LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

SORTING

AHMAD SOFYAN BADAWI

244107020073

KELAS TI-1B

PRODI D-IV TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2025

1. Percobaan Praktikum

1.1 Percobaan 1 (Implementasi Sorting menggunakan object)

1.1.1 Sorting – Bubble Sort

- Buat class Sorting, kemudian tambahkan atribut sebagai berikut:

```
public class Sorting04 {
    int[] data;
    int jumData;
```

- Buatlah konstruktor dengan parameter Data[] dan jmlData

```
    Sorting04(int Data[], int jmlData) {
        jumData = jmlData;
        data = new int[jmlData];
        for (int i = 0; i < jmlData; i++) {
            data[i] = Data[i];

        }
    }
```

- 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;
                }
            }
        }
    }
```

- 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();
    }
```

- Buat class SortingMain kemudian deklarasikan array dengan nama a[] kemudian isi array tersebut

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

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

```
Sorting04 dataurut1 = new Sorting04(a, a.length);
```

- 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 (AS
dataurut1.tampil();
```

- 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
```

1.1.2 Sorting – Selection Sort

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

```
void SelectSort() {
    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;
    }
```

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

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

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

```
Sorting04 dataurut2 = new Sorting04(b, b.length);
```

- Lakukan pemanggilan method SelectionSort dan tampil

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

- 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
```

1.1.3 Sorting – Insertion Sort

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

```
void insertSort() {
    for (int i = 0; i <= data.length; 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;
    }
}
```

- Deklarasikan array dengan nama c[] pada kelas SortingMain kemudian isi array tersebut

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

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

```
Sorting04 dataurut3 = new Sorting04(c, c.length);
```

- Lakukan pemanggilan method insertionSort dan tampil

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

- Jalankan program dan amati hasilnya!

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

1.1.4 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;
}
```

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

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

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

4. Pada Insertion sort, apakah tujuan dari perintah

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

Jawaban:

1. Fungsi: Kode ini digunakan untuk menukar (swap) dua elemen yang berdekatan (data[j-1] dan data[j]) jika elemen sebelumnya lbih besar

2.

```
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 = i;
            }        Chat (CTRL + I) / Edit (CTRL + L)
        }
        Mahasiswa04 tmp = listMhs[idxMin];
        listMhs[idxMin] = listMhs[i];
        listMhs[i] = tmp;

    }
}
```

3. **j >= 0:** Memastikan pergeseran tidak melewati batas awal array (indeks 0)

data[j] > temp: Mengecek apakah elemen di posisi j lebih besar dari nilai yang disimpan (temp)

4. Untuk menyimpan nilai dari array data indeks j ke array data indeks j+1

1.2 Percobaan 2 (Sorting Menggunakan Array of Object)

1.2.1 Mengurutkan Data Mahasiswa Berdasarkan IPK (Bubble Sort)

- Buatlah class dengan nama Mahasiswa.
- Untuk lebih jelasnya class tersebut dapat dilihat pada potongan kode di bawah ini

```java
public class Mahasiswa04 {
    String nama, kelas, nim;
    double ipk;

    Mahasiswa04() {

    }

    Mahasiswa04(String nm, String name, String kls, double ip) {
        nim = nm;
        nama = name;
        kelas = kls;
        ipk = ip;
    }

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

    }
}
```

- Buat class MahasiswaBerprestasi seperti di bawah ini!

```java
public class MahasiswaBerprestasi04 {
    Mahasiswa04[] listMhs = new Mahasiswa04[5];
    int idx;
```

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

```java
    void tambah(Mahasiswa04 m) {
        if (idx < listMhs.length) {
            listMhs[idx] = m;
            idx++;
        } else {
            System.out.println(x:"Data sudah penuh!");
        }
    }
```

- 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.

```java
    void tampil() {
        for (Mahasiswa04 m : listMhs) {
            m.tampilInformasi();
            System.out.println(x:"------------------------");
        }
    }
```

- 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) {
                Mahasiswa04 tmp = listMhs[j];
                listMhs[j] = listMhs[j-1];
                listMhs[j-1] = tmp;
            }
        }
    }
}
```

- 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.

```
public static void main(String[] args) {
    MahasiswaBerprestasi list = new MahasiswaBerprestasi();
    Mahasiswa m1 = new Mahasiswa(nm:"123", name:"Zidan",kls:"2A", ip:3.2);
    Mahasiswa m2 = new Mahasiswa(nm:"124", name:"Ayu",kls:"2A", ip:3.5);
    Mahasiswa m3 = new Mahasiswa(nm:"125", name:"Sofi",kls:"2A", ip:3.1);
    Mahasiswa m4 = new Mahasiswa(nm:"126", name:"Sita",kls:"2A", ip:3.9);
    Mahasiswa m5 = new Mahasiswa(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();
}
```

- Jalankan program dan amati hasilnya!

```
-------------------------
Data mahasiswa sebelum sorting:
-------------------------
Nama    : dika
NIM     : 222
Kelas   : 2c
IPK     : 3.0
-------------------------
Nama    : susi
NIM     : 444
Kelas   : 2c
IPK     : 3.1
-------------------------
Nama    : ayu
NIM     : 111
Kelas   : 2c
IPK     : 3.7
-------------------------
Nama    : 333
NIM     : ila
Kelas   : 2c
IPK     : 3.8
-------------------------
Nama    : yayuk
NIM     : 555
Kelas   : 2c
IPK     : 3.4
-------------------------
```

```
Data mahasiswa setelah sorting berdasarkan IPK (DESC):
-------------------------
Nama    : 333
NIM     : ila
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
-------------------------
```

- **Pertanyaan**

1. Perhatikan perulangan di dalam bubbleSort() dibawah ini!

   a. Mengapa syarat dari perulangan i adalah i<listMhs.length-1?
   b. Mengapa syarat dari perulangan j adalah j<listMhs.length-i?
   c. Jika banyak data di dalam listMhs adalah 50, maka berapakali perulangan i akan berlangsung? Dan ada berapa Tahap bubble sort yang ditempuh?

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

Jawaban:

1. a. Karena dalam bubble sort, setelah n-1 iterasi (di mana n adalah jumlah elemen), array sudah pasti terurut.
   Misalnya, jika ada 5 elemen, kita hanya perlu 4 iterasi luar untuk memastikan semua elemen berada di posisi yang benar (elemen terbesar akan "mengapung" ke posisi terakhir pada setiap iterasi). Jadi, listMhs.length - 1 memastikan kita tidak melakukan iterasi tambahan yang tidak perlu.

b. Setelah setiap iterasi i, elemen terbesar sudah berada di posisi terakhir dan tidak perlu diperiksa lagi.
Dengan mengurangi i, kita membatasi perbandingan hanya pada bagian array yang belum terurut.
Misalnya, setelah iterasi pertama (i=0), elemen terbesar sudah di akhir, sehingga pada iterasi kedua (i=1), kita hanya memeriksa n-1 elemen pertama, dst.

c. Setelah setiap iterasi i, elemen terbesar sudah berada di posisi terakhir dan tidak perlu diperiksa lagi.
Dengan mengurangi i, kita membatasi perbandingan hanya pada bagian array yang belum terurut.
Misalnya, setelah iterasi pertama (i=0), elemen terbesar sudah di akhir, sehingga pada iterasi kedua (i=1), kita hanya memeriksa n-1 elemen pertama, dst.

2.

```
int listMhs = 5;
MahasiswaBerprestasi04 list = new MahasiswaBerprestasi04();


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

    Mahasiswa04 m = new Mahasiswa04(nim, nama, kelas, ipk);
    list.tambah(m);
}
```

1.2.2 Mengurutkan Data Mahasiswa Berdasarkan IPK (Selection Sort)

- 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;
            }
        }
        Mahasiswa04 tmp = listMhs[idxMin];
        listMhs[idxMin] = listMhs[i];
        listMhs[i] = tmp;

    }
}
```

- 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 mahasiswa setelah sorting berdasarkan IPK
System.out.println(x:"-------------------------");
list.selectionSort();
list.tampil();
```

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

```
-------------------------
Data mahasiswa sebelum sorting:
-------------------------
Nama    : dika
NIM     : 222
Kelas   : 2c
IPK     : 3.0
-------------------------
Nama    : susi
NIM     : 444
Kelas   : 2c
IPK     : 3.1
-------------------------
Nama    : ayu
NIM     : 111
Kelas   : 2c
IPK     : 3.7
-------------------------
Nama    : 333
NIM     : ila
Kelas   : 2c
IPK     : 3.8
-------------------------
Nama    : yayuk
NIM     : 555
Kelas   : 2c
IPK     : 3.4
-------------------------
```

```
Data mahasiswa setelah sorting berdasarkan IPK menggunakan SELECTION 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    : 333
NIM     : ila
Kelas   : 2c
IPK     : 3.8
-------------------------
```

- **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!

Jawaban:

Untuk mencari nilai paling terkecil pada data yang ada di array

1.2.3 Mengurutkan Data Mahasiswa Berdasarkan IPK (Insertion Sort)

- 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 = 0; i < listMhs.length; i++) {
        Mahasiswa04 temp = listMhs[i];
        int j = i;
        while (j > 0 && listMhs[j-1].ipk > temp.ipk) {
            listMhs[j] = listMhs[j-1];
            j--;
        }
        listMhs[j] = temp;
    }
}
```

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

```
System.out.println(x:"Data mahasiswa setelah sorting berdasarkan IPK m
System.out.println(x:"-------------------------");
list.insertionSort();
list.tampil();
}
```

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

```
-------------------------
Data mahasiswa sebelum sorting:
-------------------------
Nama    : dika
NIM     : 222
Kelas   : 2c
IPK     : 3.0
-------------------------
Nama    : susi
NIM     : 444
Kelas   : 2c
IPK     : 3.1
-------------------------
Nama    : ayu
NIM     : 111
Kelas   : 2c
IPK     : 3.7
-------------------------
Nama    : 333
NIM     : ila
Kelas   : 2c
IPK     : 3.8
-------------------------
Nama    : yayuk
NIM     : 555
Kelas   : 2c
IPK     : 3.4
-------------------------
```

```
Data mahasiswa setelah sorting berdasarkan IPK 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    : 333
NIM     : ila
Kelas   : 2c
IPK     : 3.8
-------------------------
```

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

```
void insertionSort() {
    for (int i = 0; i < listMhs.length; i++) {
        Mahasiswa04 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. Latihan Praktikum

```
public class Dosen04 {
    String kode, nama;
    Boolean jenisKelamin;
    int usia;

    Dosen04(String kd, String name, Boolean jk, int age) {
        nama = name;
        kode = kd;
        jenisKelamin = jk;
        usia = age;
    }

    Dosen04() {

    }

    void tampilData() {
        System.out.println("Kode: " + kode);
        System.out.println("Nama: " + nama);
        System.out.println("Usia: " + usia);
        if (jenisKelamin) { // default true = laki-laki, false = perempuan
            System.out.println(x:"Jenis Kelamin: Laki-Laki");
        } else {
            System.out.println(x:"Jenis Kelamin: Perempuam ");

        }


    }
}
```

```java
public class DataDosen04 {
    Dosen04[] dataDosen = new Dosen04[10];
    int idx;

    void tambah(Dosen04 dsn) {
        if (idx < dataDosen.length) {
            dataDosen[idx] = dsn;
            idx++;
        } else {
            System.out.println(x:"Data sudah penuh");
        }
    }

    void tampil() {
        if (dataDosen[0] == null) {
            System.out.println(x:"Belum ada data!!!");

        } else {
            for (Dosen04 dsn : dataDosen) {
                    dsn.tampilData();
                    System.out.println(x:"----------------------");
                }
        }
    }

    void SortASC() {
        for (int i = 0; i < dataDosen.length; i++) {
            for (int j = 1; j < dataDosen.length - i; j++) {
                if (dataDosen[j-1].usia > dataDosen[j].usia) {
                    Dosen04 temp = dataDosen[j];
                    dataDosen[j] = dataDosen[j-1];
                    dataDosen[j-1] = temp;
                }
            }
        }
    }

    void SortDSC() {
        for (int i = 0; i < dataDosen.length - 1; i++) {
            int max = i;
            for (int j = 1; j < dataDosen.length; j++) {
                if (dataDosen[j].usia > dataDosen[max].usia) {
                    max = j;
                }
            }
            Dosen04 temp = dataDosen[max];
            dataDosen[max] = dataDosen[i];
            dataDosen[i] = temp;
        }
    }

    void InsertSort() {
        for (int i = 1; i < dataDosen.length; i++) {
            Dosen04 temp = dataDosen[i];
            int j = i;
            while (j >= 0 && dataDosen[j].usia > temp.usia) {
                dataDosen[j+1] = dataDosen[j];
                j--;
            }
            dataDosen[j+1] = temp;
        }
    }
}
```

```java
import java.util.Scanner;
public class DosenMain04 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        DataDosen04 list = new DataDosen04();
        Dosen04 dsn[] = new Dosen04[list.dataDosen.length];
        boolean pilihan = true;
        int opsi;
        String gender;

        while (pilihan) {
            System.out.println(x:"--------------------");
            System.out.println(x:"REKAP DATA DOSEN");
            System.out.println(x:"--------------------");
            System.out.println(x:"1. Tambah Data Dosen");
            System.out.println(x:"2. Tampilkan Data Dosen");
            System.out.println(x:"3. Sorting Data berdasarkan Umur (ASC)");
            System.out.println(x:"4. Sorting Data berdasarkan Umur (DSC)");
            System.out.println(x:"5. Exit");
            System.out.print(s:"Pilih Menu: ");
            opsi = sc.nextInt();
            sc.nextLine();

            switch (opsi) {
                case 1:
                    for (int i = 0; i < dsn.length; i++) {
                        if (i >= dsn.length) {
                            System.out.println(x:"Data sudah penuh");
                        } else {
                            System.out.println(x:"--------------------");
                            System.out.println("Dosen ke-" + (i+1));
                            dsn[i] = new Dosen04();
                            System.out.print(s:"Masukkan Kode Dosen: ");
                            dsn[i].kode = sc.nextLine();
                            System.out.print(s:"Masukkan Nama Dosen: ");
                            dsn[i].nama = sc.nextLine();
                            System.out.print(s:"Masukkan Usia: ");
                            dsn[i].usia = sc.nextInt();
                            sc.nextLine();
                            System.out.print(s:"Masukkan Jenis Kelamin (L/P): ");
                            gender = sc.nextLine();
                            if (gender.equalsIgnoreCase(anotherString:"L")) {
                                dsn[i].jenisKelamin = true;
                            } else if (gender.equalsIgnoreCase(anotherString:"P")) {
                                dsn[i].jenisKelamin = false;
                            } else {
                                System.out.println(x:"Jenis Kelamin tidak valid");
                                continue;
                            }
                            list.tambah(dsn[i]);
                        }
                    }
                    break;

                case 2:
                    System.out.println(x:"--------------------");
                    System.out.println(x:"DATA DOSEN");
                    System.out.println(x:"--------------------");
                    list.tampil();;
                    break;
                case 3:
                    System.out.println(x:"--------------------");
                    System.out.println(x:"Sorting Data berdasarkan Umur (ASC)");
                    System.out.println(x:"--------------------");
                    list.SortASC();
                    list.tampil();
                    break;
```

```
                case 2:
                    System.out.println(x:"--------------------");
                    System.out.println(x:"DATA DOSEN");
                    System.out.println(x:"--------------------");
                    list.tampil();;
                    break;
                case 3:
                    System.out.println(x:"--------------------");
                    System.out.println(x:"Sorting Data berdasarkan Umur (ASC)");
                    System.out.println(x:"--------------------");
                    list.SortASC();
                    list.tampil();
                    break;
                case 4:
                    System.out.println(x:"--------------------");
                    System.out.println(x:"Sorting Data berdasarkan Umur (DSC)");
                    System.out.println(x:"--------------------");
                    list.SortDSC();
                    list.tampil();
                    break;

                case 5:
                    pilihan = false;
                    System.out.println(x:"Terima kasih!");
                    break;

                default:
                    break;
            }

        }
    }
}
```