JOBSHEET - 5

SORTING (BUBBLE, SELECTION, DAN INSERTION SORT)

5.1 Tujuan Praktikum

Setelah melakukan praktikum ini diharapkan mahasiswa mampu:

a. Mahasiswa mampu membuat algoritma searching bubble sort, selection sort dan insertion sort
b. Mahasiswa mampu menerapkan algoritma searching bubble sort, selection sort dan insertion sort pada program

5.2 Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan 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
nama: String thnMasuk: int umur: int ipk: double
Mahasiswa(n: String, t: int, u: int, i: double) tampil(): void

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

DaftarMahasiswaBerprestasi
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.

5.2.1 Langkah-langkah Percobaan

1. Buat project baru dengan nama “bubble-selection-insertion”, kemudian buat package dengan nama “jobsheet6”.
2. Buatlah sebuah class dengan nama Mahasiswa
3. Sesuaikan class Mahasiswa dengan melihat class diagram di atas dengan menambahkan attribute, konstruktor, dan fungsi atau method. Untuk lebih jelasnya class tersebut dapat dilihat pada potongan kode di bawah ini

```
1   package minggu5;
2
3   public class Mahasiswa {
4       String nama;
5       int thnMasuk, umur;
6       double ipk;
7
8       Mahasiswa(String n, int t, int u, double i){
9           nama = n;
10          thnMasuk = t;
11          umur = u;
12          ipk = i;
13      }
14
15      void tampil(){
16          System.out.println("Nama = "+nama);
17          System.out.println("Tahun Masuk = "+thnMasuk);
18          System.out.println("Umur = "+umur);
19          System.out.println("IPK = "+ipk);
20      }
21  }
```

4. Buat class DaftarMahasiswaBerprestasi seperti di bawah ini!

```
1   package minggu5;
2
3   public class DafraMahasiswaBerprestasi {
4       Mahasiswa listMhs[] = new Mahasiswa[5];
5       int idx;
6
7       //setelah ini tuliskan method tambah()
8
9       //setelah ini tuliskan method tampil()
10
11      //setelah ini tuliskan method bubbleSort()
12  }
```

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

```
7       //setelah ini tuliskan method tambah()
8       void tambah(Mahasiswa m){
9           if(idx<listMhs.length){
10              listMhs[idx] = m;
11              idx++;
12          }else{
13              System.out.println("Data sudah penuh!!");
14          }
15      }
```

6. 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.

```
17      //setelah ini tuliskan method tampil()
18      void tampil(){
19          for(Mahasiswa m : listMhs){
20              m.tampil();
21              System.out.println("------------------------");
22          }
23      }
```

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

```
25      //setelah ini tuliskan method bubbleSort()
26      void bubbleSort(){
27          for(int i=0; i<listMhs.length-1; i++){
28              for(int j=1; j<listMhs.length-i; j++){
29                  if(listMhs[j].ipk > listMhs[j-1].ipk){
30                      //di bawah ini proses swap atau penukaran
31                      Mahasiswa tmp = listMhs[j];
32                      listMhs[j] = listMhs[j-1];
33                      listMhs[j-1] = tmp;
34                  }
35              }
36          }
37      }
```

8. Buat class Main dan didalamnya buat method main() seperti di bawah ini!

```
1   package minggu5;
2   import java.util.Scanner;
3
4   public class Main {
5       public static void main(String[] args) {
6
7       }
8   }
```

9. **Di dalam method main()**, buatlah sebuah objek DaftarMahasiswaBerprestasi dan buatlah 5 objek mahasiswa kemudian tambahkan semua objek mahasiswa tersebut dengan memanggil fungsi tambah pada objek DaftarMahasiswaBerprestasi. 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.

```
DaftarMahasiswaBerprestasi list = new DaftarMahasiswaBerprestasi();
Mahasiswa m1 = new Mahasiswa("Nusa", 2017, 25, 3);
Mahasiswa m2 = new Mahasiswa("Rara", 2012, 19, 4);
Mahasiswa m3 = new Mahasiswa("Dompu", 2018, 19, 3.5);
Mahasiswa m4 = new Mahasiswa("Abdul", 2017, 23, 2);
Mahasiswa m5 = new Mahasiswa("Ummi", 2019, 21, 3.75);

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

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

System.out.println("Data mahasiswa setelah sorting desc berdasarkan ipk");
list.bubbleSort();
list.tampil();
```

5.2.2 Verifikasi Hasil Percobaan

Cocokan hasilnya dengan yang terdapat pada tampilan di bawah ini

```
Data mahasiswa sebelum sorting =
Nama        = Nusa
Tahun Masuk = 2017
Umur        = 25
IPK         = 3.0
-----------------------
Nama        = Rara
Tahun Masuk = 2012
Umur        = 19
IPK         = 4.0
-----------------------
Nama        = Dompu
Tahun Masuk = 2018
Umur        = 19
IPK         = 3.5
-----------------------
Nama        = Abdul
Tahun Masuk = 2017
Umur        = 23
IPK         = 2.0
-----------------------
Nama        = Ummi
Tahun Masuk = 2019
Umur        = 21
IPK         = 3.75
-----------------------
```

```
Data mahasiswa setelah sorting desc berdasarkan ipk
Nama        = Rara
Tahun Masuk = 2012
Umur        = 19
IPK         = 4.0
-----------------------
Nama        = Ummi
Tahun Masuk = 2019
Umur        = 21
IPK         = 3.75
-----------------------
Nama        = Dompu
Tahun Masuk = 2018
Umur        = 19
IPK         = 3.5
-----------------------
Nama        = Nusa
Tahun Masuk = 2017
Umur        = 25
IPK         = 3.0
-----------------------
Nama        = Abdul
Tahun Masuk = 2017
Umur        = 23
IPK         = 2.0
-----------------------
```

5.2.3 Pertanyaan

1. Terdapat di method apakah proses bubble sort?

2. Terdapat di method apakah proses selection sort?

3. Apakah yang dimaksud proses swap? Tuliskan potongan program untuk melakukan proses swap tersebut!

4. Di dalam method bubbleSort(), terdapat baris program seperti di bawah ini:

```
29                  if(listMhs[j].ipk > listMhs[j-1].ipk){
30                      //di bawah ini proses swap atau penukaran
31                      Mahasiswa tmp = listMhs[j];
32                      listMhs[j] = listMhs[j-1];
33                      listMhs[j-1] = tmp;
34                  }
35              }
```

Untuk apakah proses tersebut?

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

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

a. Apakah perbedaan antara kegunaan perulangan i dan perulangan j?

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

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

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

5.3 Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan Selection Sort

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.

5.3.1. Langkah-langkah Percobaan.

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

```
39      //setelah ini tuliskan method selectionSort()
40      void selectionSort(){
41          for(int i=0; i<listMhs.length-1; i++){
42              int idxMin = i;
43              for(int j=i+1; j<listMhs.length; j++){
44                  if(listMhs[j].ipk < listMhs[idxMin].ipk){
45                      idxMin = j;
46                  }
47              }
48              //swap
49              Mahasiswa tmp = listMhs[idxMin];
50              listMhs[idxMin] = listMhs[i];
51              listMhs[i] = tmp;
52          }
53      }
```

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

```
System.out.println("Data mahasiswa setelah sorting asc berdasarkan ipk");
list.selectionSort();
list.tampil();
```

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

5.3.2. Verifikasi Hasil Percobaan

Pastikan output yang ditampilkan sudah benar seperti di bawah ini

```
Data mahasiswa sebelum sorting =
Nama         = Nusa
Tahun Masuk = 2017
Umur         = 25
IPK          = 3.0
-----------------------
Nama         = Rara
Tahun Masuk = 2012
Umur         = 19
IPK          = 4.0
-----------------------
Nama         = Dompu
Tahun Masuk = 2018
Umur         = 19
IPK          = 3.5
-----------------------
Nama         = Abdul
Tahun Masuk = 2017
Umur         = 23
IPK          = 2.0
-----------------------
Nama         = Ummi
Tahun Masuk = 2019
Umur         = 21
IPK          = 3.75
-----------------------
```

```
Data mahasiswa setelah sorting asc berdasarkan ipk
Nama         = Abdul
Tahun Masuk = 2017
Umur         = 23
IPK          = 2.0
-----------------------
Nama         = Nusa
Tahun Masuk = 2017
Umur         = 25
IPK          = 3.0
-----------------------
Nama         = Dompu
Tahun Masuk = 2018
Umur         = 19
IPK          = 3.5
-----------------------
Nama         = Ummi
Tahun Masuk = 2019
Umur         = 21
IPK          = 3.75
-----------------------
Nama         = Rara
Tahun Masuk = 2012
Umur         = 19
IPK          = 4.0
-----------------------
```

5.3.3. Pertanyaan

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

```
42              int idxMin = i;
43              for(int j=i+1; j<listMhs.length; j++){
44                  if(listMhs[j].ipk < listMhs[idxMin].ipk){
45                      idxMin = j;
46                  }
47              }
```

Untuk apakah proses tersebut, jelaskan!

5.4 Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan Insertion Sort

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

5.4.1 Langkah-langkah Percobaan

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

```
68      void insertionSort() {
69          for (int i = 1; i < listMhs.length; i++) {
70              Mahasiswa temp = listMhs[i];
71              int j = i;
72              while (j > 0 && listMhs[j - 1].ipk > temp.ipk) {
73                  listMhs[j] = listMhs[j - 1];
74                  j--;
75              }
76              listMhs[j] = temp;
77          }
78      }
```

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

```
System.out.println("Data mahasiswa setelah sorting asc berdasarkan ipk");
list.insertionSort();
list.tampil();
```

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

5.4.2 Verifikasi Hasil Percobaan

Pastikan output yang ditampilkan sudah benar seperti di bawah ini

```
Data mahasiswa sebelum sorting =
Nama        = Nusa
Tahun Masuk = 2017
Umur        = 25
IPK         = 3.0
----------------------
Nama        = Rara
Tahun Masuk = 2012
Umur        = 19
IPK         = 4.0
----------------------
Nama        = Dompu
Tahun Masuk = 2018
Umur        = 19
IPK         = 3.5
----------------------
Nama        = Abdul
Tahun Masuk = 2017
Umur        = 23
IPK         = 2.0
----------------------
Nama        = Ummi
Tahun Masuk = 2019
Umur        = 21
IPK         = 3.75
----------------------
```

```
Data mahasiswa setelah sorting asc berdasarkan ipk
Nama        = Abdul
Tahun Masuk = 2017
Umur        = 23
IPK         = 2.0
----------------------
Nama        = Nusa
Tahun Masuk = 2017
Umur        = 25
IPK         = 3.0
----------------------
Nama        = Dompu
Tahun Masuk = 2018
Umur        = 19
IPK         = 3.5
----------------------
Nama        = Ummi
Tahun Masuk = 2019
Umur        = 21
IPK         = 3.75
----------------------
Nama        = Rara
Tahun Masuk = 2012
Umur        = 19
IPK         = 4.0
----------------------
```

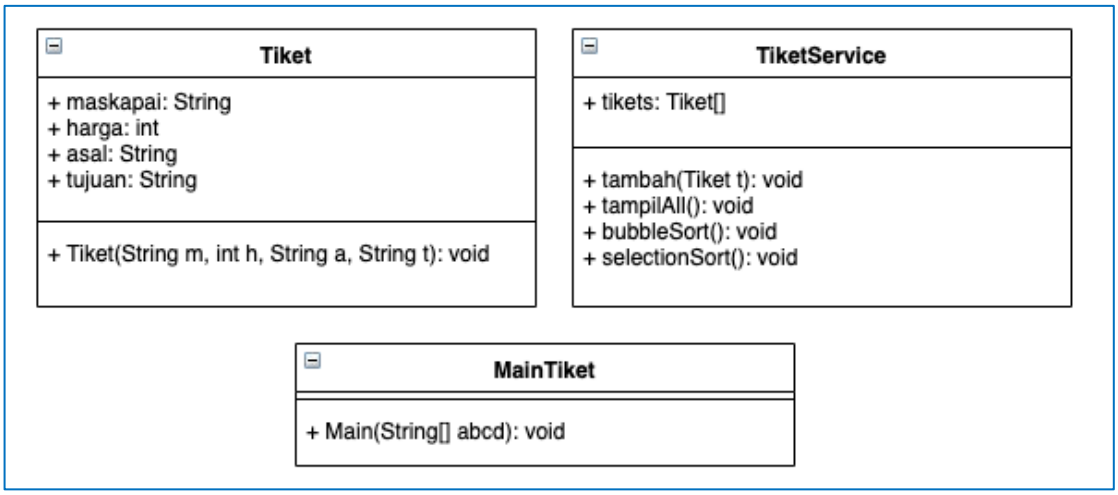
5.4.3 Pertanyaan

Ubahlah fungsi pada InsertionSort sehingga fungsi ini dapat melaksanakan proses sorting dengan cara ascending atau decending, anda dapat melakukannya dengan menambahkan parameter pada pemanggilan fungsi insertionSort.

```
80      void insertionSort(boolean asc) {
81          for (int i = 1; i < listMhs.length; i++) {
82              Mahasiswa temp = listMhs[i];
83              int j = i;
84              if (asc) {
85
86              } else {
87
88              }
89              listMhs[j] = temp;
90          }
91      }
```

5.5 Latihan Praktikum

1. Sebuah yang bergerak dalam bidang penjualan tiket pesawat sedang mengembangkan backend untuk sistem pemesanan tiket, salah satu fiturnya adalah menampilkan daftar tiket yang tersedia berdasarkan pilihan filter yang diinginkan user. Daftar tiket ini harus dapat di sorting berdasarkan harga dimulai dari harga termurah ke harga tertinggi. Implementasikanlah class diagram berikut ini kedalam bahasa pemrograman java kemudian buatlah proses sorting data untuk harga tiket menggunakan algoritma **bubble sort** dan **selection sort**.



2. Liga Inggris pada tahun 2020 sudah berjalan setengah musim, pada musim ini Liverpool merajai puncak kelasemen dengan perbedaan yang sangat mencolok hal ini dapat dilihat pada tabel kelasemen dibawah ini :

	Premier League	P	GD	PTS
1	Liverpool	29	45	82
2	Manchester City	27	39	57
3	Leicester	28	26	50
4	Chelsea	29	9	48
5	Wolverhampton Wanderers	29	7	43
6	Sheffield United	28	5	43
7	Manchester United	28	12	42
8	Tottenham Hotspur	29	7	41
9	Arsenal	28	4	40
10	Burnley	29	-6	39
11	Crystal Palace	29	-6	39
12	Everton	29	-6	37
13	Newcastle United	29	-16	35
14	Southampton	29	-17	34
15	Brighton & Hove Albion	29	-8	29
16	West Ham United	29	-15	27
17	Watford	29	-17	27
18	AFC Bournemouth	29	-18	27
19	Aston Villa	27	-18	25
20	Norwich City	29	-27	21

Manchester City

Ubahlah data kelasemen diatas menjadi sebuah class diagram yang memiliki fungsi sorting klub berdasarkan jumlah poin dari yang terbesar ke yang terkecil (descending) dan fungsi sorting klub berdasarkan jumlah poin dari yang terkecil ke yang terbesar (ascending) menggunakan algoritma insertion sort. Untuk memudahkan dibuatkan class diagram seperti pada gambar berikut ini