ALGORITMA DAN STRUKTUR DATA

“Laporan hasil Praktikum pada Jobsheet 6 “SORTING (BUBBLE, SELECTION, DAN INSERTION SORT” ”

Oleh:

Hafiz Rizqi Hernanda

NIM (244107020154)

Jurusan Teknologi informasi

Teknik Informatika

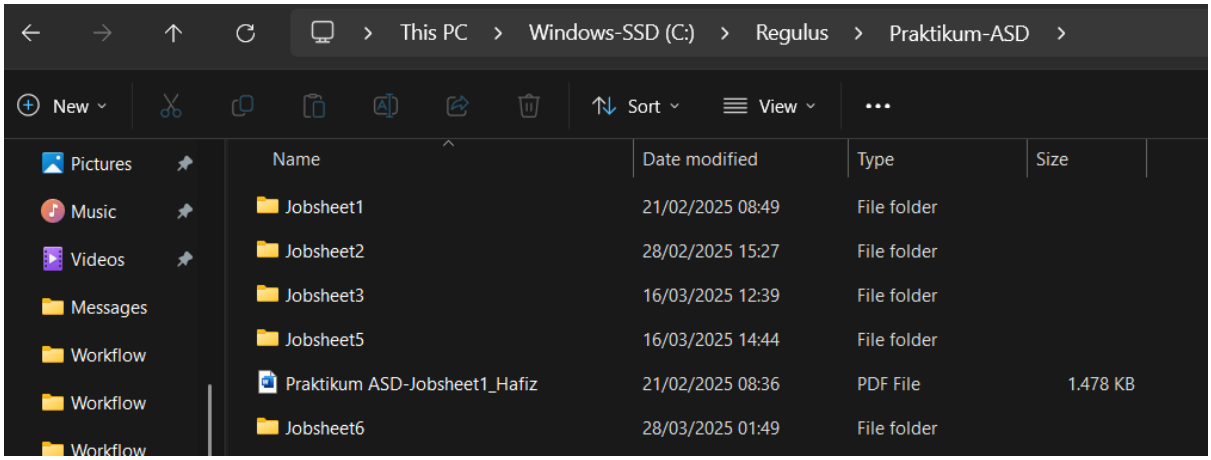
Politeknik Negeri Malang

6.6 Praktikum 1 - Mengimplementasikan Sorting menggunakan object

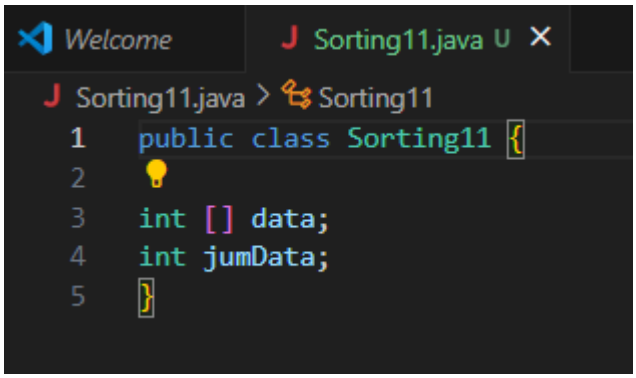
6.2.1 Langkah Praktikum 1

a. SORTING – BUBBLE SORT

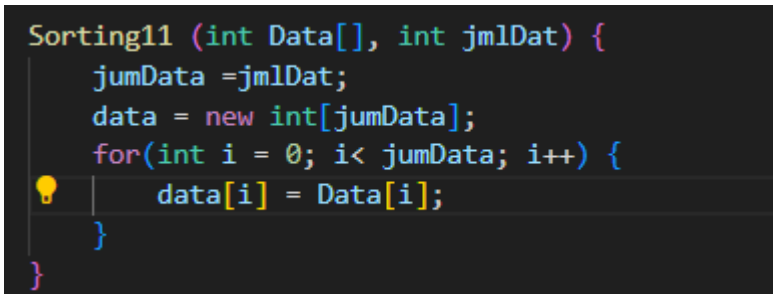
1. Buat folder baru bernama Jobsheet6 di dalam repository Praktikum ASD



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



3. Buatlah konstruktor dengan parameter Data[] dan jmlDat



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 kemudian deklarasikan array dengan nama a[] kemudian isi array tersebut

```
Sorting11.java U    SortingMain11.java 1, U
SortingMain11.java > SortingMain11 > main(String[])
1  public class SortingMain11 {
       Run | Debug
2      public static void main(String[] args) {
3          int a[] = {20, 10, 2, 7, 12};
4      }
5
6  }
7
```

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

```
Sorting11 dataurut1 = new Sorting11(a, a.length);
```

8. Lakukan pemanggilan method bubbleSort dan tampil

```
System.out.println(x:"Data awal 1");
dataurut1.tampil();
dataurut1.bubbleSort();
System.out.println(x:"Data sudah diurutkan dengan BUBBLE SORT (ASC)");
dataurut1.tampil();
```

9. Jalankan program, dan amati hasilnya!

6.2.2 Verifikasi Hasil Percobaan

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

```
Data awal 1
20 10 2 7 12
Data sudah diurutkan dengan BUBBLE SORT (ASC)
2 7 10 12 20
PS C:\Regulus\Praktikum-ASD\Jobsheet6>
```

b. SORTING – SELECTION SORT

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

```
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[i];
        data[i]=data[min];
        data[min]=temp;
    }
}
```

2. Deklarasikan array dengan nama b[] pada kelas SortingMain kemudian isi array tersebut

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

3. Buatlah objek baru dengan nama dataurut2 yang merupakan instansiasi dari class Sorting, kemudian isi parameternya

```
Sorting11 dataurut2 = new Sorting11(b, b.length);
```

4. Lakukan pemanggilan method SelectionSort dan tampil

```
System.out.println(x:"Data awal 2");
dataurut2.tampil();
dataurut2.bubbleSort();
System.out.println(x:"Data sudah diurutkan dengan SELECTION SORT (ASC)");
dataurut2.tampil();
```

5. Jalankan program dan amati hasilnya!

6.2.3 Verifikasi Hasil Percobaan

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

```
Data awal 2
30 20 2 8 14
Data sudah diurutkan dengan SELECTION SORT (ASC)
2 8 14 20 30
PS C:\Regulus\Praktikum-ASD\Jobsheet6>
```

c. SORTING – INSERTION SORT

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

```
void insertionSort() {
    for(int i = 0; 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 kemudian isi array tersebut

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

3. Buatlah objek baru dengan nama dataurut3 yang merupakan instansiasi dari class Sorting, kemudian isi parameternya

```
Sorting11 dataurut3 = new Sorting11(c, c.length);
```

4. Lakukan pemanggilan method insertionSort dan tampil

```
System.out.println(x:"Data awal 3");
dataurut3.tampil();
dataurut3.bubbleSort();
System.out.println(x:"Data sudah diurutkan dengan SELECTION SORT (ASC)");
dataurut3.tampil();
```

5. Jalankan program dan amati hasilnya!

6.2.4 Verifikasi Hasil Percobaan

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

```
Data awal 3
40 10 4 9 3
Data sudah diurutkan dengan SELECTION SORT (ASC)
3 4 9 10 40
PS C:\Regulus\Praktikum-ASD\Jobsheet6>
```

6.2.5 Pertanyaan!

1. Jelaskan fungsi kode program berikut

```
if (data[j-1]>data[j]){
    temp=data[j];
    data[j]=data[j-1];
    data[j-1]=temp;
}
```

Kode program itu bagian dari algoritma *Bubble Sort*. Fungsinya adalah menukar posisi dua elemen dalam array jika elemen sebelumnya (data[j-1]) lebih besar dari elemen saat ini (data[j]). Berarti jika elemen yang berada di indeks sebelumnya lebih besar, maka dilakukan pertukaran untuk memastikan elemen yang lebih kecil berada di depan.

2. Tunjukkan kode program yang merupakan algoritma pencarian nilai minimum pada selection sort!

```
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[i];
    data[i]=data[min];
    data[min]=temp;
```

3. Pada Insertion sort , jelaskan maksud dari kondisi pada perulangan

```
while (j>=0 && data[j]>temp)
```

Kondisi ini digunakan untuk menggeser elemen yang lebih besar ke kanan hingga menemukan posisi yang tepat untuk elemen temp.

- j >= 0 memastikan bahwa tidak terjadi akses indeks negatif.
- data[j] > temp berarti jika elemen saat ini lebih besar dari elemen yang sedang disisipkan, maka elemen akan digeser ke kanan.

4. Pada Insertion sort, apakah tujuan dari perintah

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

Perintah ini bertujuan untuk menggeser elemen data[j] ke kanan agar tersedia ruang bagi elemen yang akan disisipkan dalam posisi yang benar.

6.7 Praktikum 2- (Sorting Menggunakan Array of Object)

6.3.1 Langkah Praktikum 2 - Mengurutkan Data Mahasiswa Berdasarkan IPK (Bubble Sort)

Perhatikan diagram class Mahasiswa di bawah ini! Diagram class ini yang selanjutnya akan dibuat sebagai acuan dalam membuat kode program class Mahasiswa.

Mahasiswa
nim: String nama: String kelas: String ipk: double
Mahasiswa() Mahasiswa(nm: String, name: String, kls: String, ip: double) tampilInformasi(): void

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

MahasiswaBerprestasi
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.

6.3.2 Langkah-langkah Praktikum 2

1. Buatlah class dengan nama Mahasiswa.

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

```java
Mahasiswa11.java > Mahasiswa11 > tampilInformasi()
public class Mahasiswa11 {
    String nim;
    String nama;
    String kelas;
    Double ipk;

    //Kontruksi Default
    Mahasiswa11() {

    }

    //Kontruksi Berparameter
    Mahasiswa11(String nm, String name, String kls, double ip) {
        nim = nm;
        nama = name;
        ipk = ip;
        kelas = kls;
    }

    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 seperti di bawah ini!

```java
MahasiswaBerprestasi11.java > MahasiswaBerprestasi11 > bubbleSort()
public class MahasiswaBerprestasi11 {
    Mahasiswa11 [] listMhs = new Mahasiswa11[5];
    int idx;
```

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

```java
void tambah(Mahasiswa11 m) {
    if (idx<listMhs.length) {
     listMhs[idx] = m;
     idx++;

    }else {
     System.out.println(x:"data sudah penuh");
    }

}
```

5. 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.

```
void tampil() {
    for (Mahasiswa11 m:listMhs) {
        m.tampilInformasi();
        System.out.println(x:"----------------------------");
    }
}
```

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

```
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) {
                Mahasiswa11 tmp = listMhs[j];
                listMhs[j]=listMhs[j-1];
                listMhs[j-1]=tmp;
            }
        }
    }
}
```

7. Buat class MahasiswaDemo, 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.

```java
MahasiswaDemo11.java > MahasiswaDemo11 > main(String[])
public class MahasiswaDemo11 {
    Run | Debug
    public static void main(String[] args) {
        MahasiswaBerprestasi11 list = new MahasiswaBerprestasi11();
        Mahasiswa11 m1 = new Mahasiswa11(nm:"123", name:"Zidan", kls:"2A", ip:3.2);
        Mahasiswa11 m2 = new Mahasiswa11(nm:"124", name:"Ayu", kls:"2A", ip:3.5);
        Mahasiswa11 m3 = new Mahasiswa11(nm:"125", name:"Sofi", kls:"2A", ip:3.1);
        Mahasiswa11 m4 = new Mahasiswa11(nm:"126", name:"Sita", kls:"2A", ip:3.9);
        Mahasiswa11 m5 = new Mahasiswa11(nm:"127", name:"Miki", kls:"2A", ip:3.7);

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

6.3.3 Verifikasi Hasil Percobaan

Cocokan hasilnya dengan yang terdapat pada tampilan di bawah ini

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

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

```
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
--------------------------
PS C:\Regulus\Praktikum-ASD\Jobsheet6>
```

6.3.4 Pertanyaan

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

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

   a. Mengapa syarat dari perulangan i adalah i<listMhs.length-1 ?
      Perulangan i mengontrol jumlah iterasi utama dalam *Bubble Sort* dan Proses *Bubble Sort* membutuhkan maksimal n-1 iterasi untuk memastikan semua elemen sudah tersusun dengan benar. Oleh karena itu, batas atas i adalah listMhs.length - 1 agar tidak melakukan iterasi yang tidak perlu.
   b. Mengapa syarat dari perulangan j adalah j<listMhs.length-I ?
      Perulangan j digunakan untuk membandingkan elemen dan melakukan pertukaran jika diperlukan dan setiap iterasi i membuat elemen terbesar pada bagian yang belum terurut berpindah ke posisi yang benar. Oleh karena itu, perulangan j tidak perlu menelusuri elemen yang sudah terurut, sehingga batasnya berkurang sebanyak i.
   c. Jika banyak data di dalam listMhs adalah 50, maka berapakali perulangan i akan berlangsung? Dan ada berapa Tahap bubble sort yang ditempuh?
      Perulangan i akan berlangsung sebanyak listMhs.length - 1 = 50 - 1 = 49 kali dan bubble Sort bekerja dalam n-1 tahap, jadi dalam hal ini terdapat 49 tahap untuk mengurutkan seluruh elemen.

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

MahasiswaDemo11.java > MahasiswaDemo11 > main(String[])

```java
import java.util.Scanner;
public class MahasiswaDemo11 {
    Run | Debug
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        MahasiswaBerprestasi11 list = new MahasiswaBerprestasi11();

        System.out.print(s:"Masukkan jumlah Mahasiswa: ");
        int jmlMhs = input.nextInt();
        input.nextLine();

        for (int i = 0; i < jmlMhs; i++) {
            System.out.println("Masukkan data mahasiswa ke-" + (i + 1) + ": ");
            System.out.print(s:"NIM: ");
            String nim = input.nextLine();
            System.out.print(s:"Nama: ");
            String nama = input.nextLine();
            System.out.print(s:"Kelas: ");
            String kelas = input.nextLine();
            System.out.print(s:"IPK: ");
            double ipk = input.nextDouble();
            input.nextLine();

            Mahasiswa11 mhs = new Mahasiswa11(nim, nama, kelas, ipk);
            list.tambah(mhs);
        }

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

        System.out.println(x:"Data Mahasiswa setelah sorting berdasarkan IPK (DESC) : ");
        list.bubbleSort();
        list.tampil();
    }
}
```

6.4 Mengurutkan Data Mahasiswa Berdasarkan IPK (Selection Sort)

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.

6.4.1. Langkah-langkah Percobaan.

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-1;i++) {
        int idxMin =i;
        for ( int j = i+1; j<listMhs.length;j++) {
            if (listMhs[j].ipk < listMhs[idxMin].ipk) {
                idxMin = j;
            }
        }
        Mahasiswa11 tmp = listMhs[idxMin];
        Mahasiswa11[] listMhs
        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(x:"Data yang sudah terurut menggunakan SELECTION SORT (ASC)");
list.selectionSort();
list.tampil();
```

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

6.4.2 Verifikasi Hasil Percobaan

Pastikan output yang ditampilkan sudah benar seperti di bawah ini

```
Masukkan Data Mahasiswa ke-1
NIM   : 123
Nama  : Ali
Kelas : 2B
IPK   : 3.9
------------------------------
Masukkan Data Mahasiswa ke-2
NIM   : 124
Nama  : ila
Kelas : 2B
IPK   : 3.1
------------------------------
```

```
Masukkan Data Mahasiswa ke-3
NIM   : 125
Nama  : agus
Kelas : 2B
IPK   : 3.6
------------------------------
Masukkan Data Mahasiswa ke-4
NIM   : 126
Nama  : tika
Kelas : 2B
IPK   : 3.3
------------------------------
Masukkan Data Mahasiswa ke-5
NIM   : 127
Nama  : udin
Kelas : 2B
IPK   : 3.2
------------------------------
Data yang sudah terurut menggunakan SELECTION SORT (ASC)
Nama: ila
NIM: 124
Kelas: 2B
IPK: 3.1
----------------------------
Nama: udin
NIM: 127
Kelas: 2B
IPK: 3.2
----------------------------
Nama: tika
NIM: 126
Kelas: 2B
IPK: 3.3
----------------------------
Nama: agus
NIM: 125
Kelas: 2B
IPK: 3.6
----------------------------
Nama: Ali
NIM: 123
Kelas: 2B
IPK: 3.9
----------------------------
```

```
Masukkan jumlah Mahasiswa: 5
Masukkan data mahasiswa ke-1:
NIM: 123
Nama: Ali
Kelas: 2B
IPK: 3,9
Masukkan data mahasiswa ke-2:
NIM: 124
Nama: ila
Kelas: 2B
IPK: 3,1
Masukkan data mahasiswa ke-3:
NIM: 125
Nama: agus
Kelas: 2B
IPK: 3,6
Masukkan data mahasiswa ke-4:
NIM: 126
Nama: tika
Kelas: 2B
IPK: 3,3
Masukkan data mahasiswa ke-5:
NIM: 127
Nama: udin
Kelas: 2B
IPK: 3,2
```

```
Data yang sudah terurut menggunakan SELECTION SORT (ASC)
Nama: ila
NIM: 124
Kelas: 2B
IPK: 3.1
--------------------------
Nama: udin
NIM: 127
Kelas: 2B
IPK: 3.2
--------------------------
Nama: tika
NIM: 126
Kelas: 2B
IPK: 3.3
--------------------------
Nama: agus
NIM: 125
Kelas: 2B
IPK: 3.6
--------------------------
Nama: Ali
NIM: 123
Kelas: 2B
IPK: 3.9
--------------------------
PS C:\Regulus\Praktikum-ASD\Jobsheet6>
```

6.4.3 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!

Di mulai dengan baris int idxMin = i, menyimpan indeks sementara dari elemen dengan IPK terkecil dalam iterasi ini. Awalnya, indeks i dianggap sebagai nilai minimum.

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

if (listMhs[j].ipk < listMhs[idxMin].ipk) {

idxMin = j;

}

}

Perulangan j berjalan mulai dari indeks **i + 1** hingga akhir array (listMhs.length). Jika ditemukan elemen dengan **IPK lebih kecil**, maka idxMin diperbarui dengan indeks elemen tersebut. Setelah perulangan selesai, idxMin akan menyimpan **indeks elemen dengan IPK terkecil** di antara elemen yang tersisa.
Kode ini berfungsi untuk **menentukan indeks elemen dengan IPK terkecil** dalam sub-array yang belum terurut, bisa di sebut ascending.

6.5 Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan Insertion Sort
Yang terakhir akan diimplementasikan Teknik sorting menggunakan Insertion Sort, dengan mengurutkan IPK mahasiswa secara ascending
6.5.1 Langkah-langkah Percobaan
1. Lihat kembali class MahasiswaBerprestasi, 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 < listMhs.length; i++) {
        Mahasiswa11 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!

3. Coba jalankan kembali class MahasiswaDemo, dan amati hasilnya! Apakah kini data mahasiswa telah tampil urut menaik berdasar ipk?
6.5.2 Verifikasi Hasil Percobaan
Pastikan output yang ditampilkan sudah benar seperti di bawah ini

```
Masukkan Data Mahasiswa ke-1
NIM   : 111
Nama  : ayu
Kelas : 2c
IPK   : 3.7
------------------------------
Masukkan Data Mahasiswa ke-2
NIM   : 222
Nama  : dika
Kelas : 2c
IPK   : 3.0
------------------------------
Masukkan Data Mahasiswa ke-3
NIM   : 333
Nama  : ila
Kelas : 2c
IPK   : 3.8
------------------------------
Masukkan Data Mahasiswa ke-4
NIM   : 444
Nama  : susi
Kelas : 2c
IPK   : 3.1
------------------------------
Masukkan Data Mahasiswa ke-5
NIM   : 555
Nama  : yayuk
Kelas : 2c
IPK   : 3.4
------------------------------
```

```
Data yang sudah terurut menggunakan INSERTION SORT (ASC)
Nama: dika
NIM: 222
Kelas: 2c
IPK: 3.0
----------------------------
Nama: susi
NIM: 444
Kelas: 2c
IPK: 3.1
----------------------------
Nama: yayuk
NIM: 555
Kelas: 2c
IPK: 3.4
----------------------------
Nama: ayu
NIM: 111
Kelas: 2c
IPK: 3.7
----------------------------
Nama: ila
NIM: 333
Kelas: 2c
IPK: 3.8
----------------------------
```

```
Masukkan jumlah Mahasiswa: 5
Masukkan data mahasiswa ke-1:
NIM: 111
Nama: ayu
Kelas: 2c
IPK: 3,7
Masukkan data mahasiswa ke-2:
NIM: 222
Nama: dika
Kelas: 2c
IPK: 3,0
Masukkan data mahasiswa ke-3:
NIM: 333
Nama: ila
Kelas: 2c
IPK: 3,8
Masukkan data mahasiswa ke-4:
NIM: 444
Nama: susi
Kelas: 2c
IPK: 3,1
Masukkan data mahasiswa ke-5:
NIM: 555
Nama: yayuk
Kelas: 2c
IPK: 3,4
```

```
Data yang terurut menggunakan INSERTION SORT (ASC)
Nama: dika
NIM: 222
Kelas: 2c
IPK: 3.0
--------------------------
Nama: susi
NIM: 444
Kelas: 2c
IPK: 3.1
--------------------------
Nama: yayuk
NIM: 555
Kelas: 2c
IPK: 3.4
--------------------------
Nama: ayu
NIM: 111
Kelas: 2c
IPK: 3.7
--------------------------
Nama: ila
NIM: 333
Kelas: 2c
IPK: 3.8
--------------------------
```

6.5.3 Pertanyaan

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

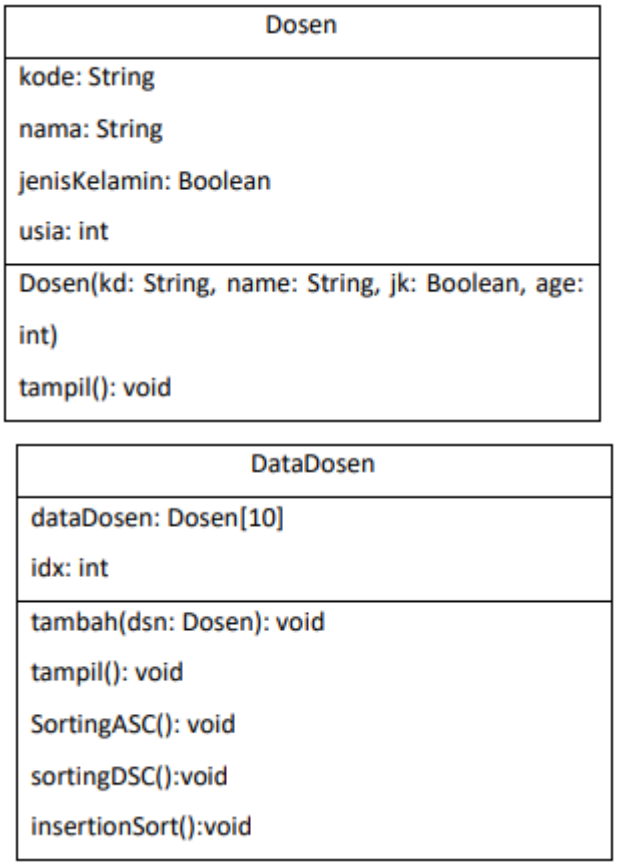
```
void insertionSort() {
    for (int i = 1; i < listMhs.length; i++) {
        Mahasiswa11 temp = listMhs[i];
        int j = i;
        while (j>0 && listMhs[j-1].ipk<temp.ipk) {
            listMhs[j] = listMhs[j-1];
            j--;
        }
        listMhs[j]=temp;
    }
}
```

Perubahan ada pada sebelumnya while(j>0 && listMhs[j-1].ipk>temp.ipk) menjadi while(j>0 && listMhs[j-1].ipk<temp.ipk)

```
Data yang terurut menggunakan INSERTION SORT (ASC)
Nama: ila
NIM: 333
Kelas: 2c
IPK: 3.8
---------------------------
Nama: ayu
NIM: 111
Kelas: 2c
IPK: 3.7
---------------------------
Nama: yayuk
NIM: 555
Kelas: 2c
IPK: 3.4
---------------------------
Nama: susi
NIM: 444
Kelas: 2c
IPK: 3.1
---------------------------
Nama: dika
NIM: 222
Kelas: 2c
IPK: 3.0
---------------------------
PS C:\Regulus\Praktikum-ASD\Jobsheet6>
```

6.6 Latihan Praktikum

Perhatikan class diagram dibawah ini:

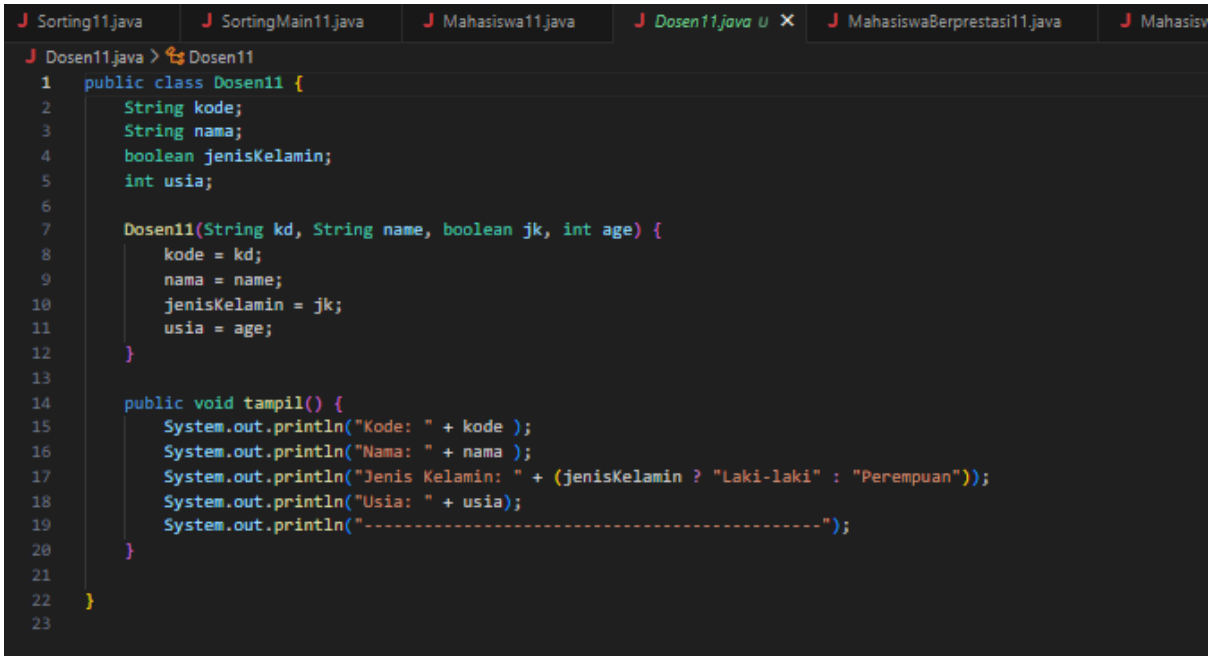


Berdasarkan class diagram diatas buatlah menu dikelas main dengan pilihan menu:

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.

DataDosen11.java > DataDosen11

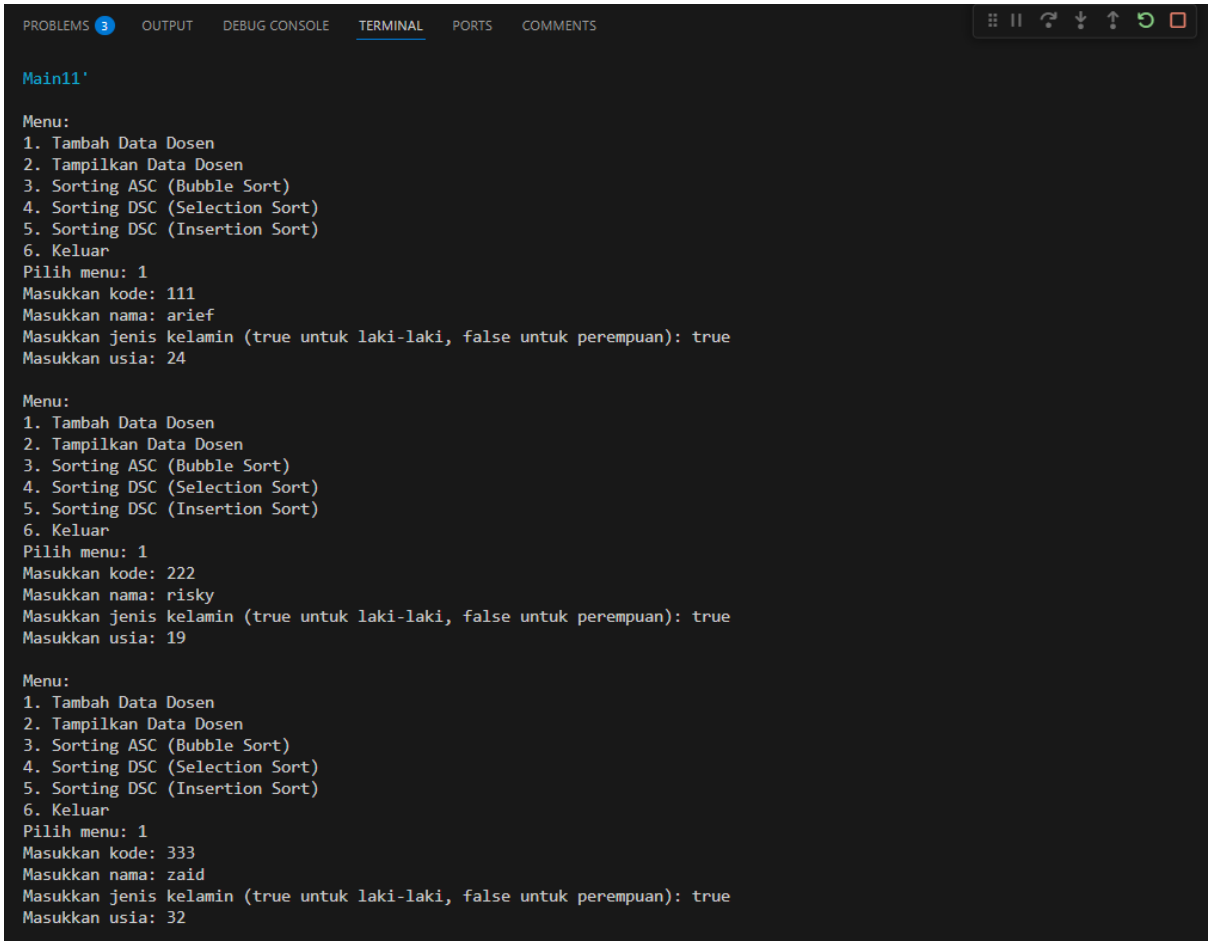
```java
public class DataDosen11 {
 Dosen11[] data11 = new Dosen11[10];
 int idx;

 void tambah(Dosen11 dsn) {
    if (idx < data11.length) {
        data11[idx] = dsn;

        idx++;
    } else {
        System.out.println("Data sudah penuh!");
    }
}

  void tampil() {
    for (int i = 0; i < idx; i++) {
        data11[i].tampil();
    }
}
void SortingASC() {
    for (int i = 0; i < idx - 1; i++) {
        for (int j = 0; j < idx -  1 - i; j++) {
            if (data11[j].usia > data11[j + 1].usia) {

                Dosen11 temp = data11[j];

                data11[j] = data11[j + 1];
                data11[j + 1] = temp;
            }
        }
    }
    System.out.println("Data berhasil diurutkan secara ascending (usia termuda ke tertua).");
}

void SortingDSC() {
    for (int i = 0; i < idx - 1; i++) {
        int idxMax = i;
        for (int j = i + 1; j < idx; j++) {
            if (data11[j].usia > data11[idxMax].usia) {
                idxMax= j;
            }
        }
        Dosen11 temp = data11[i];
        data11[i] = data11[idxMax];
        data11[idxMax] = temp;
    }
    System.out.println("Data berhasil diurutkan secara descending (usia tertua ke termuda).");
}

void insertionSort() {
    for (int i = 1; i < idx; i++) {
        Dosen11 temp = data11[i];
        int j = i;
        while (j > 0 && data11[j - 1].usia < temp.usia) {
            data11[j] = data11[j - 1];
            j--;
        }
        data11[j] = temp;
    }
    System.out.println("Data berhasil diurutkan menggunakan Insertion Sort (usia tertua ke termuda).");
}

}
```

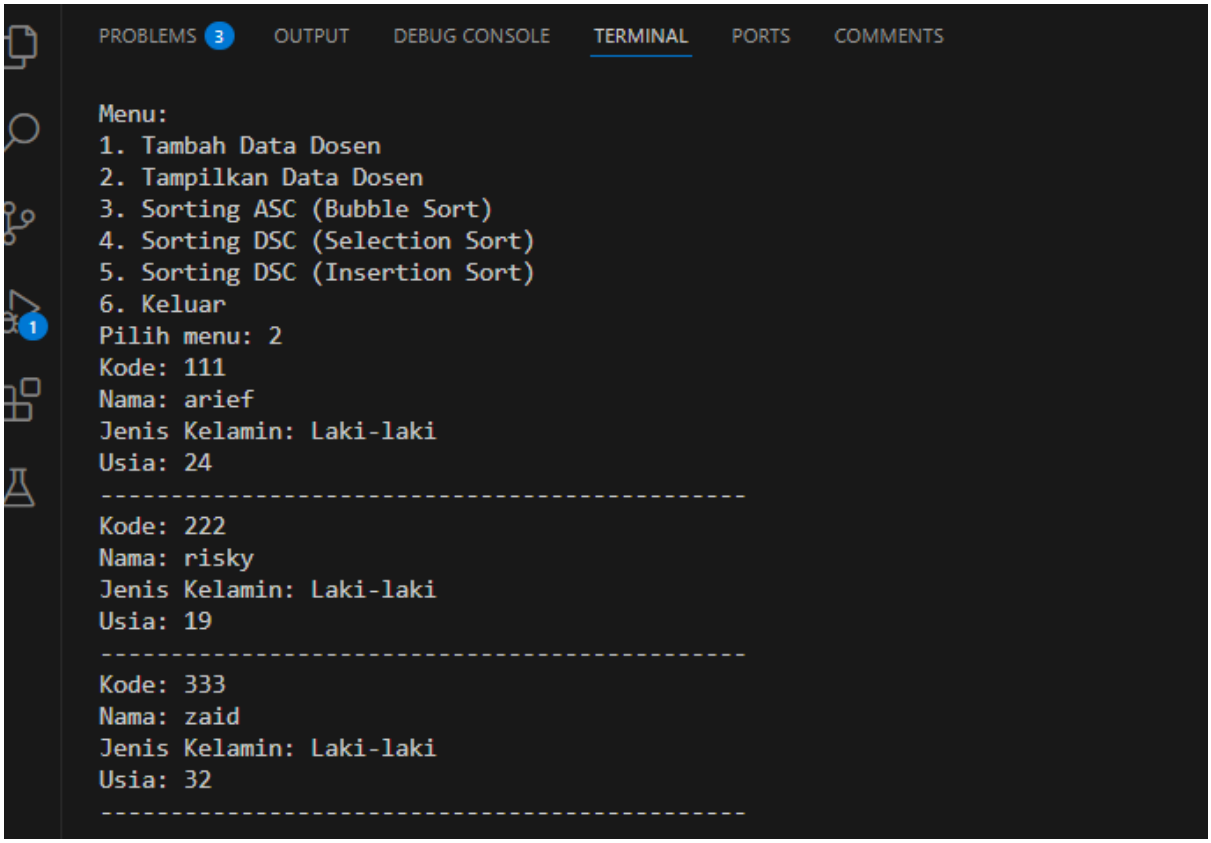

Sorting11.java
SortingMain11.java
Mahasiswa11.java
Dosen11.java U
MahasiswaBerprestasi11.java
Dosen11.java > Dosen11
public class Dosen11 {
String kode;
String nama;
boolean jenisKelamin;
int usia;
Dosen11(String kd, String name, boolean jk, int age) {
kode = kd;
nama = name;
jenisKelamin = jk;
usia = age;
}
public 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("---");
}
}

Sorting11.java | SortingMain11.java | Mahasiswa11.java | *DosenMain11.java U* | MahasiswaBerprestasi11.java | MahasiswaDemo11.java

DosenMain11.java > ...

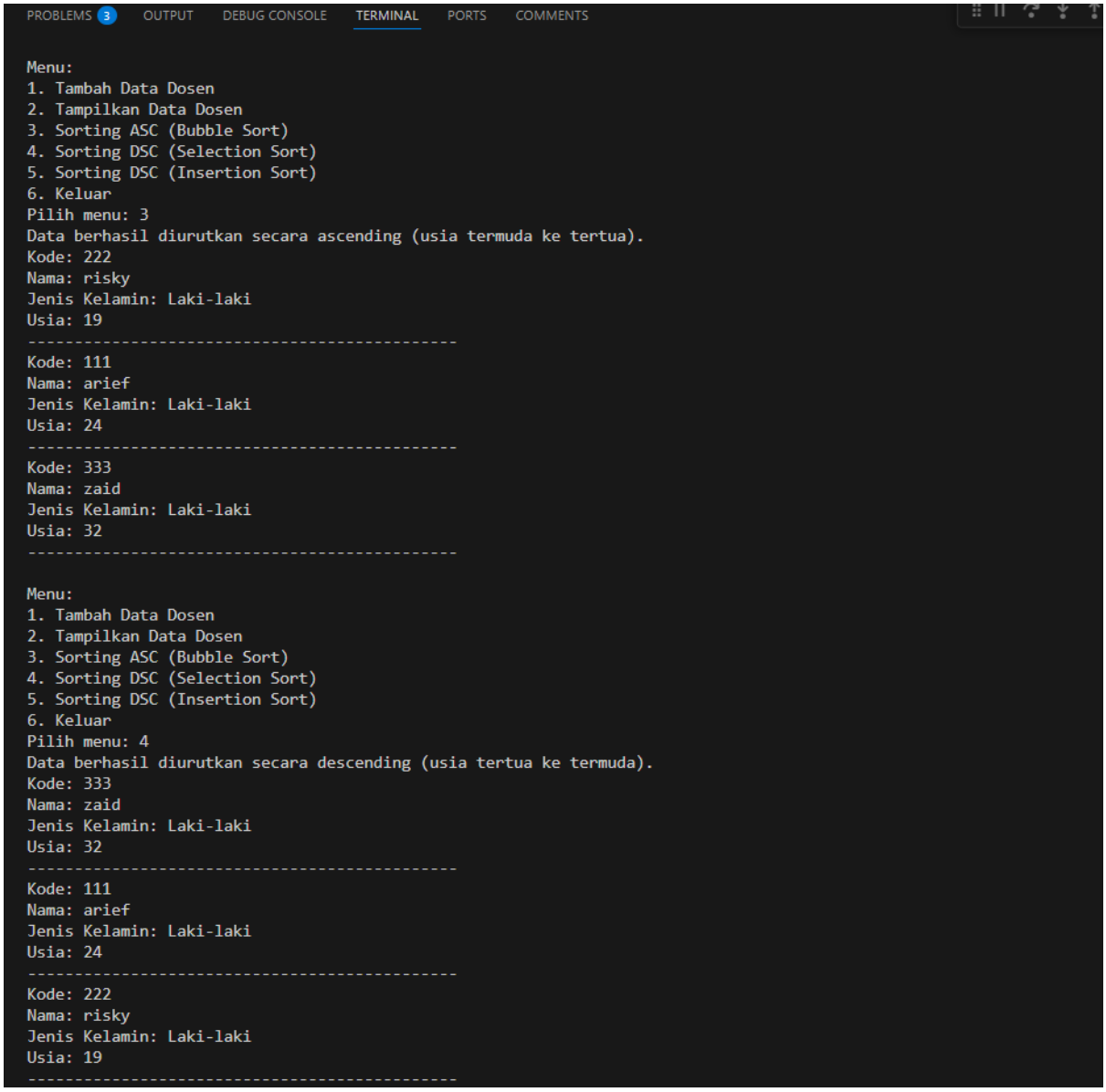
```java
import java.util.Scanner;
public class DosenMain11 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        DataDosen11 dsn = new DataDosen11();
        int pilihan;

        do {
            System.out.println("\nMenu:");
            System.out.println("1. Tambah Data Dosen");
            System.out.println("2. Tampilkan Data Dosen");
            System.out.println("3. Sorting ASC (Bubble Sort)");
            System.out.println("4. Sorting DSC (Selection Sort)");
            System.out.println("5. Sorting DSC (Insertion Sort)");
            System.out.println("6. Keluar");
            System.out.print("Pilih menu: ");
            pilihan = sc.nextInt();
            sc.nextLine();

            switch (pilihan) {
                case 1:
                    System.out.print("Masukkan kode: ");
                    String kode = sc.nextLine();
                    System.out.print("Masukkan nama: ");
                    String nama = sc.nextLine();
                    System.out.print("Masukkan jenis kelamin (true untuk laki-laki, false untuk perempuan): ");
                    boolean jk = sc.nextBoolean();
                    System.out.print("Masukkan usia: ");
                    int usia = sc.nextInt();
                    dsn.tambah(new Dosen11(kode, nama, jk, usia));
                    break;
                case 2:
                    dsn.tampil();
                    break;
                case 3:
                    dsn.SortingASC();
                    dsn.tampil();
                    break;
                case 4:
                    dsn.SortingDSC();
                    dsn.tampil();
                    break;
                case 5:
                    dsn.insertionSort();
                    dsn.tampil();
                    break;
                case 6:
                    System.out.println("Program selesai.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid!");
            }
        } while (pilihan != 6);

        sc.close();
    }

}
```


PROBLEMS 3
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
COMMENTS
Main11'
Menu:
1. Tambah Data Dosen
2. Tampilkan Data Dosen
3. Sorting ASC (Bubble Sort)
4. Sorting DSC (Selection Sort)
5. Sorting DSC (Insertion Sort)
6. Keluar
Pilih menu: 1
Masukkan kode: 111
Masukkan nama: arief
Masukkan jenis kelamin (true untuk laki-laki, false untuk perempuan): true
Masukkan usia: 24
Menu:
1. Tambah Data Dosen
2. Tampilkan Data Dosen
3. Sorting ASC (Bubble Sort)
4. Sorting DSC (Selection Sort)
5. Sorting DSC (Insertion Sort)
6. Keluar
Pilih menu: 1
Masukkan kode: 222
Masukkan nama: risky
Masukkan jenis kelamin (true untuk laki-laki, false untuk perempuan): true
Masukkan usia: 19
Menu:
1. Tambah Data Dosen
2. Tampilkan Data Dosen
3. Sorting ASC (Bubble Sort)
4. Sorting DSC (Selection Sort)
5. Sorting DSC (Insertion Sort)
6. Keluar
Pilih menu: 1
Masukkan kode: 333
Masukkan nama: zaid
Masukkan jenis kelamin (true untuk laki-laki, false untuk perempuan): true
Masukkan usia: 32


PROBLEMS 3
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
COMMENTS
Menu:
1. Tambah Data Dosen
2. Tampilkan Data Dosen
3. Sorting ASC (Bubble Sort)
4. Sorting DSC (Selection Sort)
5. Sorting DSC (Insertion Sort)
6. Keluar
Pilih menu: 2
Kode: 111
Nama: arief
Jenis Kelamin: Laki-laki
Usia: 24

Kode: 222
Nama: risky
Jenis Kelamin: Laki-laki
Usia: 19

Kode: 333
Nama: zaid
Jenis Kelamin: Laki-laki
Usia: 32


PROBLEMS 3
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
COMMENTS
Menu:
1. Tambah Data Dosen
2. Tampilkan Data Dosen
3. Sorting ASC (Bubble Sort)
4. Sorting DSC (Selection Sort)
5. Sorting DSC (Insertion Sort)
6. Keluar
Pilih menu: 3
Data berhasil diurutkan secara ascending (usia termuda ke tertua).
Kode: 222
Nama: risky
Jenis Kelamin: Laki-laki
Usia: 19

Kode: 111
Nama: arief
Jenis Kelamin: Laki-laki
Usia: 24

Kode: 333
Nama: zaid
Jenis Kelamin: Laki-laki
Usia: 32

Menu:
1. Tambah Data Dosen
2. Tampilkan Data Dosen
3. Sorting ASC (Bubble Sort)
4. Sorting DSC (Selection Sort)
5. Sorting DSC (Insertion Sort)
6. Keluar
Pilih menu: 4
Data berhasil diurutkan secara descending (usia tertua ke termuda).
Kode: 333
Nama: zaid
Jenis Kelamin: Laki-laki
Usia: 32

Kode: 111
Nama: arief
Jenis Kelamin: Laki-laki
Usia: 24

Kode: 222
Nama: risky
Jenis Kelamin: Laki-laki
Usia: 19

```
Menu:
1. Tambah Data Dosen
2. Tampilkan Data Dosen
3. Sorting ASC (Bubble Sort)
4. Sorting DSC (Selection Sort)
5. Sorting DSC (Insertion Sort)
6. Keluar
Pilih menu: 5
Data berhasil diurutkan menggunakan Insertion Sort (usia tertua ke termuda).
Kode: 333
Nama: zaid
Jenis Kelamin: Laki-laki
Usia: 32
-----------------------------------------
Kode: 111
Nama: arief
Jenis Kelamin: Laki-laki
Usia: 24
-----------------------------------------
Kode: 222
Nama: risky
Jenis Kelamin: Laki-laki
Usia: 19
-----------------------------------------

Menu:
1. Tambah Data Dosen
2. Tampilkan Data Dosen
3. Sorting ASC (Bubble Sort)
4. Sorting DSC (Selection Sort)
5. Sorting DSC (Insertion Sort)
6. Keluar
Pilih menu: 6
Program selesai.
PS C:\Users\Lenovo LOQ>
```