

LAPORAN PRAKTIKUM JOBSHEET 5
SORTING (BUBBLE, SELECTION, DAN INSERTION SORT)
MATA KULIAH ALGORITMA DAN STRUKTUR DATA



Disusun Oleh :
Jami'atul Afifah (2341760102)
SIB-1F

PROGRAM STUDI D4 SISTEM INFOEMASI BISNIS
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG
2024

Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan Bubble Sort

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     public Mahasiswa(String n, int t, int u, double i) {  
9         nama = n;  
10        thnMasuk = t;  
11        umur = u;  
12        ipk = i;  
13    }  
14  
15    public 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 }  
22
```

4. Buat class DaftarMahasiswaBerprestasi seperti di bawah ini!
5. Tambahkan method tambah() di dalam class tersebut! Method tambah() digunakan untuk menambahkan objek dari class Mahasiswa ke dalam atribut listMhs.
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.
7. Tambahkan method bubbleSort() di dalam class tersebut!

```

package minggu5;

public class DaftarMahasiswaBerprestasi {
    Mahasiswa listMhs[] = new Mahasiswa[5];
    int idx;

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

    public void tampil() {
        for (Mahasiswa m : listMhs) {
            if (m != null) {
                m.tampil();
                System.out.println(x:"-----");
            }
        }
    }

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

```

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

```

package minggu8;

public class MainBubble {
    Run | Debug
    public static void main(String[] args) {
        DaftarMahasiswaBerprestasi list = new DaftarMahasiswaBerprestasi();
        Mahasiswa m1 = new Mahasiswa(n:"Nusa", t:2017, u:25, i:3);
        Mahasiswa m2 = new Mahasiswa(n:"Rara", t:2012, u:19, i:4);
        Mahasiswa m3 = new Mahasiswa(n:"Dompur", t:2018, u:19, i:3.5);
        Mahasiswa m4 = new Mahasiswa(n:"Abdul", t:2017, u:23, i:2);
        Mahasiswa m5 = new Mahasiswa(n:"Ummi", t:2019, u:21, i:3.75);

        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 desc berdasarkan ipk");
        list.bubbleSort();
        list.tampil();
    }
}

```

Verifikasi Hasil Percobaan

Cocokan hasilnya dengan yang terdapat pada tampilan di bawah ini

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

 Run: MainB

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

Pertanyaan

1. Terdapat di method apakah proses bubble sort?

Proses bubble sort terdapat di dalam method bubbleSort() di file DaftarMahasiswaBerprestasi.java.

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?

Proses tersebut adalah bagian dari algoritma bubble sort yang bertanggung jawab untuk menukar posisi dua elemen dalam array jika kondisi tertentu terpenuhi, yaitu jika elemen ke-j memiliki IPK yang lebih besar dari elemen ke-(j-1). Ini dilakukan untuk mengurutkan elemen dari yang terkecil ke yang terbesar.

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?

Perbedaan antara penggunaan perulangan i dan j adalah bahwa perulangan i digunakan untuk mengontrol iterasi melalui seluruh array, sementara perulangan j digunakan untuk membandingkan dan menukar elemen-elemen terkait dalam setiap iterasi i.

- b. Mengapa syarat dari perulangan i adalah $i < \text{listMhs.length} - 1$?

Syarat dari perulangan i adalah $i < \text{listMhs.length} - 1$ karena setelah setiap iterasi i, elemen terakhir sudah pasti berada di posisi yang benar setelah berjalannya algoritma bubble sort. Oleh karena itu, tidak perlu membandingkan elemen terakhir di setiap iterasi.

- c. Mengapa syarat dari perulangan j adalah $j < \text{listMhs.length} - i$?

Syarat dari perulangan j adalah $j < \text{listMhs.length} - i$ karena setiap iterasi i akan mengurangi jumlah elemen yang perlu dibandingkan di setiap iterasi j. Pada awalnya, perulangan j akan membandingkan semua elemen di array, tetapi setiap kali iterasi i berjalan, elemen terakhir dalam array sudah pasti berada di posisi yang benar, sehingga tidak perlu dibandingkan lagi.

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

Jika banyak data di dalam listMhs adalah 50, maka perulangan i akan berlangsung sebanyak 49 kali. Ini karena setiap iterasi i mengurangi jumlah elemen yang harus dibandingkan. Tahap bubble sort yang ditempuh adalah

sebanyak 49 tahap, karena setiap iterasi i memindahkan setidaknya satu elemen ke posisi yang benar.

Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan Selection Sort

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

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

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

```
System.out.println(x:"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?
Ya hasilnya urut sesuai ipk dari terbesar ke terkecil

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 = Dampu
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 = Dampu
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
-----
```


Pertanyaan

1. 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!

- a. Inisialisasi Variabel idxMin: Pada awal setiap iterasi, variabel idxMin diinisialisasi dengan nilai i. Ini dilakukan untuk menandai indeks dari elemen yang memiliki nilai minimum yang terpilih saat ini dalam iterasi tersebut.
- b. Perbandingan dengan Elemen Berikutnya: Selanjutnya, dalam loop for kedua, kita membandingkan elemen pada indeks j dengan elemen pada indeks idxMin. Jika nilai ipk dari elemen pada indeks j lebih kecil dari nilai ipk dari elemen pada indeks idxMin, maka kita perbarui nilai idxMin dengan nilai j.
- c. Pembaruan idxMin: Jika ditemukan elemen dengan nilai ipk yang lebih kecil dari elemen yang sebelumnya dianggap sebagai nilai minimum (idxMin), maka idxMin akan diperbarui dengan indeks yang menunjuk ke elemen tersebut. Ini dilakukan untuk memastikan bahwa pada akhir iterasi, idxMin akan menunjuk ke indeks yang berisi nilai minimum dari seluruh elemen yang belum diurutkan.
- d. Dengan cara ini, langkah-langkah ini terus diulangi untuk setiap elemen dalam array sampai seluruh array terurut, yaitu dengan menempatkan elemen terkecil pada posisi yang benar secara berurutan. Setelah selesai iterasi, elemen dengan nilai minimum akan dipindahkan ke posisi yang tepat pada iterasi tersebut.

Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan Insertion Sort

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

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

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

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

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

Verifikasi Hasil Percobaan

Pastikan output yang ditampilkan sudah benar seperti di bawah ini

```
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 = Dompus  
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  
-----
```

Pertanyaan

Ubahlah fungsi pada InsertionSort sehingga fungsi ini dapat melaksanakan proses sorting

dengan cara descending

tambahkan kode :

// Di dalam kelas DaftarMahasiswaBerprestasi

```
void insertionSort() {
```

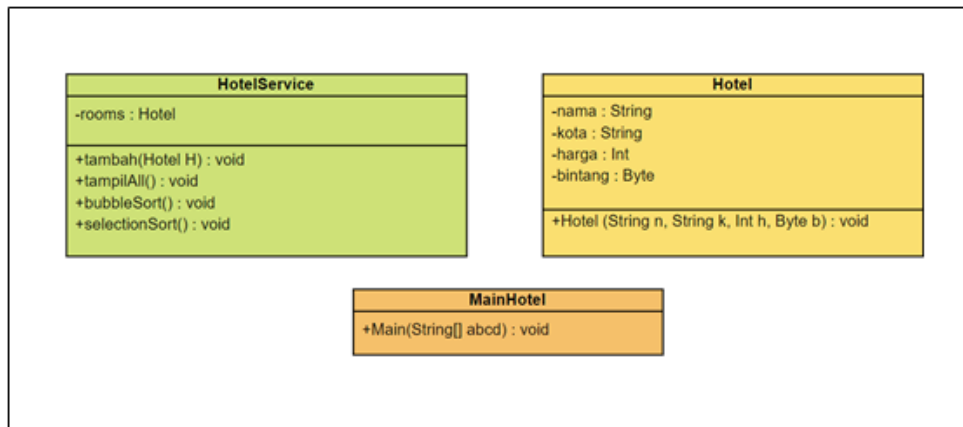
```
for (int i = 1; i < idx; i++) {  
    Mahasiswa key = listMhs[i];  
    int j = i - 1;  
  
    // Geser elemen yang lebih besar dari key ke kanan  
    while (j >= 0 && listMhs[j].ipk < key.ipk) {  
        listMhs[j + 1] = listMhs[j];  
        j--;  
    }  
    listMhs[j + 1] = key;  
}  
}
```

Latihan Praktikum

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.



Hotel

```
jobsheet6 > J Hotel14.java > ...
1 // Hotel14.java
2 package jobsheet6;
3
4 public class Hotel14 {
5     String nama;
6     String kota;
7     int harga;
8     byte bintang;
9
10    Hotel14(String n, String k, int h, byte b){
11        this.nama = n;
12        this.kota = k;
13        this.harga = h;
14        this.bintang = b;
15    }
16
17    void tampilAll(){
18        System.out.println("Nama = " + nama);
19        System.out.println("Kota = " + kota);
20        System.out.println("Harga = " + harga);
21        System.out.println("Bintang = " + bintang);
22    }
23 }
24
```

HotelService

```
2 package jobsheet6;
3
4 public class HotelService {
5     Hotel14 hotels[] = new Hotel14[4];
6     int index;
7
8     void tambah(Hotel14 H){
9         if (index < hotels.length) {
10             hotels[index] = H;
11             index++;
12         } else {
13             System.out.println(x:"Data Sudah Penuh!!");
14         }
15     }
16
17     void tampilAll(){
18         for (Hotel14 H : hotels) {
19             H.tampilAll();
20         }
21     }
22
23     void bubbleSort(){
24         for (int i = 0; i < hotels.length-1; i++) {
25             for (int j = 1; j < hotels.length-i; j++) {
26                 if (hotels[j].harga < hotels[j-1].harga) {
27                     Hotel14 tmp = hotels[j];
28                     hotels[j] = hotels[j-1];
29                     hotels[j-1] = tmp;
30                 }
31             }
32         }
33     }
34
35     void selectionSort(){
36         for (int i = 0; i < hotels.length-1; i++) {
37             int idxMin = i;
38             for (int j = i+1; j < hotels.length; j++) {
39                 if (hotels[j].harga < hotels[idxMin].harga) {
40                     idxMin = j;
41                 }
42             }
43             Hotel14 tmp = hotels[idxMin];
44             hotels[idxMin] = hotels[i];
45             hotels[i] = tmp;
46         }
47     }
48 }
```

MainHotel

```
package jobsheet6;

public class MainHotel {

    Run | Debug
    public static void main(String[] args) {
        HotelService daftar = new HotelService();

        Hotel14 ht11 = new Hotel14(n:"NANIK", k:"MALANG", h:200000, (byte) 2);
        Hotel14 ht12 = new Hotel14(n:"BUDI", k:"BATU", h:300000, (byte) 3);
        Hotel14 ht13 = new Hotel14(n:"MIKASA", k:"NGANDUK", h:400000, (byte) 4);
        Hotel14 ht14 = new Hotel14(n:"TITAN", k:"PASURUAN", h:500000, (byte) 5);

        daftar.tambah(ht11);
        daftar.tambah(ht12);
        daftar.tambah(ht13);
        daftar.tambah(ht14);

        System.out.println(x:"-----");
        System.out.println(x:"          DATA SEBELUM SORTING          ");
        System.out.println(x:"-----");
        daftar.tampilAll();

        System.out.println();
        System.out.println(x:"-----");
        System.out.println(x:"          DAFTAR HARGA SETELAH SORTING (ASC) MENGGUNAKAN BUBBLE SORT          ");
        System.out.println(x:"-----");
        daftar.bubbleSort();
        daftar.tampilAll();

        System.out.println();
        System.out.println(x:"-----");
        System.out.println(x:"          DAFTAR HARGA SETELAH SORTING (DSC) MENGGUNAKAN SELECTION SORT          ");
        System.out.println(x:"-----");
        daftar.selectionSort();
        daftar.tampilAll();
    }
}
```

Hasil:

```
-----  
-                               DATA SEBELUM SORTING                               -  
-----  
  
Nama = NANIK  
Kota = MALANG  
Harga = 200000  
Bintang = 2  
Nama = BUDI  
Kota = BATU  
Harga = 300000  
Bintang = 3  
Nama = MIKASA  
Kota = NGANJUK  
Harga = 400000  
Bintang = 4  
Nama = TITAN  
Kota = PASURUAN  
Harga = 500000  
Bintang = 5
```

```
-----  
-   DAFTAR HARGA SETELAH SORTING (ASC) MENGGUNAKAN BUBBLE SORT   -  
-----  
  
Nama = NANIK  
Kota = MALANG  
Harga = 200000  
Bintang = 2  
Nama = BUDI  
Kota = BATU  
Harga = 300000  
Bintang = 3  
Nama = MIKASA  
Kota = NGANJUK  
Harga = 400000  
Bintang = 4  
Nama = TITAN  
Kota = PASURUAN  
Harga = 500000  
Bintang = 5
```

```
-----  
-   DAFTAR HARGA SETELAH SORTING (DSC) MENGGUNAKAN SELECTION SORT   -  
-----  
  
Nama = TITAN  
Kota = PASURUAN  
Harga = 500000  
Bintang = 5  
Nama = MIKASA  
Kota = NGANJUK  
Harga = 400000  
Bintang = 4  
Nama = BUDI  
Kota = BATU  
Harga = 300000  
Bintang = 3  
Nama = NANIK  
Kota = MALANG  
Harga = 200000  
Bintang = 2
```