

LAPORAN PRAKTIKUM MINGGU KE-6
SORTING (BUBBLE, SELECTION, DAN INSERTION SORT)



Disusun oleh:

Daffa Aqila Rahmatullah

2041720098

D4 TEKNIK INFORMATIKA
TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG
2021

LAPORAN

A. KODE PROGRAM

5.2 Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan Bubble Sort

```
public class Mahasiswa {
    String nama;
    int thnMasuk, umur;
    double ipk;
    Mahasiswa (String n, int t, int u, double i){
        nama = n;
        thnMasuk = t;
        umur = u;
        ipk = i;
    }
    void Tampil(){
        System.out.println("Nama: " +nama);
        System.out.println("Tahun Masuk: "+thnMasuk);
        System.out.println("Umur: "+umur);
        System.out.println("Ipk: "+ ipk);
    }
}

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

    void tambah(Mahasiswa m){
        if (idx<listmhs.length)
        {
            listmhs[idx]=m;
            idx++;
        } else
        {
            System.out.println("Data Sudah Penuh!!");
        }
    }
}
```

```

        }
    }
    void tampil() {
        for(Mahasiswa m : listmhs)
        {
            m.Tampil();

System.out.println("=====");
        }
    }
    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;
                }
            }
        }
    }
}

public class MahasiswaMain {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        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();

```

5.3 Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan Selection Sort

```

public class Mahasiswa {
    String nama;
    int thnMasuk, umur;
    double ipk;
    Mahasiswa (String n, int t, int u, double i){
        nama = n;
        thnMasuk = t;
        umur = u;
        ipk = i;
    }
}

```

```

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

    void tambah(Mahasiswa m){
        if (idx<listmhs.length)
        {
            listmhs[idx]=m;
            idx++;
        } else
        {
            System.out.println("Data Sudah Penuh!!");
        }
    }
    void tampil(){
        for(Mahasiswa m : listmhs)
        {
            m.Tampil();

System.out.println("=====");
        }
    }
    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;
                }
            }
        }
    }
}

```

```

        }
    }
}

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;
    }
}

public class MahasiswaMain {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        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();

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

```

5.4 Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan Insertion Sort

```

public class Mahasiswa {
    String nama;
    int thnMasuk, umur;
    double ipk;
    Mahasiswa (String n, int t, int u, double i){
        nama = n;
        thnMasuk = t;
        umur = u;
        ipk = i;
    }
}

```

```

    }

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

        void tambah(Mahasiswa m) {
            if (idx < listmhs.length)
            {
                listmhs[idx] = m;
                idx++;
            } else
            {
                System.out.println("Data Sudah Penuh!!");
            }
        }

        void tampil() {
            for (Mahasiswa m : listmhs)
            {
                m.Tampil();
            }

            System.out.println("=====");
        }

        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;
    }
}

}

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

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;
    }
}

```

```

        public static void main(String[] args) {
            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();

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

```

```

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

```

B. OUTPUT PROGRAM

5.2 Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan Bubble Sort

```

Output - bubble-selection-insertion (run-single) X
javac -includes=jobsheet6\MahasiswaMain.java -Drun.class=jobsheet6.MahasiswaMain run-single
init:
Deleting: D:\Algostrkdat\bubble-selection-insertion\build\build-jar.properties
deps-jar:
[ Updating property file: D:\Algostrkdat\bubble-selection-insertion\build\build-jar.properties
s
Compiling 1 source file to D:\Algostrkdat\bubble-selection-insertion\build\classes
compile-single:
run-single:
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: Umi
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: Umi
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
=====
BUILD SUCCESSFUL (total time: 1 second)

```

5.3 Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan Selection Sort

```
put - bubble-selection-insertion (run-single) X
Deleting: D:\Algostrkdat\bubble-selection-insertion\build\build-jar.properties
depe-jar:
[Updating property file: D:\Algostrkdat\bubble-selection-insertion\build\build-jar.properties
]
Compiling 1 source file to D:\Algostrkdat\bubble-selection-insertion\build\classes
compile-single:
run-single:
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: Dompus
Tahun Masuk: 2018
Umur: 19
Ipk: 3.5
=====
Nama: Abdul
Tahun Masuk: 2017
Umur: 23
Ipk: 2.0
=====
Nama: Umni
Tahun Masuk: 2019
Umur: 21
Ipk: 3.75
=====
```

```
Output - bubble-selection-insertion (run-single) X
=====
Nama: Umni
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: Dompus
Tahun Masuk: 2018
Umur: 19
Ipk: 3.5
=====
Nama: Umni
Tahun Masuk: 2019
Umur: 21
Ipk: 3.75
=====
Nama: Rara
Tahun Masuk: 2012
Umur: 19
Ipk: 4.0
=====
BUILD SUCCESSFUL (total time: 1 second)
```

5.4 Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan Insertion Sort

```
Output - bubble-selection-insertion (run-single) X
init:
Deleting: D:\Algostrkdat\bubble-selection-insertion\build\build-jar.properties
depe-jar:
[Updating property file: D:\Algostrkdat\bubble-selection-insertion\build\build-jar.properties
]
Compiling 1 source file to D:\Algostrkdat\bubble-selection-insertion\build\classes
compile-single:
run-single:
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: Dompus
Tahun Masuk: 2018
Umur: 19
Ipk: 3.5
=====
Nama: Abdul
Tahun Masuk: 2017
Umur: 23
Ipk: 2.0
=====
Nama: Umni
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: Rusa
Tahun Masuk: 2017
Umur: 25
Ipk: 3.0
=====
Nama: Dampu
Tahun Masuk: 2018
Umur: 19
Ipk: 3.5
=====
Nama: Umni
Tahun Masuk: 2019
Umur: 21
Ipk: 3.75
=====
Nama: Rara
Tahun Masuk: 2012
Umur: 19
Ipk: 4.0
=====
BUILD SUCCESSFUL (total time: 1 second)

```

C. PENJELASAN

5.2.3 Pertanyaan

1. Terdapat di method apakah proses bubble sort?

Jawab: Proses bubble sort terdapat pada class DaftarMahasiswaBerprestasi() dan terdapat di method void bubbleSort().

2. Terdapat di method apakah proses selection sort?

Jawab: Proses selection sort terdapat pada class DaftarMahasiswaBerprestasi() dan terdapat di method void selectionSort (). Dan pada praktikum 1 belum ada proses selection sort

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

Jawab: proses swap adalah proses pertukaran nilai baik itu secara ascending maupun descending

Script

```

if(listmhs[j].ipk > listmhs[j-1].ipk)
{
    Mahasiswa tmp = listmhs[j];
    listmhs[j] = listmhs[j-1];
    listmhs[j-1] = tmp;
}

```

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

```

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

```

Untuk apakah proses tersebut?

Jawab: proses di atas adalah proses swap untuk pertukaran nilai di array list mahasiswa.

5. Perhatikan perulangan didalam bubbleSort() dibawah 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?

Jawab: Untuk perulangan i fungsinya menampilkan baris dari listMhs sedangkan j untuk menampilkan listMhs kolom

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

Jawab: jadi setiap perulangan akan mengurangi panjang listMhs dengan -1

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

Jawab: Jadi setiap perulangan akan mengurangi panjang listMhs dengan - i

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

Jawab: pada perulangan i sebanyak 49 kali dan pada tahap bubble sort

5.3.3. Pertanyaan

Didalam 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?

5.4.3. 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.

Jawab:

Script:

```
public class Mahasiswa {
    String nama;
```

```

        int thnMasuk, umur;
        double ipk;
        Mahasiswa (String n, int t, int u, double i){
            nama = n;
            thnMasuk = t;
            umur = u;
            ipk = i;
        }
        void Tampil(){
            System.out.println("Nama: " +nama);
            System.out.println("Tahun          Masuk:
"+thnMasuk);
            System.out.println("Umur: "+umur);
            System.out.println("Ipk: "+ ipk);
        }
    }
    public class DaftarMahasiswaBerprestasi {
        Mahasiswa listmhs[]= new Mahasiswa[5];
        int idx;
        void insertionSort(boolean y){
            for (int i=1; i<listmhs.length; i++)
            {
                Mahasiswa temp = listmhs[i];
                int j =i;
                if(y==true){
                    while(j > 0 && listmhs[j-1].ipk >
temp.ipk)
                    {
                        listmhs[j] = listmhs[j-1];
                        j--;
                    }
                }else {
                    while(j > 0 && listmhs[j-1].ipk <
temp.ipk)

```

```

        {
            listmhs[j] = listmhs[j-1];
            j--;
        }
    }
    listmhs[j] = temp;
}
}

public class MahasiswaMain {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        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);
    }
}

```



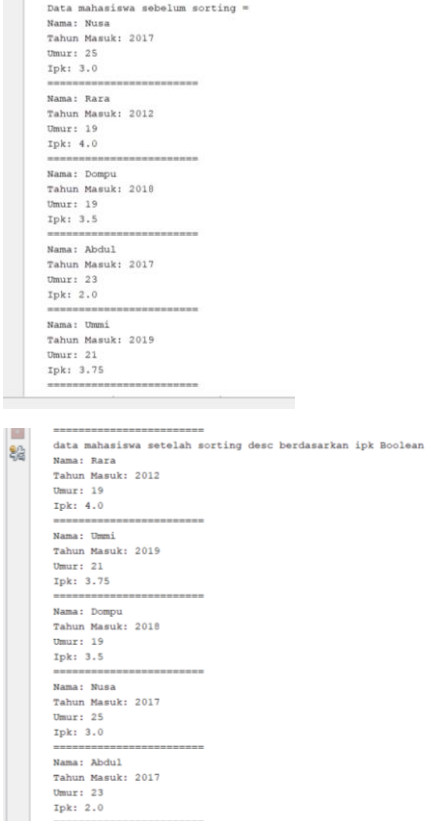
```

        System.out.println("Data mahasiswa sebelum
sorting = ");
        list.tampil();
System.out.println("data mahasiswa setelah sorting
desc berdasarkan ipk Boolean");
        list.insertionSort(false);
        list.tampil();

        System.out.println("data mahasiswa setelah
sorting asc berdasarkan ipk Boolean");
        list.insertionSort(true);
        list.tampil();
    }
}

```

Hasil Output



```

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 Boolean
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
=====

```

```
Output - bubble-selection-insertion (run-single) x
=====
Nama: Abdul
Tahun Masuk: 2017
Umur: 23
Ipk: 2.0
=====
data mahasiswa setelah sorting asc berdasarkan ipk Boolean
Nama: Abdul
Tahun Masuk: 2017
Umur: 23
Ipk: 2.0
=====
Nama: Nusa
Tahun Masuk: 2017
Umur: 25
Ipk: 3.0
=====
Nama: Dampu
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
=====
BUILD SUCCESSFUL (total time: 2 seconds)
```

Penjelasan:

Di dalam class DaftarMahasiswaBerprestasi membuat method baru yaitu insertion sort tetapi dengan menggunakan parameter di dalamnya tidak jauh beda dengan sebelumnya tetapi ditambahkan bentuk perulangan seperti sintaks di bawah ini:

```
if(y==true){
    while(j > 0 && listmhs[j-1].ipk > temp.ipk)
    {
        listmhs[j] = listmhs[j-1];
        j--;
    }
}
```

Dan diatas merupakan sintaks jika pernyataan benar maka akan dilakukan ascending

Dan selanjutnya perhatikan sintaks dibawah ini:

```
}else {
    while(j > 0 && listmhs[j-1].ipk < temp.ipk)
    {
        listmhs[j] = listmhs[j-1];
        j--;
    }
}
listmhs[j] = temp;
```

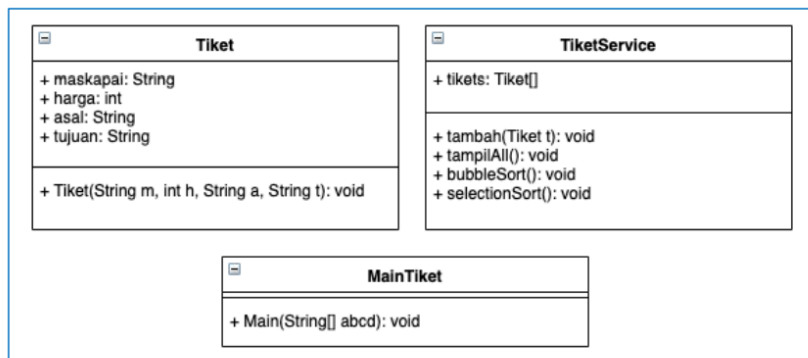
di atas merupakan sintaks jika pernyataan salah maka akan dilakukan descending

D. KESIMPULAN

Kesimpulan yang saya dapat yaitu saya bisa mengetahui tentang bubble sort insertion sort dan selection sort dan saya mohon maaf jika laporan saya masih banyak kekurangannya

TUGAS

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.



A. KODE PROGRAM

```
public class Tiket {
    String Maskapai, Asal, Tujuan;
    int harga;

    public Tiket(String m, int h, String a, String t){
        Maskapai = m;
        harga = h;
        Asal = a;
        Tujuan = t;
    }
}

public class TiketService {
    Tiket[] tickets = new Tiket[5];
    int a;
    void tambah(Tiket t){
```

```

        if (a<tickets.length)
        {
            tickets[a] = t;
            a++;
        } else
        {
            System.out.println("Data Sudah Penuh!!");
        }
    }

    void tampil(){
        for(int i=0;i<tickets.length;i++)
        {
            int b=1;
            b+=i;
            System.out.println(b+"    Daftar Tiket");
            System.out.println("Maskapai :
"+tickets[i].Maskapai);
            System.out.println("Harga \t:
Rp."+tickets[i].harga);
            System.out.println("Asal \t:
"+tickets[i].Asal);
            System.out.println("Tujuan \t:
"+tickets[i].Tujuan);

            System.out.println("=====");
            b++;
        }
    }

    void bubbleSort(){
        for (int i=0; i<tickets.length-1; i++)

```

```

        {
            for (int j=1; j<tickets.length-i; j++)
            {
                if(tickets[j].harga > tickets[j-1].harga)
                {
                    Tiket tmp = tickets[j];
                    tickets[j] = tickets[j-1];
                    tickets[j-1] = tmp;
                }
            }
        }
    }

    void selectionSort(){
        for (int i=0; i<tickets.length-1; i++)
        {
            int idxMin = i;
            for (int j=i+1; j<tickets.length-i; j++)
            {
                if(tickets[j].harga <
tickets[idxMin].harga)
                {
                    idxMin = j;
                }
            }
            Tiket tmp = tickets[idxMin];
            tickets[idxMin] = tickets[i];
            tickets[i] = tmp;
        }
    }
}

public class TiketMain {

    /**

```

```

    * @param args the command line arguments
    */
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        TiketService list = new TiketService();
        Tiket t1 = new Tiket("Garuda", 980600, "
semarang", "jakarta");
        Tiket t2 = new Tiket("Air Asia", 541000, "
surabaya", "semarang");
        Tiket t3 = new Tiket("Citilink", 949420, " medan",
"semarang");
        Tiket t4 = new Tiket("Lion Air", 1380600, "
semarang", "aceh");
        Tiket t5 = new Tiket("Batik Air", 2754200, "
jakarta", "Manokwari");

        list.tambah(t1);
        list.tambah(t2);
        list.tambah(t3);
        list.tambah(t4);
        list.tambah(t5);

        int ulang = 0;
        do{
            System.out.println("\t\t\t\t\tProgram Tiket
Pesawat\t\t\t");
            System.out.println("");
            System.out.println("Pilih menu dibawah
ini!");
            System.out.println("1. Memunculkan Tiket Yang
belum di sorting");
            System.out.println("2. Memunculkan Tiket dari
termahal sampai termurah");

```

```

        System.out.println("3. Memunculkan Tiket dari
termurah sampai termahal");
        System.out.println("4. Keluar");
        System.out.print("Pilihah menu :");
        int pilih = sc.nextInt();
        if(pilih == 1){
            System.out.println("Tiket Pesawat Sebelum
di sorting = ");
            list.tampil();
            continue;
        }else if(pilih == 2){
            System.out.println("Tiket Pesawat Sebelum
di sorting = ");
            list.tampil();
            System.out.println("===== bubble
sort =====");
            System.out.println("Menampilkan data
Tiket Pesawat");
            list.bubbleSort();
            list.tampil();
            continue;
        }else if(pilih ==3){
            System.out.println("Tiket Pesawat Sebelum
di sorting = ");
            list.tampil();
            System.out.println("=====
selectionSort =====");
            list.selectionSort();
            list.tampil();
        }else if(pilih==4){
            System.out.println("Anda telah keluar
dari program");
            break;
        }else{

```



```

        System.out.println("Menu yang anda
berikan tidak tersedia");

        System.out.println("");

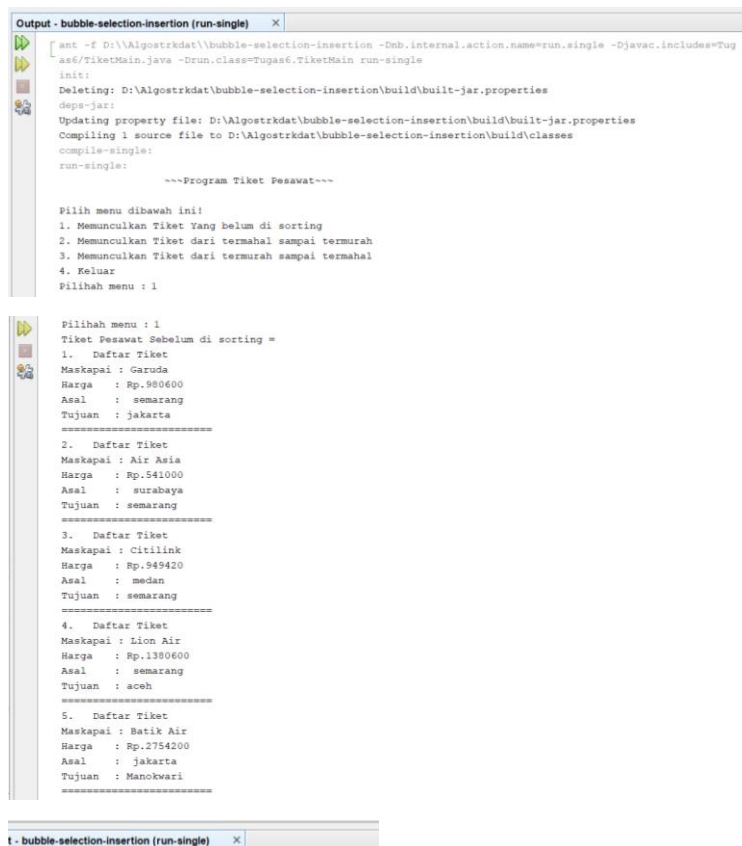
    }

    }while(ulang !=0 || ulang<3);

}
}

```

B. OUTPUT PROGRAM



```

Output - bubble-selection-insertion (run-single) X
[ant -f D:\Algostrkdat\bubble-selection-insertion -Dmh.internal.action.name=run.single -Djavac.includes=Tugas6/TiketMain.java -Drun.class=Tugas6.TiketMain run-single
init:
Deleting: D:\Algostrkdat\bubble-selection-insertion\build\build-jar.properties
deps-jar:
Updating property file: D:\Algostrkdat\bubble-selection-insertion\build\build-jar.properties
Compiling 1 source file to D:\Algostrkdat\bubble-selection-insertion\build\classes
compile-single:
run-single:

    ~~~Program Tiket Pesawat~~~

Pilih menu dibawah ini!
1. Memunculkan Tiket Yang belum di sorting
2. Memunculkan Tiket dari termahal sampai termurah
3. Memunculkan Tiket dari termurah sampai termahal
4. Keluar
Pilihah menu : 1

Pilihah menu : 1
Tiket Pesawat Sebelum di sorting =
1. Daftar Tiket
Maskapai : Garuda
Harga : Rp.900600
Asal : semarang
Tujuan : jakarta
=====
2. Daftar Tiket
Maskapai : Air Asia
Harga : Rp.541000
Asal : surabaya
Tujuan : semarang
=====
3. Daftar Tiket
Maskapai : Citilink
Harga : Rp.949420
Asal : medan
Tujuan : semarang
=====
4. Daftar Tiket
Maskapai : Lion Air
Harga : Rp.1300600
Asal : semarang
Tujuan : aceh
=====
5. Daftar Tiket
Maskapai : Batik Air
Harga : Rp.2754200
Asal : jakarta
Tujuan : Manokwari
=====

t - bubble-selection-insertion (run-single) X

    ~~~Program Tiket Pesawat~~~

Pilih menu dibawah ini!
1. Memunculkan Tiket Yang belum di sorting
2. Memunculkan Tiket dari termahal sampai termurah
3. Memunculkan Tiket dari termurah sampai termahal
4. Keluar
Pilihah menu : 2

```

```

Pilihah menu : 2
Tiket Pesawat Sebelum di sorting =
1.  Daftar Tiket
Maskapai : Garuda
Harga   : Rp.980600
Asal    : semarang
Tujuan  : jakarta
=====
2.  Daftar Tiket
Maskapai : Air Asia
Harga   : Rp.541000
Asal    : surabaya
Tujuan  : semarang
=====
3.  Daftar Tiket
Maskapai : Citilink
Harga   : Rp.949420
Asal    : medan
Tujuan  : semarang
=====
4.  Daftar Tiket
Maskapai : Lion Air
Harga   : Rp.1380600
Asal    : semarang
Tujuan  : aceh
=====
5.  Daftar Tiket
Maskapai : Batik Air
Harga   : Rp.2754200
Asal    : jakarta
Tujuan  : Manokwari
=====

```

```

Output - bubble-selection-insertion (run-single) X
===== bubble sort =====
Menampilkan data Tiket Pesawat
1.  Daftar Tiket
Maskapai : Batik Air
Harga   : Rp.2754200
Asal    : jakarta
Tujuan  : Manokwari
=====
2.  Daftar Tiket
Maskapai : Lion Air
Harga   : Rp.1380600
Asal    : semarang
Tujuan  : aceh
=====
3.  Daftar Tiket
Maskapai : Garuda
Harga   : Rp.980600
Asal    : semarang
Tujuan  : jakarta
=====
4.  Daftar Tiket
Maskapai : Citilink
Harga   : Rp.949420
Asal    : medan
Tujuan  : semarang
=====
5.  Daftar Tiket
Maskapai : Air Asia
Harga   : Rp.541000
Asal    : surabaya
Tujuan  : semarang
=====

~~~Program Tiket Pesawat~~~

Pilih menu dibawah ini!
1. Memunculkan Tiket Yang belum di sorting
2. Memunculkan Tiket dari termahal sampai termurah
3. Memunculkan Tiket dari termurah sampai termahal
4. Keluar
Pilihah menu : 3

```

```

Pilihah menu : 3
Tiket Pesawat Sebelum di sorting =
1.  Daftar Tiket
Maskapai : Batik Air
Harga   : Rp.2754200
Asal    : jakarta
Tujuan  : Manokwari
=====
2.  Daftar Tiket
Maskapai : Lion Air
Harga   : Rp.1380600
Asal    : semarang
Tujuan  : aceh
=====
3.  Daftar Tiket
Maskapai : Garuda
Harga   : Rp.980600
Asal    : semarang
Tujuan  : jakarta
=====
4.  Daftar Tiket
Maskapai : Citilink
Harga   : Rp.949420
Asal    : medan
Tujuan  : semarang
=====
5.  Daftar Tiket
Maskapai : Air Asia
Harga   : Rp.541000
Asal    : surabaya
Tujuan  : semarang
=====

```

```
Output - bubble-selection-insertion (run-single) X
Tujuan : semarang
=====
===== selectionSort =====
1. Daftar Tiket
Maskapai : Air Asia
Harga : Rp.541000
Asal : surabaya
Tujuan : semarang
=====
2. Daftar Tiket
Maskapai : Citilink
Harga : Rp.949420
Asal : medan
Tujuan : semarang
=====
3. Daftar Tiket
Maskapai : Garuda
Harga : Rp.980600
Asal : semarang
Tujuan : jakarta
=====
4. Daftar Tiket
Maskapai : Lion Air
Harga : Rp.1380600
Asal : semarang
Tujuan : aceh
=====
5. Daftar Tiket
Maskapai : Batik Air
Harga : Rp.2754200
Asal : jakarta
Tujuan : Manokwari
```

C. PENJELASAN

Didalam class Tiket terdapat 2 tipe data yaitu String dan Integer. Dan mempunyai method dengan parameter. Selanjutnya di dalam class tiketService terdapat array dengan elemen 5 dan integer selanjutnya ada method tambah dengan parameter didalam isinya jika isinya kurang dari array langsung ditambahkan dan jika lebih dari array langsung keluar data sudah penuh selanjutnya terdapat method tampil untuk menampilkan array selanjutnya terdapat method bubblesort untuk mengurutkan data tiket dari termahal ke termurah selanjutnya terdapat method selectionsort untuk mengurutkan data tiket dari termurah ke termahal selanjutnya di main class sebelumnya di instansiasi terlebih dahulu dan selanjutnya baru mengisi data dengan parameter selanjutnya nama variabel tersebut di masukkan di method tambah dan selanjutnya data tersebut dikeluarkan

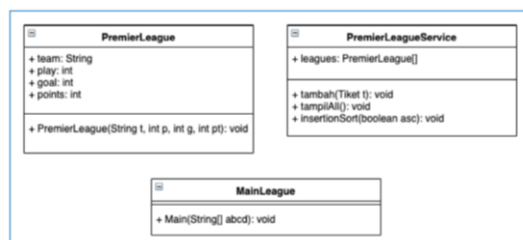
D. Kesimpulan

Kesimpulan yang saya dapat yaitu saya bisa mengetahui bagaimana cara menyelesaikan studi kasus yang berkaitan dengan pengurutan dan saya mohon maaf jika laporan saya masih banyak kekurangannya

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

<	Premier League	>	P	GD	PTS
1	Liverpool		29	45	82
2	Manchester City		27	39	57
3	Leicester	Manchester City	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

Ubahlah data klasemen 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



E. KODE PROGRAM

```

public class PremierLeague {
    String team;
    int play, goal, points;

    public PremierLeague(String t, int p, int g, int pt){
        team = t;
        play = p;
        goal = g;
        points = pt;
    }
}
  
```

```

public class PremierLeagueService {
    PremierLeague[] Leagues = new PremierLeague[20];
    int a;
    void tambah(PremierLeague t)
    {

        if (a<Leagues.length)
        {
            Leagues[a] = t;
            a++;
        } else
        {
            System.out.println("Data Sudah Penuh!!");
        }

    }

    void tampilall()
    {
        System.out.println("|\\tKlub          |");
        for(int i=0;i<Leagues.length;i++)
        {
            int a;
            a=i+1;
            System.out.println(a +" " + Leagues[i].team
+"\\t\\t | "
                        +Leagues[i].play + " | "
+Leagues[i].goal + " | " +Leagues[i].points+"|");
            System.out.println("-----");
            a++;
        }
    }

    void insertionSort(boolean asc){
        if(asc == true){
            for (int i=1; i<Leagues.length; i++)
            {
                PremierLeague temp = Leagues[i];
                int j =i;
                while(j > 0 && Leagues[j-1].points <
temp.points)
                {
                    Leagues[j] = Leagues[j-1];
                    j--;
                }
                Leagues[j] = temp;
            }
        } else
        {
            for (int i=1; i<Leagues.length; i++)
            {

```

```

PremierLeague temp = Leagues[i];
    int j =i;
    while(j > 0 && Leagues[j-1].points >
temp.points)
        {
            Leagues[j] = Leagues[j-1];
            j--;
        }
        Leagues[j] = temp;
    }
}

}

}

public class LeagueMain {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        PremierLeagueService list = new
PremierLeagueService();
        PremierLeague t1 = new PremierLeague ("Chelsea
\t\t\t", 29,9, 48);
        PremierLeague t2 = new PremierLeague("Manchester
City \t\t", 27,39, 57);
        PremierLeague t3 = new PremierLeague("Manchester
United \t\t", 28,12, 42);
        PremierLeague t4 = new PremierLeague("Burnley
\t\t\t", 29,-6, 39);
        PremierLeague t5 = new PremierLeague("Arsenal
\t\t\t", 28,4, 40);
        PremierLeague t6 = new PremierLeague("Tottenham
Hotspur \t\t", 29,7, 41);
        PremierLeague t7 = new PremierLeague("Everton
\t\t\t", 29,-6, 37);
        PremierLeague t8 = new PremierLeague("West Ham
United \t\t", 29,-15, 27);
        PremierLeague t9 = new PremierLeague("Watford
\t\t\t", 29,-17, 27);
        PremierLeague t10 = new
PremierLeague("Southampton \t\t\t", 29,-17, 34);
        PremierLeague t11 = new PremierLeague("Newcastle
United \t\t", 29,-16, 35);
        PremierLeague t12 = new PremierLeague("Aston
Villa \t\t\t", 27,-18, 25);
        PremierLeague t13 = new PremierLeague("Leicester
\t\t\t", 28,26, 50);
        PremierLeague t14 = new PremierLeague("Brighton &
Hove Albion \t", 29,-8, 29);

```

```

PremierLeague t15 = new PremierLeague("Norwich
City \t\t", 29,-27, 21);
PremierLeague t16 = new PremierLeague("Liverpool
\t\t\t", 29,45, 82);
PremierLeague t17 = new PremierLeague("AFC
Bournemouth \t\t", 29,-18, 27);
PremierLeague t18 = new PremierLeague("Cristal
Palace \t\t", 29,-6, 39);
PremierLeague t19 = new PremierLeague("Sheffield
United \t\t", 28,5, 43);
PremierLeague t20 = new
PremierLeague("Wolverhampton Wanderes \t", 29,7, 43);

```

```

list.tambah(t1);
list.tambah(t2);
list.tambah(t3);
list.tambah(t4);
list.tambah(t5);
list.tambah(t6);
list.tambah(t7);
list.tambah(t8);
list.tambah(t9);
list.tambah(t10);
list.tambah(t11);
list.tambah(t12);
list.tambah(t13);
list.tambah(t14);
list.tambah(t15);
list.tambah(t16);
list.tambah(t17);
list.tambah(t18);
list.tambah(t19);
list.tambah(t20);

```

```

System.out.println("Klasemen sebelum sorting =
");
list.tampilall();

System.out.println("Klasemen setelah sorting asc
berdasarkan Point");
list.insertionSort(true);
list.tampilall();
System.out.println("Klasemen setelah sorting desc
berdasarkan Point");
list.insertionSort(false);
list.tampilall();
}

```

F. OUTPUT PROGRAM

```

Klasemen sebelum sorting =
| Klub |
1) Chelsea | 29 | 9 | 48|
-----
2) Manchester City | 27 | 39 | 57|
-----
3) Manchester United | 28 | 12 | 42|
-----
4) Burnley | 29 | -6 | 39|
-----
5) Arsenal | 28 | 4 | 40|
-----
6) Tottenham Hotspur | 29 | 7 | 41|
-----
7) Everton | 29 | -6 | 37|
-----
8) West Ham United | 29 | -15 | 27|
-----
9) Watford | 29 | -17 | 27|
-----
10) Southampton | 29 | -17 | 34|
-----

11) Newcastle United | 29 | -16 | 35|
-----
12) Aston Villa | 27 | -18 | 25|
-----
13) Leicester | 28 | 26 | 50|
-----
14) Brighton & Hove Albion | 29 | -8 | 29|
-----
15) Norwich City | 29 | -27 | 21|
-----
16) Liverpool | 29 | 45 | 82|
-----
17) AFC Bournemouth | 29 | -18 | 27|
-----
18) Crystal Palace | 29 | -6 | 39|
-----
19) Sheffield United | 28 | 5 | 43|
-----
20) Wolverhampton Wanderers | 29 | 7 | 43|
-----

Klasemen setelah sorting desc berdasarkan Point
| Klub |
1) Liverpool | 29 | 45 | 82|
-----
2) Manchester City | 27 | 39 | 57|
-----
3) Leicester | 28 | 26 | 50|
-----
4) Chelsea | 29 | 9 | 48|
-----
5) Sheffield United | 28 | 5 | 43|
-----
6) Wolverhampton Wanderers | 29 | 7 | 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|
-----

Output - bubble-selection-insertion (run-single) x
Klasemen setelah sorting asc berdasarkan Point
| Klub |
1) Norwich City | 29 | -27 | 21|
-----
2) Aston Villa | 27 | -18 | 25|
-----
3) West Ham United | 29 | -15 | 27|
-----
4) Watford | 29 | -17 | 27|
-----
5) AFC Bournemouth | 29 | -18 | 27|
-----
6) Brighton & Hove Albion | 29 | -8 | 29|
-----
7) Southampton | 29 | -17 | 34|
-----
8) Newcastle United | 29 | -16 | 35|
-----
9) Everton | 29 | -6 | 37|
-----
10) Burnley | 29 | -6 | 39|
-----

```


11) Cristal Palace	29 -6 39
12) Arsenal	28 4 40
13) Tottenham Hotspur	29 7 41
14) Manchester United	28 12 42
15) Sheffield United	28 5 43
16) Wolverhampton Wanderes	29 7 43
17) Chelsea	29 9 48
18) Leicester	28 26 50
19) Manchester City	27 39 57
20) Liverpool	29 45 82

BUILD SUCCESSFUL (total time: 1 second)

G. PENJELASAN

Didalam class PremierLeague terdapat 2 tipe data yaitu String dan Integer. Dan mempunyai method dengan parameter. Selanjutnya di dalam class PremierLeagueService terdapat array dengan elemen 20 dan interger selanjutnya ada method tambah dengan parameter didalam isinya jika isinya kurang dari array langsung ditambahkan dan jika lebih dari array langsung keluar data sudah penuh selanjutnya terdapat method tampil untuk menampilkan array selanjutnya pada method insertionsort dengan parameter boolean didalamnya terdapat jika true maka akan mengulang dengan descending jika false maka akan mengulang dengan ascending didalam main sebelumnya di instansiasi terlebih dahulu dan selanjutnya baru mengisi data dengan parameter selanjutnya nama variabel tersebut di masukkan di method tambah dan selanjutnya data tersebut dikeluarkan

H. Kesimpulan

Kesimpulan yang saya dapat yaitu saya bisa mengetahui tentang bagaimana cara menyelesaikan studi kasus yang berkaitan dengan pengurutan dan saya mohon maaf jika laporan saya masih banyak kekurangannya