LAPORAN HASIL PRAKTIKUM

ALGORITMA DAN STRUKTUR DATA

JOBSHEET 5

NAMA :MAULIDYAAFRIANI

NIM: 2441070200559

KELAS : 1E

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2025

1.1 PERCOBAAN 1

1. Buat folder baru dengan nama **jobsheet05**.

2. Buat class **Sorting**<No Presensi>, kemudian tambahkan atribut sebagai berikut:

```
public class Sorting15 {

    int [] data;

    int jumData;
```

3. Buatlah konstruktor dengan parameter Data[] dan jmlDat

```
Sorting15 (int Data[], int jmlData) {

    jumData = jmlData;

    data=new int[jmlData];

    for (int i=0; i<jmlData; i++) {

        data[i]=Data[i];

    }

}
```

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

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

```
            }
        }
    }
}
```

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

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

6. Buat class **SortingMain**<15> kemudian deklarasikan array dengan nama a[] kemudian isi array tersebut

```
public class SortingMain15 {
    public static void main(String[] args) {
        int a[] = {20, 10, 2, 7, 12};
```

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

```
Sorting15 dataurut1 = new Sorting15(a, a.length);
```

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

```
     System.out.println("Data awal 1:");
        dataurut1.tampil();
        dataurut1.bubbleSort();
```

```
        System.out.println("Data sudah diurutkan dengan BUBBLE SORT
(ASC):");

        dataurut1.tampil();

    }

}
```

9. Jalankan program, dan amati hasilnya!

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

1. Pada class **Sorting**<15> yang sudah dibuat di praktikum sebelumnya tambahkan method **SelectionSort** yang mengimplementasikan pengurutan menggunakan algoritma selection sort.

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

}
```

2. Deklarasikan array dengan nama b[] pada kelas **SortingMain**<15> kemudian isi array tersebut. Buatlah objek baru dengan nama **dataurut2** yang merupakan instansiasi dari class **Sorting**, kemudian isi parameternya, Lakukan pemanggilan method **SelectionSort** dan t**ampil**

```
public class SortingMain15 {
```

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

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

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


        Sorting15 dataurut1 = new Sorting15(a, a.length);

       Sorting15 dataurut2 = new Sorting15(b, b.length);

        System.out.println("Data awal 1:");

        dataurut1.tampil();

        dataurut1.bubbleSort();


        System.out.println("Data  sudah  diurutkan  dengan  BUBBLE
SORT (ASC):");

        dataurut1.tampil();

        System.out.println("Data awal 2");

        dataurut2.tampil();

        dataurut2.selectionSort();

        System.out.println("data sudah diurutkan dengan SELECTION
SORT (ASC)");

    }

}
```

3. Jalankan program dan amati hasilnya!

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

1. Pada class **Sorting**<15> yang sudah dibuat di praktikum sebelumnya tambahkan method **insertionSort** yang mengimplementasikan pengurutan menggunakan algoritma insertion sort.

```
void insertionSort() {
```

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

        }
        data[j+1]=temp;
    }
}
}
```

2. Deklarasikan array dengan nama c[] pada kelas **SortingMain**<15> kemudian isi array tersebut. Buatlah objek baru dengan nama **dataurut3** yang merupakan instansiasi dari class **Sorting**, kemudian isi parameternya. Lakukan pemanggilan method **insertionSort** dan t**ampil**

```
public class SortingMain15 {

    public static void main(String[] args) {

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

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

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




        Sorting15 dataurut1 = new Sorting15(a, a.length);

       Sorting15 dataurut2 = new Sorting15(b, b.length);

       Sorting15 dataurut3 = new Sorting(c, c.length);


        System.out.println("Data awal 1:");

        dataurut1.tampil();

        dataurut1.bubbleSort();

        System.out.println("Data  sudah  diurutkan  dengan  BUBBLE
SORT (ASC):");

        dataurut1.tampil();


        System.out.println("Data awal 2");

        dataurut2.tampil();

        dataurut2.selectionSort();

        System.out.println("data sudah diurutkan dengan SELECTION
SORT (ASC)");
```

```
        dataurut2.tampil();


        System.out.println("Data awal 3");

        dataurut3.tampil();

        dataurut3.insertionSort();

        System.out.println("Data sudag diurutkan dengan INSERTION
SORT(ASC)");

        dataurut3.tampil();

    }

}
```

3. Jalankan program dan amati hasilnya!

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

Pertanyaan :

1. Menukar dua elemen dalam array jika elemen sebelumnya lebih besar daripada elemen sesudahnya, dilakukan untuk menyusun array secara menaik (ascending).
2. Kode yang merupakan algoritma pencarian nilai minimum pada selection sort adalah

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

3. Digunakan untuk menghitung berapa banyak elemen yang perlu digeser ke kanan agar elemen baru (temp) dapat disisipkan pada posisi yang tepat dalam proses pengurutan.
4. Elemen array digeser ke kanan satu posisi untuk memberikan ruang kosong bagi elemen yang akan disisipkan sesuai urutan yang benar.

1.2 PERCOBAAN 2

1. Buatlah class dengan nama **Mahasiswa**<15>.

2. Untuk lebih jelasnya class tersebut dapat dilihat pada potongan kode di bawah ini

```
public class Mahasiswa15 {


    String nim;
    String nama;
    String kelas;
    double ipk;


    Mahasiswa15() {


    }


    Mahasiswa15(String nm, String nma, String kls, double ip) {
        nim = nm;
        nama = nma;
        kelas = kls;
        ipk = ip;
    }


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

3. Buat class **MahasiswaBerprestasi**<15> seperti di bawah ini! Tambahkan method tambah() di dalam class tersebut! Method tambah() digunakan untuk menambahkan objek dari class Mahasiswa ke dalam atribut listMhs. 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. Tambahkan method bubbleSort() di dalam class tersebut!

```
public class mahasiswaBerprestasi15 {
    Mahasiswa15 [] listMhs = new Mahasiswa15[5];
    int idx;

    void tambah (Mahasiswa15 m) {
        if (idx<listMhs.length) {
            listMhs[idx] = m;
            idx++;
        }else {
            System.out.println("data sudah penuh");
        }
    }
    void tampil() {
        for (Mahasiswa15 m:listMhs) {
            m.tampilInformasi();
            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) {
                    Mahasiswa15 tmp =listMhs[j];
                    listMhs[j] = listMhs[j-1];
                    listMhs[j-1]  = tmp;
                }
            }
        }
    }
}
```

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

```
public class MahasiswaDemo15 {

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

```
        mahasiswaBerprestasi15 list = new
mahasiswaBerprestasi15();
        Mahasiswa15 m1 = new Mahasiswa15("123", "Zidan", "2A",
3.2);
        Mahasiswa15 m2 = new Mahasiswa15("124", "Ayu", "2A",
3.5);
        Mahasiswa15 m3 = new Mahasiswa15("125", "Sofi", "2A",
3.1);
        Mahasiswa15 m4 = new Mahasiswa15("126", "Sita", "2A",
3.9);
        Mahasiswa15 m5 = new Mahasiswa15("127", "Miki", "2A",
3.7);

        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
berdasarkan IPK (DESC) : ");
        list.bubbleSort();
        list.tampil();
    }
}
```

Hasil run

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

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

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

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

jawab: Setelah **n-1** iterasi, data akan terurut, karena elemen terbesar sudah menggelembung ke posisi akhir setelah setiap iterasi.

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

jawab: Setiap iterasi luar (i) memastikan elemen terbesar berada di posisi akhir, sehingga jumlah elemen yang perlu dibandingkan berkurang 1 pada iterasi berikutnya.

c. Jika banyak data di dalam listMhs adalah 50, maka berapakali perulangan i akan berlangsung?

Jawab: Dan ada berapa **Tahap** bubble sort yang ditempuh? Jika jumlah elemen **n = 50**, maka perulangan terjadi sebanyak **49 kali**, sesuai dengan batas iterasi i < n - 1.

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

```
public class Mahasiswa15 {



    String nim;
    String nama;
    String kelas;
    double ipk;

    public Mahasiswa15() {

    }

    public Mahasiswa15(String nm, String name, String kls, double ip) {
        nim = nm;
        nama = name;
        kelas = kls;
        ipk = ip;
    }

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

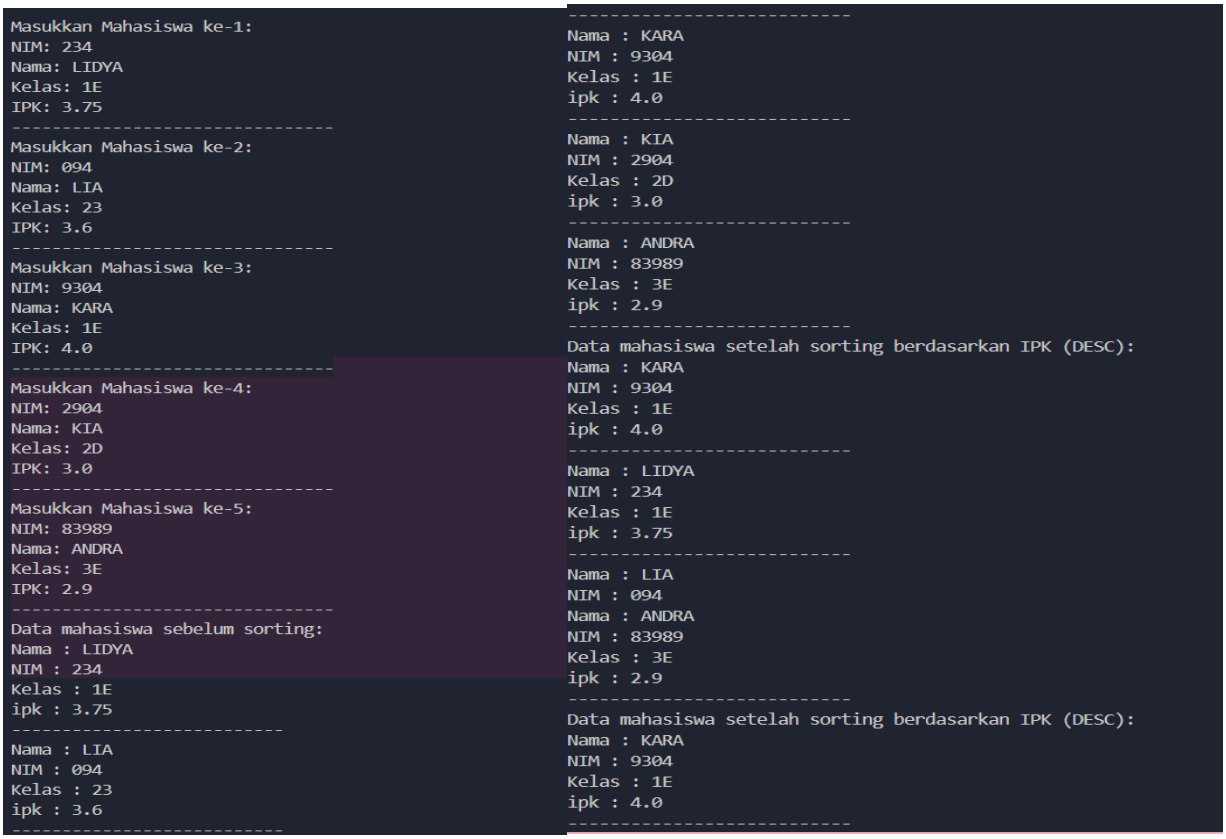
```
public class mahasiswaBerprestasi15 {
    Mahasiswa15 [] listMhs = new Mahasiswa15[5];
    int idx;

    void tambah (Mahasiswa15 m) {
        if (idx<listMhs.length) {
            listMhs[idx] = m;
            idx++;
        }else {
            System.out.println("data sudah penuh");
        }
```

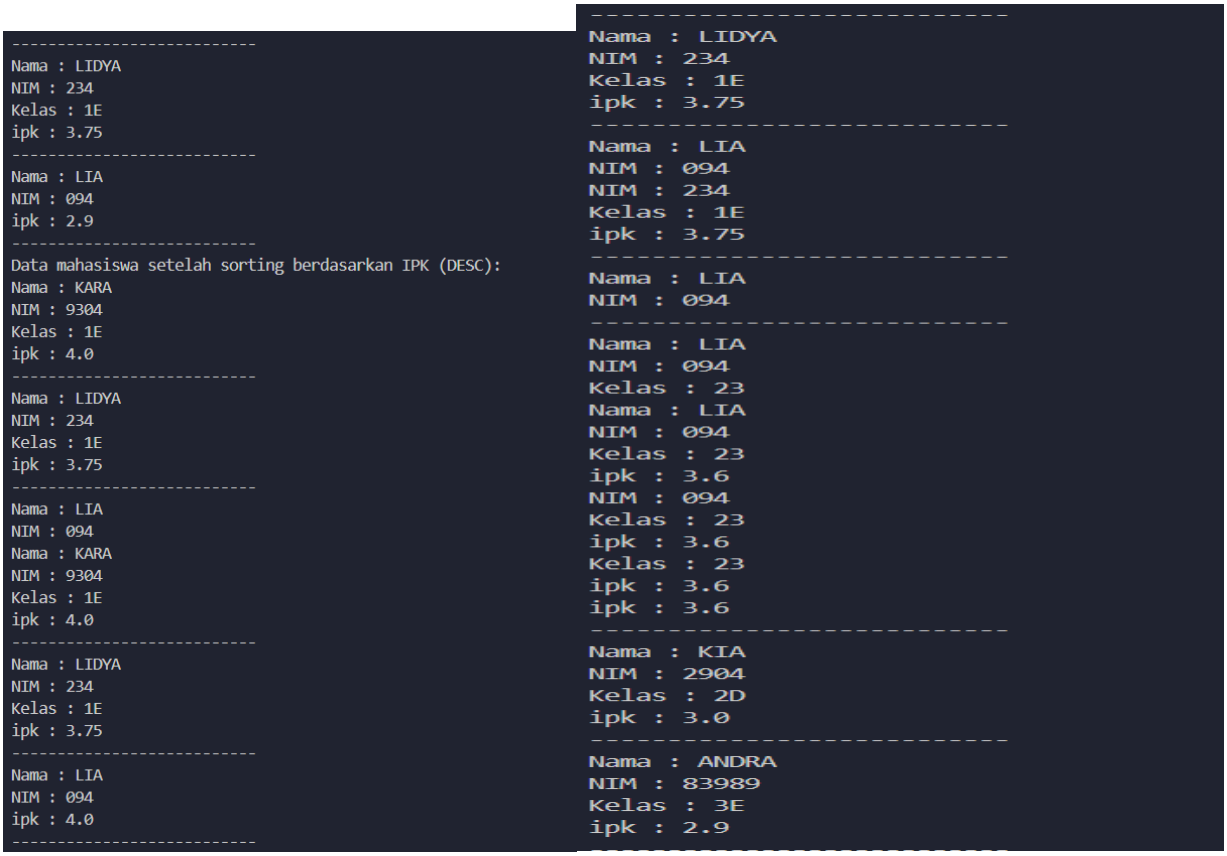
```
    }
    void tampil() {
        for (Mahasiswa15 m:listMhs) {
            m.tampilInformasi();
            System.out.println("-------------------------");
        }
    }
    void bubbleSort() {
        for (int i = 0; i < listMhs.length-1; i++) {
            for (int j = 1; j < listMhs.length-i; j++) {
                if (listMhs[j].ipk>listMhs[j-1].ipk) {
                    Mahasiswa15 tmp =listMhs[j];
                    listMhs[j] = listMhs[j-1];
                    listMhs[j-1]  = tmp;
                }
            }
        }
    }
}
```

```
import java.util.Scanner;

public class MahasiswaDemo15 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        mahasiswaBerprestasi15 list = new mahasiswaBerprestasi15();

        for (int i = 0; i < 5; i++) {
            System.out.println("Masukkan Mahasiswa ke-" + (i+1) +":");
            System.out.print("NIM: ");
            String nim = sc.nextLine();
            System.out.print("Nama: ");
            String nama = sc.nextLine();
            System.out.print("Kelas: ");
            String kelas = sc.nextLine();
            System.out.print("IPK: ");
            double ipk = sc.nextDouble();
            System.out.println("--------------------------------");
```

```
            sc.nextLine();

            Mahasiswa15 m = new Mahasiswa15(nim, nama, kelas, ipk);
            list.tambah(m);

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

        System.out.println("Data mahasiswa setelah sorting berdasarkan IPK
(DESC): ");
        list.bubbleSort();
        list.tampil();

        sc.close();
    }

}
```

HASIL RUN





MENGURUTKAN DATA

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

```
        void selectionSort(){
             for (int i=0;i < listMhs.length; i++){
                 int idxMin = i;
                 for (int j= i + 1;j < listMhs.length; j++) {
                     if (listMhs[j].ipk<listMhs[idxMin].ipk) {
                         idxMin = j;
                     }
                 }
                 Mahasiswa15 tmp = listMhs[idxMin];
                 listMhs[idxMin] = listMhs[i];
                 listMhs[i] = tmp;

             }
          }
        }
```

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

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

Hasil run

```
Masukkan Mahasiswa ke-1:
NIM: 9302
Nama: lidya
Kelas: 2w
IPK: 4.0
--------------------------------
Masukkan Mahasiswa ke-2:
NIM: 9314
Nama: kia
Kelas: 2d
IPK: 3.6
--------------------------------
Masukkan Mahasiswa ke-3:
NIM: 9031
Nama: koko
Kelas: 2a
IPK: 2.5
--------------------------------
Masukkan Mahasiswa ke-4:
NIM: 09310
Nama: rere
Kelas: 2f
IPK: 3.7
--------------------------------
Masukkan Mahasiswa ke-5:
NIM: 8321
Nama: lala
Kelas: 2i
IPK: 3.3
--------------------------------
Data mahasiswa sebelum sorting:
Nama : lidya
NIM : 9302
Kelas : 2w
ipk : 4.0
----------------------------
Nama : kia
NIM : 9314
Kelas : 2d
ipk : 3.6
```

```
Nama : koko
NIM : 9031
Kelas : 2a
ipk : 2.5
----------------------------
Nama : rere
NIM : 09310
Kelas : 2f
ipk : 3.7
----------------------------
Nama : lala
NIM : 8321
Kelas : 2i
ipk : 3.3
----------------------------
Data mahasiswa setelah sorting berdasarkan IPK (DESC):
Nama : lidya
NIM : 9302
Kelas : 2w
ipk : 4.0
----------------------------
Nama : rere
NIM : 09310
Kelas : 2f
ipk : 3.7
----------------------------
Nama : kia
NIM : 9314
Kelas : 2d
ipk : 3.6
----------------------------
Nama : lala
NIM : 8321
Kelas : 2i
ipk : 3.3
----------------------------
Nama : koko
NIM : 9031
Kelas : 2a
ipk : 2.5
```

e

```
----------------------------
data yang sudah terurut menggunakan SELECTION SORT(ASC)
Nama : koko
NIM : 9031
Kelas : 2a
ipk : 2.5
----------------------------
Nama : lala
NIM : 8321
Kelas : 2i
ipk : 3.3
----------------------------
Nama : kia
NIM : 9314
Kelas : 2d
ipk : 3.6
----------------------------
Nama : rere
NIM : 09310
Kelas : 2f
ipk : 3.7
----------------------------
Nama : lidya
NIM : 9302
Kelas : 2w
ipk : 4.0
----------------------------
```

Pertanyaan:

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

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

Untuk apakah proses tersebut, jelaskan!

Jawab: Kode tersebut berfungsi untuk menemukan posisi elemen dengan IPK terkecil dari indeks ke-i hingga akhir daftar. Variabel idxMin digunakan untuk menyimpan indeks elemen dengan IPK terkecil selama proses pencarian berlangsung. Proses ini merupakan langkah penting dalam algoritma selection sort untuk menentukan elemen dengan nilai terkecil.

MENGURUTKAN DATA MAHASISWA BERDASARKAN IPK MENGGUNAKAN INSERTION SORT

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

```
        void interionSort(){
             for (int i = 1; i < listMhs.length; i++){
                 Mahasiswa15 temp = listMhs[i];
                 int j = i;
                 while (j > 0 && listMhs[j-1].ipk>temp.ipk){
                     listMhs[j] = listMhs[j - 1];
                     j--;
                 }
                 listMhs[j] = temp;
             }
    }
```

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

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

kode program

```
Data yang sudah terurut menggunakan INSERTION SORT (ASC)
Nama : sila
NIM : 93498
Kelas : 3d
ipk : 2.7
--------------------------
Nama : opi
NIM : 01903
Kelas : 2a
ipk : 2.8
--------------------------
Nama : lili
NIM : 3109
Kelas : 2w
ipk : 3.3
--------------------------
Nama : popi
NIM : 1839370
Kelas : 1t
ipk : 4.0
--------------------------
Nama : kaka
NIM : 92589
Kelas : 1e
ipk : 4.0
--------------------------
```

Pertanyaan

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

```
void insertionSort(){

        for (int i = 1; i < listMhs.length; i++){

            Mahasiswa15 temp = listMhs[i];

            int j = i;

            while (j > 0 && listMhs[j-1].ipk < temp.ipk){

                listMhs[j] = listMhs[j - 1];

                j--;

            }

            listMhs[j] = temp;

        }

    }
```

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

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

```
      public class Dosen15 {
          String kode;
          String nama;
          boolean jenisKelamin;
          int usia;

          Dosen15(String kd, String name, boolean jk, int age) {
              kode = kd;
              nama = name;
              jenisKelamin = jk;
              usia = age;
          }

          void tampil() {
              System.out.println("Kode           : " + kode);
              System.out.println("Nama           : " + nama);
              System.out.println("Jenis Kelamin  : " + (jenisKelamin ?
      "Laki-laki" : "Perempuan"));
              System.out.println("Usia           : " + usia);
              System.out.println("---------------------------------");
          }
}
```

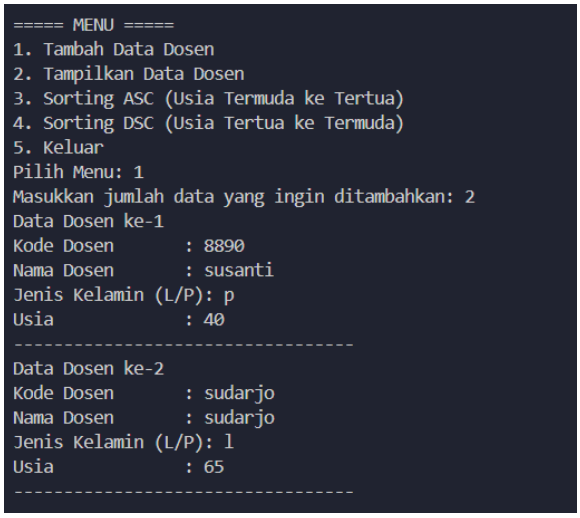
```
public class DataDosen15 {
    Dosen15[] dataDosen = new Dosen15[10];
    int idx;

    void tambah(Dosen15 dsn) {
        if (idx < dataDosen.length) {
            dataDosen[idx++] = dsn;
        } else {
            System.out.println("Data Dosen Sudah Penuh!");
        }
    }

    void tampil() {
        if (idx == 0) {
            System.out.println("Tidak Ada Data Dosen.");
            return;
        }
        for (int i = 0; i < idx; i++) {
            dataDosen[i].tampil();
        }
    }

    void sortingASC() {
        for (int i = 0; i < idx - 1; i++) {
            for (int j = 0; j < idx - 1 - i; j++) {
                if (dataDosen[j].usia > dataDosen[j + 1].usia) {
                    Dosen15 temp = dataDosen[j];
                    dataDosen[j] = dataDosen[j + 1];
                    dataDosen[j + 1] = temp;
                }
            }
        }
    }

    void sortingDSC() {
        for (int i = 0; i < idx - 1; i++) {
            for (int j = 0; j < idx - 1 - i; j++) {
                if (dataDosen[j].usia < dataDosen[j + 1].usia) {
                    Dosen15 temp = dataDosen[j];
                    dataDosen[j] = dataDosen[j + 1];
                    dataDosen[j + 1] = temp;
                }
            }
        }
    }
}
```

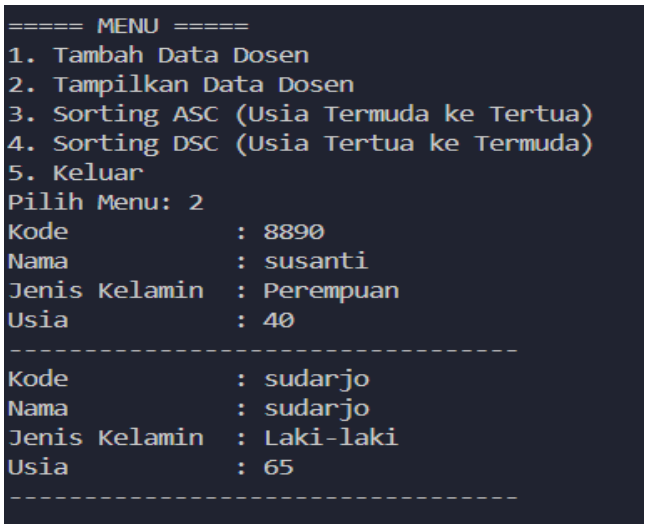
```
import java.util.Scanner;

public class DosenMain15 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        DataDosen15 datadsn = new DataDosen15();
        int pilihan;

        do {
```

```
            System.out.println("\n===== MENU =====");
            System.out.println("1. Tambah Data Dosen");
            System.out.println("2. Tampilkan Data Dosen");
            System.out.println("3. Sorting ASC (Usia Termuda ke
Tertua)");
            System.out.println("4. Sorting DSC (Usia Tertua ke
Termuda)");
            System.out.println("5. Keluar");
            System.out.print("Pilih Menu: ");
            pilihan = sc.nextInt();
            sc.nextLine();

            switch (pilihan) {
                case 1:
                    if (datadsn.idx >= 10) {
                        System.out.println("Kapasitas data sudah
penuh.");
                        break;
                    }
                    System.out.print("Masukkan jumlah data yang ingin
ditambahkan: ");
                    int jumlah = sc.nextInt();
                    sc.nextLine();

                    for (int i = 0; i < jumlah; i++) {
                        if (datadsn.idx >= 10) {
                            System.out.println("Kapasitas data sudah
penuh.");
                            break;
                        }
                        System.out.println("Data Dosen ke-" +
(datadsn.idx + 1));
                        System.out.print("Kode Dosen       : ");
                        String kode = sc.nextLine();
                        System.out.print("Nama Dosen       : ");
                        String nama = sc.nextLine();
                        System.out.print("Jenis Kelamin (L/P): ");
                        char jk = sc.next().toUpperCase().charAt(0);
                        boolean jenisKelamin = jk == 'L';
                        System.out.print("Usia             : ");
                        int usia = sc.nextInt();
                        sc.nextLine();

                        Dosen15 dsn = new Dosen15(kode, nama,
jenisKelamin, usia);
                        datadsn.tambah(dsn);
                        System.out.println("-------------------------
-----");
                    }
                    break;

                case 2:
                    datadsn.tampil();
                    break;

                case 3:
                    datadsn.sortingASC();
                    System.out.println("Data Dosen Diurutkan (Termuda ke
Tertua):");
```

```
                    datadsn.tampil();
                    break;

                case 4:
                    datadsn.sortingDSC();
                    System.out.println("Data Dosen Diurutkan (Tertua ke
Termuda):");
                    datadsn.tampil();
                    break;

                case 5:
                    System.out.println("Keluar dari program.");
                    break;

                default:
                    System.out.println("Pilihan tidak valid. Coba
lagi.");
            }

        } while (pilihan != 5);

        sc.close();
    }
}
```

Hasil run





```
===== MENU =====
1. Tambah Data Dosen
2. Tampilkan Data Dosen
3. Sorting ASC (Usia Termuda ke Tertua)
4. Sorting DSC (Usia Tertua ke Termuda)
5. Keluar
Pilih Menu: 3
Data Dosen Diurutkan (Termuda ke Tertua):
Kode           : 8890
Nama           : susanti
Jenis Kelamin  : Perempuan
Usia           : 40
---------------------------------
Kode           : sudarjo
Nama           : sudarjo
Jenis Kelamin  : Laki-laki
Usia           : 65
---------------------------------
```

```
===== MENU =====
1. Tambah Data Dosen
2. Tampilkan Data Dosen
3. Sorting ASC (Usia Termuda ke Tertua)
4. Sorting DSC (Usia Tertua ke Termuda)
5. Keluar
Pilih Menu: 4
Data Dosen Diurutkan (Tertua ke Termuda):
Kode           : sudarjo
Nama           : sudarjo
Jenis Kelamin  : Laki-laki
Usia           : 65
---------------------------------
Kode           : 8890
Nama           : susanti
Jenis Kelamin  : Perempuan
Usia           : 40
---------------------------------

===== MENU =====
1. Tambah Data Dosen
2. Tampilkan Data Dosen
3. Sorting ASC (Usia Termuda ke Tertua)
4. Sorting DSC (Usia Tertua ke Termuda)
5. Keluar
Pilih Menu: 5
Keluar dari program.
```