JOBSHEET VII
SEARCHING

7.1. Tujuan Praktikum

Setelah melakukan materi praktikum ini, mahasiswa mampu:

1. Menjelaskan mengenai algoritma Searching.
2. Membuat dan mendeklarasikan struktur algoritma Searching.
3. Menerapkan dan mengimplementasikan algoritma Searching.

7.2. Searching / Pencarian Menggunakan Agoritma Sequential Search

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, akan dibuat class Mahasiswa yang berfunsi untuk membuat objek mahasiswa yang akan dimasukan ke dalam sebuah array. Terdapat sebuah konstruktor berparameter dan juga fungsi tampilInformasi() untuk menampilkan semua attribute yang ada.

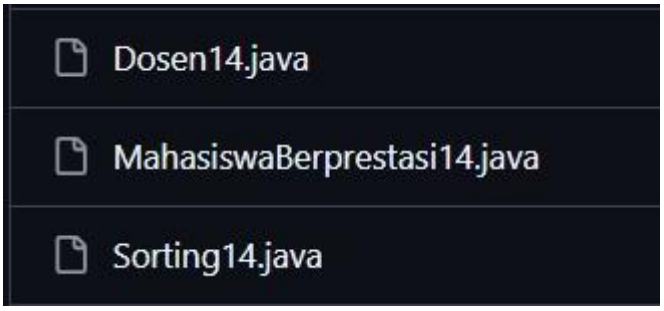
MahasiswaBerprestasi
listMhs: Mahasiswa[5] idx: int
tambah(mhs: Mahasiswa): void tampil(): void sequentialSearch(double cari): int tampilPoisisi(double x,int pos): void tampilDataSearch(double x,int pos) :void

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, untuk melakukan pencarian berdasarkan IPK menggunakan algoritma Sequential Search, menampilkan posisi dari data yang dicari, serta

menampilkan data mahasiswa yang dicari.

7.1.1. Langkah-langkah Percobaan Sequential Search

1. Pada pertemuan Jobsheet 7 ini akan menggunakan class **Mahasiswa**<no Presensi>, **MahasiswaBerprestasi**<no Presensi>, dan **MahasiswaDemo**<no presensi> pada pertemuan Jobsheet 6 sebelumnya
2. Buat folder baru bernama **Jobsheet7** di dalam repository **Praktikum ASD**, kemudian buka ketiga class dari Jobsheet 6 tersebut dan copy ke folder Jobsheet 7



3. Tambahkan method **`sequentialSearching`** bertipe integer dengan parameter **`cari`** bertipe double pada class **MahasiswaBerprestasi**<no presensi>. Kemudian Deklarasikan isi method **`sequentialSearching`** dengan algoritma pencarian data menggunakan teknik sequential searching.

```
int sequentialSearching(double cari) {
    int posisi = -1;
    for (int j=0; j<listMhs.length; j++) {
        if (listMhs[j].ipk==cari) {
            posisi=j;
            break;
        }
    }
    return posisi;
}
```

4. Buatlah method **`tampilPoisisi`** bertipe void dan Deklarasikan isi dari method **`tampilPoisisi`**

pada class MahasiswaBerprestasi<no presensi>.

```
void tampilPosisi(double x,int pos) {
    if (pos!=-1) {
        System.out.println("data mahasiswa dengan IPK :" +X+ " ditemukan pada indeks " +pos);
    }
    else {
        System.out.println("data " +X+ "tidak ditemukan");
    }
}
```

5. Pada class MahasiswaBerprestasi<no presensi>, buatlah method **`tampilDataSearch`** bertipe void dan Deklarasikan isi dari method **`tampilDataSearch .`**

```
void tampilataSearch(double x, int pos){
    if (pos !=-1){
        System.out.println("nim\t : "+listMhs[pos].nim);
        System.out.println("nama\t : "+listMhs[pos].nama);
        System.out.println("kelas\t "+listMhs[pos].kelas);
        System.out.println("ipk\t : "+x);
    }
}
```

6. Pada class **MahasiswaDemo**<noPresensi> , tambahkan kode program berikut ini untuk melakukan pencarian data dengan algoritma sequential searching.

```
        list.tampil();
        //melakukan pencarian data sequential
        System.out.println(x:"-----------------------------------------");
        System.out.println(x:"Pencarian data");
        System.out.println(x:"-----------------------------------------");
        System.out.println(x:"masukkan ipk mahasiswa yang dicari: ");
        System.out.print(s:"IPK: ");
        double cari = input.nextDouble();

        System.out.println(x:"menggunakan sequential searching");
        double posisi = list.sequentialSearching(cari);
        int pss= (int)posisi;
        list.tampilPosisi(cari, pss);
        list.tampilDataSearch(cari, pss);
    }
}
```

7. Jalankan dan amati hasilnya.

7.1.2. Verifikasi Hasil Percobaan

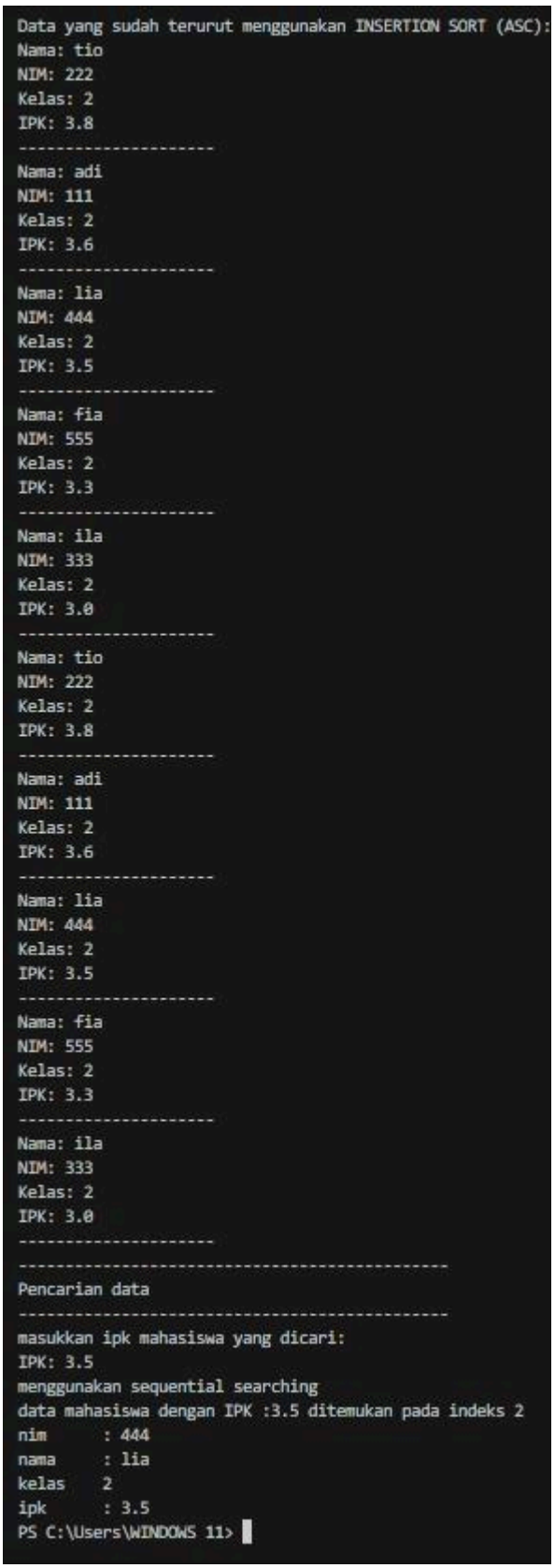
Cocokkan hasil kode program anda dengan gambar berikut ini.

```
Masukkan Data Mahasiswa ke-1
NIM   : 111
Nama  : adi
Kelas : 2
IPK   : 3.6
--------------------------------
Masukkan Data Mahasiswa ke-2
NIM   : 222
Nama  : tio
Kelas : 2
IPK   : 3.8
--------------------------------
Masukkan Data Mahasiswa ke-3
NIM   : 333
Nama  : ila
Kelas : 2
IPK   : 3.0
--------------------------------
Masukkan Data Mahasiswa ke-4
NIM   : 444
Nama  : lia
Kelas : 2
IPK   : 3.5
--------------------------------
Masukkan Data Mahasiswa ke-5
NIM   : 555
Nama  : fia
Kelas : 2
IPK   : 3.3
--------------------------------
```

```
Nama: adi
NIM: 111
Kelas: 2
IPK: 3.6
----------------------------
Nama: tio
NIM: 222
Kelas: 2
IPK: 3.8
----------------------------
Nama: ila
NIM: 333
Kelas: 2
IPK: 3.0
----------------------------
Nama: lia
NIM: 444
Kelas: 2
IPK: 3.5
----------------------------
Nama: fia
NIM: 555
Kelas: 2
IPK: 3.3
----------------------------
------------------------------------------------
Pencarian data
------------------------------------------------
masukkan ipk mahasiswa yang dicari:
IPK: 3.5
menggunakan sequential searching
data mahasiswa dengan IPK :3.5  ditemukan pada indeks 3
nim      : 444
nama     : lia
kelas    : 2
ipk      : 3.5
```

```
Masukkan jumlah mahasiswa: 5
Masukkan data mahasiswa ke-1:
NIM: 111
Nama: adi
Kelas: 2
IPK: 3.6
Masukkan data mahasiswa ke-2:
NIM: 222
Nama: tio
Kelas: 2
IPK: 3.8
Masukkan data mahasiswa ke-3:
NIM: 333
Nama: ila
Kelas: 2
IPK: 3.0
Masukkan data mahasiswa ke-4:
NIM: 444
Nama: lia
Kelas: 2
IPK: 3.5
Masukkan data mahasiswa ke-5:
NIM: 555
Nama: fia
Kelas: 2
IPK: 3.3

Data mahasiswa sebelum sorting:
Nama: adi
NIM: 111
Kelas: 2
IPK: 3.6
----------------------
Nama: tio
NIM: 222
Kelas: 2
IPK: 3.8
----------------------
Nama: ila
NIM: 333
Kelas: 2
IPK: 3.0
----------------------
Nama: lia
NIM: 444
Kelas: 2
IPK: 3.5
----------------------
Nama: fia
NIM: 555
Kelas: 2
IPK: 3.3
----------------------
```



7.1.3. Pertanyaan

1. Jelaskan perbedaan metod **`tampilDataSearch`** dan **`tampilPosisi`** pada class MahasiswaBerprestasi!

2. Jelaskan fungsi **`break`** pada kode program dibawah ini!

```
if (listMhs[j].ipk==cari){
    posisi=j;
    break;
}
```

Jawaban Pertanyaan

1. pada metho tampilPosisi hanya menampilkan ipk dan indeks dimana pada indeks tersebut adalah mahasiswa yang dicari oleh user, sedangkan pada method tampilDataSearch akan menampilkan semua data yang ada di array ke indeks(posisi)
2. jika data pada array sudah sesuai variabel “cari” maka sequantial search akan berhenti

7.1. Searching / Pencarian Menggunakan Binary Search

7.1.1. Langkah-langkah Percobaan Binary Search

1. Pada percobaan 6.2.1 (sequential search) tambahkan method **`findBinarySearch`** bertipe integer pada class **`MahasiswaBerprestasi`**. Kemudian Deklarasikan isi method **`findBinarySearch`** dengan algoritma pencarian data menggunakan teknik binary searching.

```
    int findBinarySearch(double cari, int left,int right){
        int mid;
        if (right>=left) {
            mid =(left+right)/2;
            if (cari ==listMhs[mid].ipk) {
                return (mid);
            }
            else if (listMhs[mid].ipk < cari) {
                return findBinarySearch(cari, left, mid-1);
            }
            else{
                return findBinarySearch(cari, mid+1,right);
            }
        }
        return -1;
    }
}
```

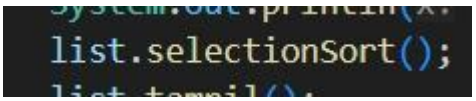
2. Panggil method **`findBinarySearch`** terdapat pada class **`MahasiswaBerprestasi`** di kelas

`MahasiswaDemo.` Kemudia panggil **`method tampilPosisi`** dan **`tampilDataSearch`**

```
    list.tampil();
    //melakukan pencarian data sequential
    System.out.println(x:"------------------------------------------");
    System.out.println(x:"Pencarian data");
    System.out.println(x:"------------------------------------------");
    System.out.println(x:"masukkan ipk mahasiswa yang dicari: ");
    System.out.print(s:"IPK: ");
    double cari = input.nextDouble();

    System.out.println(x:"------------------------------------------");
    System.out.println(x:"menggunakan binary search");
    System.out.println(x:"------------------------------------------");
    double posisi2 = list.findBinarySearch(cari,left:0,jumlah-1);
    int pss2= (int)posisi2;
    list.tampilPosisi(cari, pss2);
    list.tampilDataSearch(cari, pss2);
}
```

3. Jalankan dan amati hasilnya (inputkan data IPK secara terurut -ASC seperti verifikasi hasil percobaan dibawah ini).



7.1.2. Verifikasi Hasil Percobaan

Cocokkan hasil kode program anda dengan gambar berikut ini.

```
Masukkan Data Mahasiswa ke-1
NIM   : 111
Nama  : adi
Kelas : 2
IPK   : 3.1
--------------------------------
Masukkan Data Mahasiswa ke-2
NIM   : 222
Nama  : ila
Kelas : 2
IPK   : 3.2
--------------------------------
Masukkan Data Mahasiswa ke-3
NIM   : 333
Nama  : lia
Kelas : 2
IPK   : 3.3
--------------------------------
Masukkan Data Mahasiswa ke-4
NIM   : 444
Nama  : susi
Kelas : 2
IPK   : 3.5
--------------------------------
Masukkan Data Mahasiswa ke-5
NIM   : 555
Nama  : anita
Kelas : 2
IPK   : 3.7
--------------------------------
```

```
Nama: adi
NIM: 111
Kelas: 2
IPK: 3.1
----------------------------
Nama: ila
NIM: 222
Kelas: 2
IPK: 3.2
----------------------------
Nama: lia
NIM: 333
Kelas: 2
IPK: 3.3
----------------------------
Nama: susi
NIM: 444
Kelas: 2
IPK: 3.5
----------------------------
Nama: anita
NIM: 555
Kelas: 2
IPK: 3.7
----------------------------
------------------------------------------------
Pencarian data
------------------------------------------------
masukkan ipk mahasiswa yang dicari:
IPK: 3.7
--------------------------------
menggunakan binary search
--------------------------------
data mahasiswa dengan IPK :3.7  ditemukan pada indeks 4
nim      : 555
nama     : anita
kelas    : 2
ipk      : 3.7
```

```
Masukkan jumlah mahasiswa: 5
Masukkan data mahasiswa ke-1:
NIM: 111
Nama: adi
Kelas: 2
IPK: 3.1
Masukkan data mahasiswa ke-2:
NIM: 222
Nama: ila
Kelas: 2
IPK: 3.2
Masukkan data mahasiswa ke-3:
NIM: 333
Nama: lia
Kelas: 2
IPK: 3.3
Masukkan data mahasiswa ke-4:
NIM: 444
Nama: susi
Kelas: 2
IPK: 3.5
Masukkan data mahasiswa ke-5:
NIM: 555
Nama: anita
Kelas: 2
IPK: 3.7

Data mahasiswa sebelum sorting:
Nama: adi
NIM: 111
Kelas: 2
IPK: 3.1
---------------------
Nama: ila
NIM: 222
Kelas: 2
IPK: 3.2
---------------------
Nama: lia
NIM: 333
Kelas: 2
IPK: 3.3
---------------------
Nama: susi
NIM: 444
Kelas: 2
IPK: 3.5
---------------------
Nama: anita
NIM: 555
Kelas: 2
IPK: 3.7
---------------------
```

```
Data yang sudah terurut menggunakan INSERTION SORT (ASC):
Nama: adi
NIM: 111
Kelas: 2
IPK: 3.1
---------------------
Nama: ila
NIM: 222
Kelas: 2
IPK: 3.2
---------------------
Nama: lia
NIM: 333
Kelas: 2
IPK: 3.3
---------------------
Nama: susi
NIM: 444
Kelas: 2
IPK: 3.5
---------------------
Nama: anita
NIM: 555
Kelas: 2
IPK: 3.7
---------------------
Nama: adi
NIM: 111
Kelas: 2
IPK: 3.1
---------------------
Nama: ila
NIM: 222
Kelas: 2
IPK: 3.2
---------------------
Nama: lia
NIM: 333
Kelas: 2
IPK: 3.3
---------------------
Nama: susi
NIM: 444
Kelas: 2
IPK: 3.5
---------------------
Nama: anita
NIM: 555
Kelas: 2
IPK: 3.7
---------------------
-----------------------------------------------
Pencarian data
-----------------------------------------------
```

```
-----------------------------------------
Pencarian data
-----------------------------------------
masukkan ipk mahasiswa yang dicari:
IPK: 3.7
-----------------------------------------
menggunakan binary search
-----------------------------------------
data mahasiswa dengan IPK :3.7 ditemukan pada indeks 4
nim      : 555
nama     : anita
kelas    2
ipk      : 3.7
PS C:\Users\WINDOWS 11>
```

7.1.3. Pertanyaan

1. Tunjukkan pada kode program yang mana proses divide dijalankan!
2. Tunjukkan pada kode program yang mana proses conquer dijalankan!
3. Jika data IPK yang dimasukkan tidak urut. Apakah program masih dapat berjalan? Mengapa demikian!
4. Jika IPK yang dimasukkan dari IPK terbesar ke terkecil (missal : 3.8, 3.7, 3.5, 3.4, 3.2) dan elemen yang dicari adalah 3.2. Bagaimana hasil dari binary search? Apakah sesuai? Jika tidak sesuai maka ubahlah kode program binary seach agar hasilnya sesuai
5. Modifikasilah program diatas yang mana jumlah mahasiswa yang di inputkan sesuai dengan masukan dari keyboard.

Jawaban Pertanyaan

1.
```
mid =(left+right)/2;
```

2.
```
if (right>=left) {
    mid =(left+right)/2;
    if (cari ==listMhs[mid].ipk) {
        return (mid);
    }
    else if (listMhs[mid].ipk>cari) {
        return findBinarySearch(cari, left, mid-1);
    }
    else{
        return findBinarySearch(cari, mid+1,right);
    }
}
```

3. program masih akan tetap berjalan tapi hasilnya tetap data tidak ditemukan, karena binary

search hanya bekerja dengan benar jika data sudah dalam keadaan terurut. contoh :

[3.5, 2.0, 3.8, 1.9, 3.2] cari 3.2, lalu mid ketemu di 3.8.

Karena 3.2 < 3.8, akan buang sisi kanan dan lanjut ke kiri padahal 3.2 aa di kanan

4. tidak sesuai, karena

```
mid =(left+right)/2;
if (cari ==listMhs[mid].ipk) {
    return (mid);
}
else if (listMhs[mid].ipk>cari) {
    return findBinarySearch(cari, left, mid-1);
}
else{
    return findBinarySearch(cari, mid+1,right);
}
```

j

karena kode diatas tertulis ipk > cari maka mid -1, jadi misal mid = 3.5, cari = 3.2, maka malah akan ke kiri, paahal 3.2 ada dikanan

solusi:

```
else if (listMhs[mid].ipk < cari) {
```

tinggal ubah ipk > cari menjadi ipk < cari

5.

```java
import java.util.Scanner;
public class MahasiswaDemo14 {
    Run | Debug
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print(s:"Masukkan jumlah mahasiswa: ");
        int jumlah = input.nextInt();
        input.nextLine();
        MahasiswaBerprestasi14 list = new MahasiswaBerprestasi14(jumlah);
        for (int i = 0; i < jumlah; 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();
            Mahasiswa14 m = new Mahasiswa14(nim, nama, kelas, ipk);
            list.tambah(m);
        }
        System.out.println(x:"\nData mahasiswa sebelum sorting:");
        list.tampil();
        System.out.println(x:"Data yang sudah terurut menggunakan INSERTION SORT (ASC):");
        list.selectionSort();
        list.tampil();

        list.tampil();
        //melakukan pencarian data sequential
        System.out.println(x:"-----------------------------------------");
        System.out.println(x:"Pencarian data");
        System.out.println(x:"-----------------------------------------");
        System.out.println(x:"masukkan ipk mahasiswa yang dicari: ");
        System.out.print(s:"IPK: ");
        double cari = input.nextDouble();

        System.out.println(x:"-----------------------------------------");
        System.out.println(x:"menggunakan binary search");
        System.out.println(x:"-----------------------------------------");
        double posisi2 = list.findBinarySearch(cari,left:0,jumlah-1);
        int pss2= (int)posisi2;
        list.tampilPosisi(cari, pss2);
        list.tampilDataSearch(cari, pss2);
    }
}
```

1.

7.5. Latihan Praktikum

1. Pada Latihan praktikum pertemuan sebelumnya pada Jobsheet 6 yang terdapat 3 class yaitu **Dosen**<no presensi>, **DataDosen**<no presensi> , dan **DosenDemo**<no presensi>, tambahkan method:
   a. PencarianDataSequential<no presensi> : digunakan untuk mencari data dosen berdasarkan nama dengan algoritma sequential search.
   b. PencarianDataBinary<no presensi> : digunakan untuk mencari data dosen berdasarkan usia dengan algoritma Binary Search.
   c. Buat aturan untuk mendeteksi hasil pencarian lebih dari 1 hasil dalam bentuk kalimat peringatan! Pastikan algoritma yang diterapkan sesuai dengan kasus yang diberikan!

//FILE MAIN

```java
import java.util.Scanner;

public class DosenMain14 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc14 = new Scanner(System.in);
        DataDosen14 data14 = new DataDosen14(jumlah14:10);
        int pilihan14;
                do {
                    System.out.println(x:"-------- Menu --------");
                    System.out.println(x:"1. Tambah Data Dosen");
                    System.out.println(x:"2. Tampilkan Data Dosen");
                    System.out.println(x:"3. Urutkan Data (ASC) - Bubble Sort");
                    System.out.println(x:"4. Urutkan Data (DESC) - Selection Sort");
                    System.out.println(x:"5. Keluar");
                    System.out.println(x:"6. Cari Dosen berdasarkan Usia (Binary Search)");
                    System.out.println(x:"7. Cari Dosen berdasarkan Nama (Sequential Search)");
                    System.out.print(s:"Pilih menu: ");
                    pilihan14 = sc14.nextInt();
                    sc14.nextLine();
                    switch (pilihan14) {
                        case 1:
                    System.out.print(s:"Masukkan kode: ");
                    String kode14 = sc14.nextLine();
                    System.out.print(s:"Masukkan nama: ");
                    String nama14 = sc14.nextLine();
                    System.out.print(s:"Masukkan jenis kelamin (true = Laki-laki, false = Perempuan): ");
                    boolean jk14 = sc14.nextBoolean();
                    System.out.print(s:"Masukkan usia: ");
                    int usia14 = sc14.nextInt();
                    data14.tambah14(new Dosen14(kode14, nama14, jk14, usia14));
                    break;
        case 2:
                    data14.tampil14();
                    break;
                case 3:
                    data14.sortingASC14();
                    System.out.println(x:"Data berhasil diurutkan ASCENDING.");
                    data14.tampil14();
                    break;
                case 4:
                        data14.sortingDSC14();
                        System.out.println(x:"Data berhasil diurutkan DESCENDING.");
                        data14.tampil14();
                        break;
                case 5:
                    System.out.println(x:"Program selesai.");
                    break;
                case 6:
                    data14.sortingASC14();
                    System.out.print(s:"Masukkan usia dosen yang ingin dicari: ");
                    int usiaCari = sc14.nextInt();
                    int hasil = data14.binarySearchUsia14(usiaCari, left:0, data14.idx14 - 1);
                    if (hasil != -1) {
                        data14.tampilDosenUsiaSama(usiaCari);
                    } else {
                        System.out.println("Dosen dengan usia " + usiaCari + " tidak ditemukan.");
                    }
                    break;
                case 7:
                    System.out.print(s:"Masukkan nama dosen yang ingin dicari: ");
                    String cariNama = sc14.nextLine();
                    int posisi = data14.PencarianDataSquential14(cariNama);
                    if (posisi != -1) {
                        System.out.println("Dosen ditemukan pada index: " + posisi);
                        data14.dataDosen14[posisi].tampil();
                    } else {
                        System.out.println("Dosen dengan nama " + cariNama + " tidak ditemukan.");
                    }
                    break;
                default:
                    System.out.println(x:"Pilihan tidak valid!");
            }
        } while (pilihan14 != 5);
    }
}
```

//FILE DATADOSEN

```java
public class DataDosen14 {
    Dosen14[] dataDosen14;
    int idx14;
    public DataDosen14(int jumlah14) {
        dataDosen14 = new Dosen14[jumlah14];
        idx14 = 0;
    }
        public void tambah14(Dosen14 dsn14) {
            if (idx14 < dataDosen14.length) {
                dataDosen14[idx14] = dsn14;
                idx14++;
            } else {
                System.out.println(x:"Data penuh!");
            }
        }
            public void tampil14() {
            for (int i = 0; i < idx14; i++) {
                dataDosen14[i].tampil();
            }
        }
    public void sortingASC14() {
        for (int i = 0; i < idx14 - 1; i++) {
            for (int j = 0; j < idx14 - i - 1; j++) {
                if (dataDosen14[j].usia > dataDosen14[j + 1].usia) {
                    Dosen14 temp14 = dataDosen14[j];
                    dataDosen14[j] = dataDosen14[j + 1];
                    dataDosen14[j + 1] = temp14;
                }
            }
        }
    }
            public void sortingDSC14() {
        for (int i = 0; i < idx14 - 1; i++) {
            int maxIdx14 = i;
            for (int j = i + 1; j < idx14; j++) {
                if (dataDosen14[j].usia > dataDosen14[maxIdx14].usia) {
                    maxIdx14 = j;
                }
            }
                    Dosen14 temp14 = dataDosen14[maxIdx14];
                    dataDosen14[maxIdx14] = dataDosen14[i];
                    dataDosen14[i] = temp14;
                }
    }
    int PencarianDataSquential14(String cari) {
        int posisi = -1;
        for (int j=0;j<dataDosen14.length;j++) {
            if (dataDosen14[j].nama.equalsIgnoreCase(cari)) {
                posisi = j;
                break;
            }
        }
        return posisi;
    }
    public int binarySearchUsia14(int usiaCari, int left, int right) {
        if (right >= left) {
            int mid = (left + right) / 2;
            if (dataDosen14[mid].usia == usiaCari) {
                return mid;
            } else if (dataDosen14[mid].usia > usiaCari) {
                return binarySearchUsia(usiaCari, left, mid - 1);
            } else {
                return binarySearchUsia(usiaCari, mid + 1, right);
            }
        }
        return -1;
    }
    public void tampilDosenUsiaSama(int usia) {
        int count = 0;
        for (int i = 0; i < idx14; i++) {
            if (dataDosen14[i].usia == usia) {
                dataDosen14[i].tampil();
                count++;
            }
        }
        if (count > 1) {
            System.out.println("Terdapat " + count + " dosen dengan usia " + usia);
        } else if (count == 1) {
            System.out.println("Terdapat 1 dosen dengan usia " + usia);
```

```
    public void tampilDosenUsiaSama(int usia) {
        int count = 0;
        for (int i = 0; i < idx14; i++) {
            if (dataDosen14[i].usia == usia) {
                dataDosen14[i].tampil();
                count++;
            }
        }
        if (count > 1) {
            System.out.println("Terdapat " + count + " dosen dengan usia " + usia);
        } else if (count == 1) {
            System.out.println("Terdapat 1 dosen dengan usia " + usia);
        } else {
            System.out.println("Tidak ditemukan dosen dengan usia " + usia);
        }
    }

    void tampilDataSearch(String x,int pos) {
        if (pos!=-1) {
            System.out.println("kode: " + dataDosen14.kode);
            System.out.println("nama: " + dataDosen14.nama);
            System.out.println("Jenis Kelamin: " + (dataDosen14.jenisKelamin ? "Laki-laki" : "Perempuan"));
            System.out.println("usia: " + dataDosen14.usia);
        }
    }
    void tampilPosisi(String x,int pos) {
        if (x != -1) {
            System.out.println("data Dosen dengan nama: " + x + " ditemukan pada index " + pos);
        } else {
            System.out.println(x:"data dosen tidak ditemukan");
        }
    }
}
```

//FILE DOSEN

```
public class Dosen14 {
    String kode;
    String nama;
    boolean jenisKelamin;
    int usia;

    public Dosen14(String kd, String name, boolean jk, int age) {
        this.kode = kd;
        this.nama = name;
        this.jenisKelamin = jk;
        this.usia = age;
    }

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