JOBSHEET VI

SORTING (BUBBLE, SELECTION, DAN INSERTION SORT)

6.5 Tujuan Praktikum

Setelah melakukan praktikum ini diharapkan mahasiswa mampu:

a. Mahasiswa mampu membuat algoritma sorting menggunakan bubble sort, selection sort dan insertion sort

b. Mahasiswa mampu menerapkan algoritma sorting menggunakan bubble sort, selection sort dan insertion sort pada program

6.6 Praktikum 1 - Mengimplementasikan Sorting menggunakan object

Waktu : 60 menit

6.2.1 Langkah Praktikum 1

a. SORTING – BUBBLE SORT

1. Buat folder baru bernama **Jobsheet6** di dalam repository **Praktikum ASD**
2. Buat class **Sorting**<No Presensi>, kemudian tambahkan atribut sebagai berikut:

```
public class Sorting {

    int [] data;
    int jumData;
}
```

3. Buatlah konstruktor dengan parameter Data[] dan jmlDat

```
Sorting (int Data[], int jmlDat){
    jumData=jmlDat;
    data=new int[jmlDat];
    for (int i=0; i<jumData; i++){
        data[i]= Data[i];
    }
}
```

4. Buatlah method **bubbleSort** bertipe void dan deklarasikan isinya menggunakan algoritma Bubble Sort.

```
void bubbleSort(){
    int temp=0;
    for (int i=0; i<jumData-1; i++){
        for (int j=1; j<jumData-i; j++){
            if (data[j-1]>data[j]){
                temp=data[j];
                data[j]=data[j-1];
                data[j-1]=temp;
            }
        }
    }
}
```

5. Buatlah method **tampil** bertipe void dan deklarasikan isi method tersebut.

```
void tampil(){
    for (int i=0; i<jumData;i++){
        System.out.print(data[i]+" ");
    }
    System.out.println();
}
```

6. Buat class **SortingMain**<No Presensi> kemudian deklarasikan array dengan nama a[] kemudian isi array tersebut

```
int a[]= {20, 10, 2, 7, 12};
```

7. Buatlah objek baru dengan nama **dataurut1** yang merupakan instansiasi dari class **Sorting**, kemudian isi parameternya

```
Sorting dataurut1 = new Sorting(a, a.length);
```

8. Lakukan pemanggilan method **bubbleSort** dan t**ampil**

```
System.out.println(x:"Data awal 1");
dataurut1.tampil();
dataurut1.bubbleSort();
System.out.println(x:"Data sudah diurutkan dengan BUBBLE SORT (ASC)");
dataurut1.tampil();
```

9. Jalankan program, dan amati hasilnya!

6.2.2 Verifikasi Hasil Percobaan

```
Data awal 1
20 10 2 7 12
Data sudah diurutkan dengan BUBBLE SORT (ASC)
2 7 10 12 20
```

b. SORTING – SELECTION SORT

1. Pada class **Sorting**<No Presensi> yang sudah dibuat di praktikum sebelumnya tambahkan method **SelectionSort** yang mengimplementasikan pengurutan menggunakan algoritma selection sort.

```
void SelectionSort(){
    for (int i=0; i<jumData-1; i++){
        int min=i;
        for (int j=i+1; j<jumData; j++){
            if(data[j]<data[min]){
                min=j;
            }
        }
        int temp=data[i];
        data[i]=data[min];
        data[min]=temp;
    }
}
```

2. Deklarasikan array dengan nama b[] pada kelas **SortingMain**<No Presensi> kemudian isi array tersebut

```
int b[]= {30, 20, 2, 8, 14};
```

3. Buatlah objek baru dengan nama **dataurut2** yang merupakan instansiasi dari class **Sorting**, kemudian isi parameternya

```
Sorting dataurut2 = new Sorting(b, b.length);
```

4. Lakukan pemanggilan method **SelectionSort** dan t**ampil**

```
System.out.println(x:"Data awal 2");
dataurut2.tampil();
dataurut2.SelectionSort();
System.out.println(x:"Data sudah diurutkan dengan SELECTION SORT (ASC)");
dataurut2.tampil();
```

5. Jalankan program dan amati hasilnya!

6.2.3 Verifikasi Hasil Percobaan

```
Data awal 2
30 20 2 8 14
Data sudah diurutkan dengan SELECTION SORT (ASC)
2 8 14 20 30
```

c. SORTING – INSERTION SORT

1. Pada class **Sorting**<No Presensi> yang sudah dibuat di praktikum sebelumnya tambahkan method **insertionSort** yang mengimplementasikan pengurutan menggunakan algoritma insertion sort.

```
void insertionSort(){
    for (int i=1; i<=data.length-1; i++){
        int temp=data[i];
        int j=i-1;
        while (j>=0 && data[j]>temp){
            data[j+1]=data[j];
            j--;
        }
        data[j+1]=temp;
    }
}
```

2. Deklarasikan array dengan nama c[] pada kelas **SortingMain**<No Presensi> kemudian isi array tersebut

```
int c[]= {40, 10, 4, 9, 3};
```

3. Buatlah objek baru dengan nama **dataurut3** yang merupakan instansiasi dari class **Sorting**, kemudian isi parameternya

```
Sorting dataurut3 = new Sorting(c, c.length);
```

4. Lakukan pemanggilan method **insertionSort** dan t**ampil**

```
System.out.println(x:"Data awal 3");
dataurut3.tampil();
dataurut3.insertionSort();
System.out.println(x:"Data sudah diurutkan dengan INSERTION SORT (ASC)");
dataurut3.tampil();
```

5. Jalankan program dan amati hasilnya!

6.2.4 Verifikasi Hasil Percobaan

```
Data awal 3
40 10 4 9 3
Data sudah diurutkan dengan INSERTION SORT (ASC)
3 4 9 10 40
```

6.2.5 Pertanyaan!

1. Jelaskan fungsi kode program berikut

```
if (data[j-1]>data[j]){
    temp=data[j];
    data[j]=data[j-1];
    data[j-1]=temp;
}
```

2. Tunjukkan kode program yang merupakan algoritma pencarian nilai minimum pada selection sort!
3. Pada Insertion sort , jelaskan maksud dari kondisi pada perulangan `while (j>=0 && data[j]>temp)`
4. Pada Insertion sort, apakah tujuan dari perintah `data[j+1]= data[j];`

6.7 Praktikum 2- (Sorting Menggunakan Array of Object)

Waktu : 45 menit

6.3.1 Langkah Praktikum 2 - Mengurutkan Data Mahasiswa Berdasarkan IPK (Bubble Sort)

Perhatikan diagram class Mahasiswa di bawah ini! Diagram class ini yang selanjutnya akan dibuat sebagai acuan dalam membuat kode program class Mahasiswa.

Mahasiswa
nim: String nama: String kelas: String ipk: double
Mahasiswa() Mahasiswa(nm: String, name: String, kls: String, ip: double) tampilInformasi(): void

Berdasarkan class diagram di atas, kita akan membuat sebuah class Mahasiswa yang berfungsi untuk membuat objek mahasiswa yang akan dimasukan ke dalam sebuah array. Terdapat sebuah konstruktor default dan berparameter dan juga fungsi tampil() untuk menampilkan semua attribute yang ada.

MahasiswaBerprestasi
listMhs: Mahasiswa[5] idx: int
tambah(mhs: Mahasiswa): void

tampil(): void bubbleSort(): void

Selanjutnya class diagram di atas merupakan representasi dari sebuah class yang berfungsi untuk melakukan operasi-operasi dari objek array mahasiswa, misalkan untuk menambahkan objek mahasiswa, menampilkan semua data mahasiswa, dan juga untuk mengurutkan menggunakan Teknik bubble sort berdasarkan nilai IPK mahasiswa.

6.3.2 Langkah-langkah Praktikum 2

1. Buatlah class dengan nama **Mahasiswa**<No Presensi>.
2. Untuk lebih jelasnya class tersebut dapat dilihat pada potongan kode di bawah ini

```
public class Mahasiswa {
    String nim;
    String nama;
    String kelas;
    double ipk;

    // Konstruktor default
    Mahasiswa() {
    }

    // Konstruktor berparameter (dibuat ada yang nama var parameter inputnya sama ada yang tidak)
    Mahasiswa(String nm, String name,  String kls, double ip) {
        nim = nm;
        nama = name;
        ipk = ip;
        kelas = kls;
    }

    void tampilInformasi() {
        System.out.println("Nama: " + nama);
        System.out.println("NIM: " + nim);
        System.out.println("Kelas: " + kelas);
        System.out.println("IPK: " + ipk);
    }
}
```

3. Buat class **MahasiswaBerprestasi**<No Presensi> seperti di bawah ini!

```
public class MahasiswaBerprestasi {
    Mahasiswa [] listMhs= new Mahasiswa [5];
    int idx;
}
```

4. Tambahkan method tambah() di dalam class tersebut! Method tambah() digunakan untuk menambahkan objek dari class Mahasiswa ke dalam atribut listMhs.

```
void tambah (Mahasiswa m){
    if (idx<listMhs.length){
        listMhs[idx]=m;
        idx++;
    }else {
        System.out.println(x:"data sudah penuh");
    }
}
```

5. Tambahkan method tampil() di dalam class tersebut! Method tampil() digunakan untuk menampilkan semua data mahasiswa-mahasiswa yang ada di dalam class tersebut! Perhatikan penggunaan sintaks for yang agak berbeda dengan for yang telah dipelajari sebelumnya, meskipun secara konsep sebenarnya mirip.

```
void tampil (){
    for (Mahasiswa m:listMhs){
        m.tampilInformasi();
        System.out.println(x:"------------------------------");
    }
}
```

6. Tambahkan method bubbleSort() di dalam class tersebut!

```
void bubbleSort(){
    for (int i=0; i<listMhs.length-1; i++){
        for (int j=1; j<listMhs.length-i; j++){
            if (listMhs[j].ipk>listMhs[j-1].ipk){
                Mahasiswa tmp = listMhs[j];
                listMhs[j]=listMhs[j-1];
                listMhs[j-1]=tmp;
            }
        }
    }
}
```

7. Buat class **MahasiswaDemo**<No Presensi>, kemudian buatlah sebuah objek MahasiswaBerprestasi dan buatlah 5 objek mahasiswa kemudian tambahkan semua objek mahasiswa tersebut dengan memanggil fungsi tambah pada objek MahasiswaBerprestasi. Silakan dipanggil fungsi tampil() untuk melihat semua data yang telah dimasukan, urutkan data tersebut dengan memanggil fungsi bubbleSort() dan yang terakhir panggil fungsi tampil kembali.

```
public static void main(String[] args) {
    MahasiswaBerprestasi list = new MahasiswaBerprestasi();
    Mahasiswa m1 = new Mahasiswa(nm:"123", name:"Zidan",kls:"2A", ip:3.2);
    Mahasiswa m2 = new Mahasiswa(nm:"124", name:"Ayu",kls:"2A", ip:3.5);
    Mahasiswa m3 = new Mahasiswa(nm:"125", name:"Sofi",kls:"2A", ip:3.1);
    Mahasiswa m4 = new Mahasiswa(nm:"126", name:"Sita",kls:"2A", ip:3.9);
    Mahasiswa m5 = new Mahasiswa(nm:"127", name:"Miki",kls:"2A", ip:3.7);

    list.tambah(m1);
    list.tambah(m2);
    list.tambah(m3);
    list.tambah(m4);
    list.tambah(m5);

    System.out.println(x:"Data mahasiswa sebelum sorting: ");
    list.tampil();

    System.out.println(x:"Data Mahasiswa setelah sorting berdasarkan IPK (DESC) : ");
    list.bubbleSort();
    list.tampil();
}
```

6.3.3 Verifikasi Hasil Percobaan

Cocokan hasilnya dengan yang terdapat pada tampilan di bawah ini

```
Data mahasiswa sebelum sorting:
Nama: Zidan
NIM: 123
IPK: 3.2
Kelas: 2A
----------------------------
Nama: Ayu
NIM: 124
IPK: 3.5
Kelas: 2A
----------------------------
Nama: Sofi
NIM: 125
IPK: 3.1
Kelas: 2A
----------------------------
Nama: Sita
NIM: 126
IPK: 3.9
Kelas: 2A
----------------------------
Nama: Miki
NIM: 127
IPK: 3.7
Kelas: 2A
----------------------------
```

```
Data Mahasiswa setelah sorting berdasarkan IPK (DESC) :
Nama: Sita
NIM: 126
IPK: 3.9
Kelas: 2A
----------------------------
Nama: Miki
NIM: 127
IPK: 3.7
Kelas: 2A
----------------------------
Nama: Ayu
NIM: 124
IPK: 3.5
Kelas: 2A
----------------------------
Nama: Zidan
NIM: 123
IPK: 3.2
Kelas: 2A
----------------------------
Nama: Sofi
NIM: 125
IPK: 3.1
Kelas: 2A
----------------------------
```

6.3.4 Pertanyaan

1. Perhatikan perulangan di dalam bubbleSort() di bawah ini:

```
for (int i=0; i<listMhs.length-1; i++){
    for (int j=1; j<listMhs.length-i; j++){
```

a. Mengapa syarat dari perulangan i adalah i<listMhs.length-1 ?

b. Mengapa syarat dari perulangan j adalah j<listMhs.length-i ?

c. Jika banyak data di dalam listMhs adalah 50, maka berapakali perulangan i akan berlangsung? Dan ada berapa **Tahap** bubble sort yang ditempuh?

2. Modifikasi program diatas dimana data mahasiswa bersifat dinamis (input dari keyborad) yang terdiri dari nim, nama, kelas, dan ipk!

6.4 Mengurutkan Data Mahasiswa Berdasarkan IPK (Selection Sort)

Waktu : 30 menit

Jika pada praktikum yang sebelumnya kita telah mengurutkan data mahasiswa berdasarkan IPK menggunakan Bubble Sort secara descending, pada kali ini kita akan mencoba untuk menambahkan fungsi pengurutan menggunakan Selection Sort.

6.4.1. Langkah-langkah Percobaan.

1. Lihat kembali class MahasiswaBerprestasi, dan tambahkan method selectionSort() di dalamnya! Method ini juga akan melakukan proses sorting secara **ascending**, tetapi menggunakan pendekatan selection sort.

```
void selectionSort(){
    for (int i=0; i<listMhs.length-1; i++){
        int idxMin=i;
        for (int j=i+1; j<listMhs.length; j++){
            if (listMhs[j].ipk<listMhs[idxMin].ipk){
                idxMin=j;
            }
        }
        Mahasiswa tmp = listMhs[idxMin];
        listMhs[idxMin]=listMhs[i];
        listMhs[i]=tmp;
    }
}
```

2. Setelah itu, buka kembali class MahasiswaDemo, dan di dalam method main() tambahkan baris program untuk memanggil method selectionSort() tersebut, kemudian panggil method tampil() untuk menampilkan data yang sudah diurutkan!

```
System.out.println(x:"Data yang sudah terurut menggunakan SELECTION SORT (ASC)");
list.selectionSort();
list.tampil();
```

3. Coba jalankan kembali class MahasiswaDemo, dan amati hasilnya! Apakah kini data mahasiswa telah tampil urut menaik berdasar ipk?

6.4.2 Verifikasi Hasil Percobaan

Pastikan output yang ditampilkan sudah benar seperti di bawah ini

```
Masukkan Data Mahasiswa ke-1
NIM   : 123
Nama  : Ali
Kelas : 2B
IPK   : 3.9
------------------------------
Masukkan Data Mahasiswa ke-2
NIM   : 124
Nama  : ila
Kelas : 2B
IPK   : 3.1
------------------------------
```

```
Masukkan Data Mahasiswa ke-3
NIM   : 125
Nama  : agus
Kelas : 2B
IPK   : 3.6
------------------------------
Masukkan Data Mahasiswa ke-4
NIM   : 126
Nama  : tika
Kelas : 2B
IPK   : 3.3
------------------------------
Masukkan Data Mahasiswa ke-5
NIM   : 127
Nama  : udin
Kelas : 2B
IPK   : 3.2
------------------------------
Data yang sudah terurut menggunakan SELECTION SORT (ASC)
Nama: ila
NIM: 124
Kelas: 2B
IPK: 3.1
----------------------------
Nama: udin
NIM: 127
Kelas: 2B
IPK: 3.2
----------------------------
Nama: tika
NIM: 126
Kelas: 2B
IPK: 3.3
----------------------------
Nama: agus
NIM: 125
Kelas: 2B
IPK: 3.6
----------------------------
Nama: Ali
NIM: 123
Kelas: 2B
IPK: 3.9
----------------------------
```

6.4.3 Pertanyaan

Di dalam method selection sort, terdapat baris program seperti di bawah ini:

```
int idxMin=i;
for (int j=i+1; j<listMhs.length; j++){
    if (listMhs[j].ipk<listMhs[idxMin].ipk){
        idxMin=j;
    }
}
```

Untuk apakah proses tersebut, jelaskan!

6.5 Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan Insertion Sort

Waktu : 30 menit

Yang terakhir akan diimplementasikan Teknik sorting menggunakan Insertion Sort, dengan mengurutkan IPK mahasiswa secara ascending.

6.5.1 Langkah-langkah Percobaan

1. Lihat kembali class MahasiswaBerprestasi, dan tambahkan method insertionSort() di dalamnya. Method ini juga akan melakukan proses sorting secara ***ascending***, tetapi menggunakan pendekatan Insertion Sort.

```
void insertionSort(){
    for (int i=1; i<listMhs.length; i++){
        Mahasiswa temp = listMhs[i];
        int j=i;
        while (j>0 && listMhs[j-1].ipk>temp.ipk){
            listMhs[j]=listMhs[j-1];
            j--;
        }
        listMhs[j]=temp;
    }
}
```

2. Setelah itu, buka kembali class MahasiswaDemo, dan di dalam method main() tambahkan baris program untuk memanggil method insertionSort() dan tampil () tersebut!

```
System.out.println(x:"Data yang sudah terurut menggunakan INSERTION SORT (ASC)");
list.insertionSort();
list.tampil();
```

3. Coba jalankan kembali class MahasiswaDemo, dan amati hasilnya! Apakah kini data mahasiswa telah tampil urut menaik berdasar ipk?

6.5.2 Verifikasi Hasil Percobaan

Pastikan output yang ditampilkan sudah benar seperti di bawah ini

```
Masukkan Data Mahasiswa ke-1
NIM   : 111
Nama  : ayu
Kelas : 2c
IPK   : 3.7
------------------------------
Masukkan Data Mahasiswa ke-2
NIM   : 222
Nama  : dika
Kelas : 2c
IPK   : 3.0
------------------------------
Masukkan Data Mahasiswa ke-3
NIM   : 333
Nama  : ila
Kelas : 2c
IPK   : 3.8
------------------------------
Masukkan Data Mahasiswa ke-4
NIM   : 444
Nama  : susi
Kelas : 2c
IPK   : 3.1
------------------------------
Masukkan Data Mahasiswa ke-5
NIM   : 555
Nama  : yayuk
Kelas : 2c
IPK   : 3.4
------------------------------
```

```
Data yang sudah terurut menggunakan INSERTION SORT (ASC)
Nama: dika
NIM: 222
Kelas: 2c
IPK: 3.0
-----------------------------
Nama: susi
NIM: 444
Kelas: 2c
IPK: 3.1
-----------------------------
Nama: yayuk
NIM: 555
Kelas: 2c
IPK: 3.4
-----------------------------
Nama: ayu
NIM: 111
Kelas: 2c
IPK: 3.7
-----------------------------
Nama: ila
NIM: 333
Kelas: 2c
IPK: 3.8
-----------------------------
```

6.5.3 Pertanyaan

Ubahlah fungsi pada InsertionSort sehingga fungsi ini dapat melaksanakan proses sorting dengan cara descending.

6.6 Latihan Praktikum

Waktu : 45 Menit

Perhatikan class diagram dibawah ini:

Dosen
kode: String nama: String jenisKelamin: Boolean usia: int
Dosen(kd: String, name: String, jk: Boolean, age: int) tampil(): void

DataDosen
dataDosen: Dosen[10] idx: int
tambah(dsn: Dosen): void tampil(): void SortingASC(): void sortingDSC():void insertionSort():void

Berdasarkan class diagram diatas buatlah menu dikelas main dengan pilihan menu:

1. Tambah data digunakan untuk menambahkan data dosen
2. Tampil data digunakan untuk menampilkan data seluruh dosen
3. Sorting ASC digunakan untuk mengurutkan data dosen berdasarkan usia dimulai dari dosen termuda ke dosen tertua menggunakan bublle Sort.
4. Sorting DSC digunakan untuk mengurutkan data dosen berdasarkan usia dimulai dari tertua ke dosen termuda dapat menggunakan algoritma selection sort atau insertion sort.