("\nSEBELUM DISORTING");
            for (int i = 0; i < animals.size(); i++) {
                System.out.println("animal " + i + " : " + animals.get(i));
                }
            Collections.sort(animals);
            System.out.println("\nSETELAH DISORTING");
```

```
                for (int i = 0; i < animals.size(); i++) {
                    System.out.println("animal " + i + " : " +
animals.get(i));
                }
    }
}
```

3. Compile dan jalankan kode pada class **ArrayString** dan perhatikan hasilnya.

```
PS C:\Users\ASUS\Documents\Struktur Data\modul4> java ArrayString

SEBELUM DISORTING
animal 0 : snake
animal 1 : kangaroo
animal 2 : wombat
animal 3 : bird

SETELAH DISORTING
animal 0 : bird
animal 1 : kangaroo
animal 2 : snake
animal 3 : wombat
```

4. Compile dan jalankan pula kode pada class **ArrayListString** dan perhatikan hasilnya.

```
PS C:\Users\ASUS\Documents\Struktur Data\modul4> java ArrayListString

SEBELUM DISORTING
animal 0 : snake
animal 1 : kangaroo
animal 2 : wombat
animal 3 : bird

SETELAH DISORTING
animal 0 : bird
animal 1 : kangaroo
animal 2 : snake
animal 3 : wombat
```

5. Apa fungsi dari method **Arrays**.**sort**()?
Jawab: Mengurutkan array yang diindeksnya dalam urutan menaik.
6. Apa fungsi dari method **Collections**.**sort**()?
Jawab: Untuk tipe data objek seperti ArrayList dan LinkedList
7. Apakah method **Arrays**.**sort**() dan **Collections**.**sort**() memberikan hasil yang sama?
Jawab: Iya memberikan hasil yang sama.

Latihan 2: Penggunaan Java native sorting method untuk data berupa objek

1. Buat sebuah file bernama **Mahasiswa**.**java** kemudian tuliskan kode berikut:

```
public class Mahasiswa {
    private String nrp;
    private String nama;
    public Mahasiswa(String nrp, String nama) {
        this.nrp = nrp;
        this.nama = nama;
```

```
    }
    public String getNama() {
        return nama;
    }
    public void setNama(String nama) {
    this.nama = nama;
    }
    public String getNrp() {
    return nrp;
    }
    public void setNrp(String nrp) {
    this.nrp = nrp;
    }
        @Override
    public String toString() {
    return "Mahasiswa {nrp=" + nrp + " nama=" + nama + "}";
    }
}
```

2. Buat juga sebuah file bernama **TestMahasiswa.java** dan tuliskan kode berikut ke dalamnya:

```
import java.util.Arrays;
public class TestMahasiswa {
public static void main(String[] args) {
        Mahasiswa dataMhs[] = {
            new Mahasiswa("05", "Cahya"),
            new Mahasiswa("04", "Rudi"),
            new Mahasiswa("01", "Endah"),
            new Mahasiswa("03", "Rita"),
            new Mahasiswa("02", "Tika")
            };
    System.out.println("SEBELUM SORTING");
    System.out.println(Arrays.toString(dataMhs));
    Arrays.sort(dataMhs);
    System.out.println();
    System.out.println("SESUDAH SORTING");
    System.out.println(Arrays.toString(dataMhs));
        }
}
```

3. Compile file **TestMahasiswa.java** dan jalankan kode tersebut kemudian perhatikan hasilnya.

```
SEBELUM SORTING
[Mahasiswa {nrp=05 nama=Cahya}, Mahasiswa {nrp=04 nama=Rudi}, Mahasiswa {nrp=01 nama=Endah}, Mahasiswa {nrp=03 nama=Rita}, Mahasiswa {nrp=02 nama=Tika}]
Exception in thread "main" java.lang.ClassCastException: class Mahasiswa cannot be cast to class java.lang.Comparable (Mahasiswa is in unnamed module of loader 'app'; ja
va.lang.Comparable is in module java.base of loader 'bootstrap')
        at java.base/java.util.ComparableTimSort.countRunAndMakeAscending(ComparableTimSort.java:320)
        at java.base/java.util.ComparableTimSort.sort(ComparableTimSort.java:188)
        at java.base/java.util.Arrays.sort(Arrays.java:1040)
        at TestMahasiswa.main(TestMahasiswa.java:13)
PS C:\Users\ASUS\Documents\Struktur Data\modul4>
```

4. Nampak bahwa saat array yang berisi objek-objek dari class Mahasiswa diurutkan menggunakan method **Arrays.sort()** akan memunculkan sebuah exception. Jelaskan mengapa **exception** tersebut terjadi?
Jawab:

Latihan 3: Implementasi interface Comparable

1. Modifikasi class Mahasiswa pada file **Mahasiswa.java** agar mengimplementasikan interface **Comparable** lalu definisikan juga method **compareTo()** di dalam class tersebut seperti berikut:

```
import java.lang.Comparable;

public class Mahasiswa implements Comparable<Mahasiswa>  {
    private String nrp;
    private String nama;
    public Mahasiswa(String nrp, String nama) {
        this.nrp = nrp;
        this.nama = nama;
    }
    public String getNama() {
        return nama;
    }
    public void setNama(String nama) {
    this.nama = nama;
    }
    public String getNrp() {
    return nrp;
    }
    public void setNrp(String nrp) {
    this.nrp = nrp;
    }
        @Override
    public String toString() {
    return "Mahasiswa {nrp=" + nrp + " nama=" + nama + "}";
    }

    public int compareTo(Mahasiswa m) {
        return this.nrp.compareTo(m.nrp);
        }

}
```

2. Compile kembali file **TestMahasiswa.java** kemudian jalankan ulang kode tersebut

```
import java.util.Arrays;
```

```
public class TestMahasiswa {
public static void main(String[] args) {
        Mahasiswa dataMhs[] = {
            new Mahasiswa("05", "Cahya"),
            new Mahasiswa("04", "Rudi"),
            new Mahasiswa("01", "Endah"),
            new Mahasiswa("03", "Rita"),
            new Mahasiswa("02", "Tika")
            };
    System.out.println("SEBELUM SORTING");
    System.out.println(Arrays.toString(dataMhs));
    Arrays.sort(dataMhs);
    System.out.println();
    System.out.println("SESUDAH SORTING");
    System.out.println(Arrays.toString(dataMhs));
        }
}
```

```
SEBELUM SORTING
[Mahasiswa {nrp=05 nama=Cahya}, Mahasiswa {nrp=04 nama=Rudi}, Mahasiswa {nrp=01 nama=Endah}, Mahasiswa {nrp=03 nama=Rita}, Mahasiswa {nrp=02 nama=Tika}]

SESUDAH SORTING
[Mahasiswa {nrp=01 nama=Endah}, Mahasiswa {nrp=02 nama=Tika}, Mahasiswa {nrp=03 nama=Rita}, Mahasiswa {nrp=04 nama=Rudi}, Mahasiswa {nrp=05 nama=Cahya}]
PS C:\Users\ASUS\Documents\Struktur Data\modul4>
```

3. Apakah objek-objek mahasiswa tersebut dapat diurutkan? Jika iya, properti apa yang menjadi patokan pengurutan tersebut?
Jawab:
4. Method apa yang berperan penting dalam proses pengurutan objek sejenis menggunakan **Arrays**.**sort**()?
Jawab:

Latihan 4: Implementasi interface Comparator

1. Buat sebuah file baru bernama **NamaComparator**.**java** kemudian tuliskan kode berikut:

```
import java.util.Comparator;
public class NamaComparator implements Comparator<Mahasiswa> {
    public int compare(Mahasiswa m1, Mahasiswa m2) {
        return m1.getNama().compareTo(m2.getNama());
    }
}
```

2. Modifikasi class **TestMahasiswa** pada file **TestMahasiswa**.**java** agar objek dari class **NamaComparator** digunakan sebagai parameter kedua pada method **Arrays**.sort():

```
import java.util.Arrays;
public class TestMahasiswa {
public static void main(String[] args) {
        Mahasiswa dataMhs[] = {
            new Mahasiswa("05", "Cahya"),
            new Mahasiswa("04", "Rudi"),
            new Mahasiswa("01", "Endah"),
            new Mahasiswa("03", "Rita"),
            new Mahasiswa("02", "Tika")
            };
    System.out.println("SEBELUM SORTING");
    System.out.println(Arrays.toString(dataMhs));
    Arrays.sort(dataMhs, new NamaComparator());
    System.out.println();
    System.out.println("SESUDAH SORTING");
    System.out.println(Arrays.toString(dataMhs));
        }
}
```

```
SEBELUM SORTING
[Mahasiswa {nrp=05 nama=Cahya}, Mahasiswa {nrp=04 nama=Rudi}, Mahasiswa {nrp=01 nama=Endah}, Mahasiswa {nrp=03 nama=Rita}, Mahasiswa {nrp=02 nama=Tika}]

SESUDAH SORTING
[Mahasiswa {nrp=05 nama=Cahya}, Mahasiswa {nrp=01 nama=Endah}, Mahasiswa {nrp=03 nama=Rita}, Mahasiswa {nrp=04 nama=Rudi}, Mahasiswa {nrp=02 nama=Tika}]
PS C:\Users\ASUS\Documents\Struktur Data\modul4>
```

4. Apakah objek-objek mahasiswa masih tetap dapat diurutkan? Jika iya, berdasarkan properti apa pengurutan itu terjadi?

Latihan 5: Comparator untuk pengurutan secara descending

1. Buat file bernama **DescendingOrderComparator.java** kemudian tulis kode berikut:

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;

    public class DescendingOrderComparator {
        public static void main(String[] args) {
            // create an ArrayList object
            ArrayList<String> arrayList = new ArrayList<String>();
            // Add elements to Arraylist
            arrayList.add("A");
            arrayList.add("B");
            arrayList.add("C");
            arrayList.add("D");
            arrayList.add("E");
    System.out.println("Before sorting ArrayList in descending order :"
+arrayList);
        Comparator<String> comparator = Collections.reverseOrder();
            Collections.sort(arrayList, comparator);
```

```
            System.out.println("After sorting ArrayList in descending order :
" +arrayList);
    }
}
```

```
Before sorting ArrayList in descending order :[A, B, C, D, E]
After sorting ArrayList in descending order : [E, D, C, B, A]
PS C:\Users\ASUS\Documents\Struktur Data>
```

3. Apa fungsi dari method **Colections.reverseOrder**()?
Jawab: Digunakan untuk mendapatkan pembanding yang memaksakan dari urutan alami pada kumpulan objek yang mengimplementasikan antarmuka sebanding.

4. Apakah hasil pengurutan berupa pengurutan menurun (descending)?
Jawab:

C. Tugas Praktikum

Tuliskan dan jabarkan hasil pengerjaan Tugas Praktikum yang tertera di dalam modul lengkap dengan *source code* yang digunakan.

Kembangkan class **Mahasiswa** pada kegiatan latihan di atas dengan menambahkan properti baru berupa nilai **IPK** yang bertipe **double** atau **float**. Selanjutnya lakukan pengurutan objek Mahasiswa berdasarkan NRP, nama, serta IPK-nya dengan memanfaatkan interface **Comparable** maupun **Comparator**.

```
import java.lang.Comparable;


public class Mahasiswa implements Comparable<Mahasiswa>{
    private String nrp;
    private String nama;
    private String IPK;
    public Mahasiswa(String nrp, String nama, String IPK) {
        this.nrp = nrp;
        this.nama = nama;
        this.IPK = IPK;
    }
    public String getNama() {
        return nama;
    }
    public void setNama(String nama) {
        this.nama = nama;
    }
    public String getNrp() {
        return nrp;
    }
    public void setNrp(String nrp) {
        this.nrp = nrp;
    }
    public String getIPK() {
        return IPK;
```

```
    }
    public void setIPK(String IPK) {
        this.IPK = IPK;
    }
    @Override
    public String toString() {
        return "Mahasiswa {nrp=" + nrp + " nama=" + nama + " IPK=" + IPK +"}";
    }

    @Override
    public int compareTo(Mahasiswa m) {
        return this.nrp.compareTo(m.nrp);
    }

}
```

```
import java.util.Comparator;
public class IPKComparator implements Comparator<Mahasiswa> {
    public int compare(Mahasiswa m1, Mahasiswa m2) {
        return m1.getIPK().compareTo(m2.getIPK());
    }
}
```

```
import java.util.Arrays;

public class Main {
    public static void main(String[] args) {
        Mahasiswa dataMhs[] = {new Mahasiswa("01","John F Kenedy","3.1"),
        new Mahasiswa("02","Putin","3.4"), new
Mahasiswa("03","HiroHito","3.8"),
        new Mahasiswa("04","Obama","2.9"), new Mahasiswa("05","Hitler","3.0")
        };
        System.out.println("SEBELUM DISORTING");
        System.out.println(Arrays.toString(dataMhs));
        Arrays.sort(dataMhs, new IPKComparator());
        System.out.println("\nSESUDAH DISORTING");
        System.out.println(Arrays.toString(dataMhs));
    }
}
```

```
SEBELUM DISORTING
[Mahasiswa {nrp=01 nama=John F Kenedy IPK=3.1}, Mahasiswa {nrp=02 nama=Putin IPK=3.4}, Mahasiswa {nrp=03 nama=HiroHito IPK=3.8}, Mahasiswa {nrp=04 nama=O
bama IPK=2.9}, Mahasiswa {nrp=05 nama=Hitler IPK=3.0}]

SESUDAH DISORTING
[Mahasiswa {nrp=04 nama=Obama IPK=2.9}, Mahasiswa {nrp=05 nama=Hitler IPK=3.0}, Mahasiswa {nrp=01 nama=John F Kenedy IPK=3.1}, Mahasiswa {nrp=02 nama=Put
in IPK=3.4}, Mahasiswa {nrp=03 nama=HiroHito IPK=3.8}]
PS C:\Users\ASUS\Documents\Struktur Data>
```