JOBSHEET - 5

SORTING (BUBBLE, SELECTION, DAN INSERTION SORT)

Nama: Rizqi Bagus Andrean

Kelas: TI-1D

Absen: 25

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

Waktu : 50 menit

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?
Method bubbleSort.

2. 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?

Bagian kode ini menukar elemen dalam bubble sort. Membandingkan IPK elemen terdekat (listMMhs[j] dan listMMhs[3j-1]). Jika IPK elemen sekarang lebih besar, maka ditukar menggunakan variabel temporary temp. Ini memastikan elemen terurut sesuai IPK (naik dalam kasus ini).

3. 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?

Perulangan i dan j bekerja sama untuk mengurutkan elemen dalam listMMhs berdasarkan IPK. Perulangan i mengontrol berapa kali perbandingan dan pertukaran dilakukan, sedangkan perulangan j melakukan perbandingan dan

pertukaran tersebut.

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

Syarat i < listMMhs.length - 1 memastikan perulangan i berhenti sebelum mencapai batas array dan elemen terakhir diurutkan dengan benar.

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

Syarat `j < listMMhs.length - i` memastikan perulangan `j` hanya membandingkan elemen yang belum terurut dan meningkatkan efisiensi bubble sort

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

Perulangan `i` dalam bubble sort dengan 50 data akan berlangsung sebanyak **49** kali

5.3 Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan Selection Sort

Waktu : 30 menit

Jika pada praktikum yang sebelumnya kita telah mengurutkan data mahasiwa 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!

Bagian kode itu mencari indeks mahasiswa dengan IPK terkecil dalam sub-list yang belum terurut. Looping `j` iterasi di seluruh sub-list dan membandingkan IPK. Jika ditemukan IPK yang lebih kecil, indeks minimum (`idxMin`) diperbarui.
Bagian kode itu mencari indeks mahasiswa dengan IPK terkecil dalam sub-list yang belum terurut. Looping `j` iterasi di seluruh sub-list dan membandingkan IPK. Jika ditemukan IPK yang lebih kecil, indeks minimum (`idxMin`) diperbarui.

5.4 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.

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 descending.

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

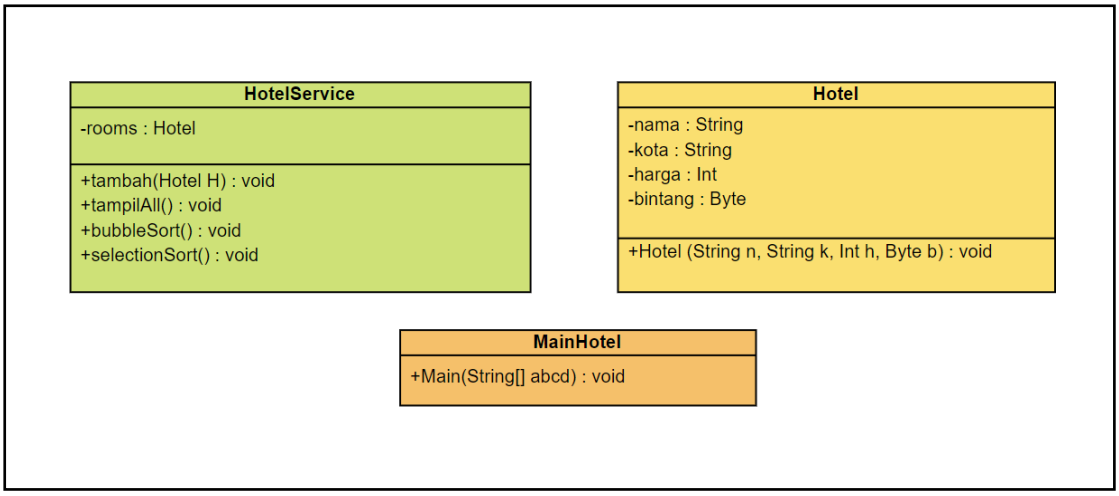
5.5 Latihan Praktikum

Waktu : 90 Menit

Sebuah platform travel yang menyediakan layanan pemesanan kebutuhan travelling sedang mengembangkan backend untuk sistem pemesanan/reservasi akomodasi (penginapan), salah satu fiturnya adalah menampilkan daftar penginapan yang tersedia berdasarkan pilihan filter yang diinginkan user. Daftar penginapan ini harus dapat disorting berdasarkan

1. Harga dimulai dari harga termurah ke harga tertinggi.
2. Rating bintang penginapan dari bintang tertinggi (5) ke terendah (1)

Buatlah proses sorting data untuk kedua filter tersebut dengan menggunakan algoritma **bubble sort** dan **selection sort**.



```java
package travel;

public class HotelService {
    Hotel[] rooms = new Hotel[10];

    void tambah(Hotel h) {
        boolean isFull = false;
     for (int i = 0; i < rooms.length; i++) {
        isFull = rooms[i] != null;
         if (rooms[i] == null) {
             rooms[i] = h;
             break;
         }else{
                isFull = true;
         }
     }

        if (isFull) {
            System.out.println("Data sudah penuh!");
        }
    }

    void tampilAll() {
        for (Hotel h : rooms) {
            if (h != null) {
                System.out.println("Nama: " + h.nama);
                System.out.println("Kota: " + h.kota);
                System.out.println("Harga: " + h.harga);
                System.out.println("Bintang: " + Double.toString(h.bintang));
                System.out.println("--------------------");
            }
        }
    }

    void selectionSort(String filterBy) {
        if (filterBy == "harga") {
            for (int i = 0; i < rooms.length - 1; i++) {
                int idxMin = i;
                for (int j = i + 1; j < rooms.length; j++) {
                    if (rooms[j].harga < rooms[idxMin].harga) {
                        idxMin = j;
                    }
                }
                Hotel temp = rooms[idxMin];
                rooms[idxMin] = rooms[i];
                rooms[i] = temp;
            }
        } else if (filterBy == "bintang") {
            for (int i = 0; i < rooms.length - 1; i++) {
                int idxMin = i;
                for (int j = i + 1; j < rooms.length; j++) {
                    if (rooms[j].bintang < rooms[idxMin].bintang) {
                        idxMin = j;
                    }
                }
            }
        }
    }

```

```
59
60      void bubbleSort(String filterBy) {
61          if (filterBy == "harga") {
62              for (int i = 0; i < rooms.length - 1; i++) {
63                  for (int j = 1; j < rooms.length - i; j++) {
64                      if (rooms[j].harga < rooms[j - 1].harga) {
65                          Hotel temp = rooms[j];
66                          rooms[j] = rooms[j - 1];
67                          rooms[j - 1] = temp;
68                      }
69                  }
70              }
71          } else if (filterBy == "bintang") {
72              for (int i = 0; i < rooms.length - 1; i++) {
73                  for (int j = 1; j < rooms.length - i; j++) {
74                      if (rooms[j].bintang > rooms[j - 1].bintang) {
75                          Hotel temp = rooms[j];
76                          rooms[j] = rooms[j - 1];
77                          rooms[j - 1] = temp;
78                      }
79                  }
80              }
81          }
82      }
83
84
85      public static void main(String[] args) {
86
87          HotelService hs = new HotelService();
88
89          hs.tambah(new Hotel("Hotel B", "Jakarta", 2000000, 4.0));
90          hs.tambah(new Hotel("Hotel C", "Jakarta", 3000000, 3.5));
91          hs.tambah(new Hotel("Hotel F", "Jakarta", 6000000, 2.0));
92          hs.tambah(new Hotel("Hotel D", "Jakarta", 4000000, 3.0));
93          hs.tambah(new Hotel("Hotel A", "Jakarta", 1000000, 4.5));
94          hs.tambah(new Hotel("Hotel J", "Jakarta", 10000000, 0.0));
95          hs.tambah(new Hotel("Hotel E", "Jakarta", 5000000, 2.5));
96          hs.tambah(new Hotel("Hotel G", "Jakarta", 7000000, 1.5));
97          hs.tambah(new Hotel("Hotel I", "Jakarta", 9000000, 0.5));
98          hs.tambah(new Hotel("Hotel H", "Jakarta", 8000000, 1.0));
99
100         hs.tampilAll();
101
102         System.out.println("Setelah diurutkan by harga");
103
104         hs.selectionSort("harga");
105         hs.tampilAll();
106
107         System.out.println("Setelah diurutkan by bintang");
108
109         hs.selectionSort("bintang");
110         hs.tampilAll();
111
112
113     }
114 }
115
```