Generic Programming: bagaimana caranya merancang dan mengimplementasikan struktur data atau algoritma yang dapat bekerja untuk berbagai jenis tipe data?

1. Comparable<T>

Perhatikan bahwa sesungguhnya interface **Comparable** adalah sebuah interface generic. Hal ini berdampak pada signature method **compareTo**.

```
public interface Comparable<T> {
   public int compareTo(T other);
}
```

Coba Anda buat kelas **City** yang mempunyai dua buah atribut: **nama (String)** & **luas (int)**. Kemudian, Anda ingin mengurutkan objek-objek **City** dengan aturan berikut:

- Urutkan objek **City** (kota) dari yang **paling luas** ke yang **paling sempit**.
- Jika ada 2 objek **City** yang luasnya sama, maka urutkan berdasarkan nama-nya secara alfabetis (leksikografis).

Gunakan Comparable yang menggunakan generics !

Coba jelaskan apa bedanya jika Anda menggunakan **Comparable yang non-generics** ?

2. Generic Class

Perhatikan implementasi dari kelas **Pair** berikut

```
public class Pair<T, S>
{
    private T first;
    private S second;

    public Pair(T firstElement, S secondElement)
    {
        first = firstElement;
        second = secondElement;
    }

    public T getFirst() { return first; }
    public S getSecond() { return second; }
}
```

a). Ada berapa **type parameter** pada kelas **Pair** tersebut ?

b). Perhatikan implementasi kelas **Person** berikut

```
public class Person {
    private String nama;
    private int    umur;
    private int    id;

    // setter – getter dari masing-masing atribut
    ...
}
```

Gunakan kelas Pair untuk mengimplementasikan static method **searchPerson** berikut. Method ini menerima *array of person* dan sebuah **id**, kemudian mengembalikan pasangan informasi **<nama, indeks>** dari orang yang dicari berdasarkan id-nya. **Indeks** merupakan indeks pada array dimana orang tersebut berada. Untuk proses pencarian, silakan gunakan algoritma **linear search**. Jika tidak ditemukan, kembalikan pasangan **<null, -1>**.

```
public static _____________searchPerson(Person[] people, int id) {



}
```

c). Modifikasi kode implementasi dari kelas Pair di atas sehingga dua buah nilai yang dikandung mempunyai **tipe data yang sama**.

d). **[lanjutan dari soal (c)]** tambahkan sebuah method yang bernama **swap()** kedalam kelas Pair yang baru Anda implementasikan di persoalan (c). Kelas swap() bertugas untuk **menukar** elemen pertama dengan elemen kedua pada sebuah **Pair**.

3. Generic Linear Search

Kita mengetahui implementasi algoritma linear search pada **array of integer** berikut:

```
public int linearSearch(int[] list, int target) {
    for (int i = 0; i < list.length; i++) {
        if (list[i] == target)
           return i;
    }
    return -1;
}
```

Modifikasi method **linearSearch** sehingga ia menjadi lebih generic ! dalam artian dapat menerima array dari berbagai macam jenis data.

4. Generic max2

Perhatikan interface **Measurable** berikut (seperti yang sudah dibahas di kelas sebelumnya)

```
public interface Measurable {
   public int getMeasure();
}
```

Implementasikan method **max2** yang menerima **2 buah object yang berasal dari tipe yang sama**, dan mengembalikan alamat object yang berisi nilai yang paling besar. Object tersebut harus merupakan instance dari class yang mengimplementasikan **Measurable**.

5. Generic findMin

Buatlah sebuah **generic static method** untuk mengembalikan **object yang bernilai paling minimal** dari sebuah **ArrayList** masukan. Elemen-elemen dari sebuah ArrayList merupakan elemen-elemen yang berasal dari **object yang dapat diukur (*measurable*)**. Nilai dari sebuah object diakses menggunakan **getMeasure()**.

```
public static ________________  __ findMin(ArrayList<___________> objects) {

     ___  smallest = objects.get(0);









     return smallest;
}
```

6. Generic Sort

Buatlah sebuah **generic static method** yang dapat melakukan pengurutan data apapun, selama data tersebut merupakan kumpulan Objek-objek yang dapat **diukur**. Method tersebut mengimplementasikan algoritma **selection sort**. Masukannya adalah sebuah Array.

```
public static ________________  void genericSort(________[]  arr) {









}
```