**LAPORAN HASIL PRAKTIKUM ALSD**

**JOBSHEET 5 SORTING (BUBBLE, SELECTION, DAN INSERTION SORT)**



Faiva Puspa Sahara

244107020036

TI – 1E

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

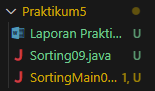
POLITEKNIK NEGERI MALANG

2025

**HASIL PERCOBAAN**

* 1. **Percobaan 1**

1. ***SORTING – BUBBLE SORT***
2. Buat folder baru dengan nama **Praktikum5**



1. Buat class **Sorting09** dan lengkapi dengan atribut

public class Sorting09 {

int[] data;

int jumData;

}

1. Buatlah konstruktor dengan parameter Data[] dan jmlDat

Sorting09 (int Data[], int jmlDat) {

jumData = jmlDat;

data = new int[jmlDat];

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

data[i] = Data[i];

}

}

1. 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; j++) {

if (data[j-1]>data[j]) {

temp = data[j];

data[j] = data[j-1];

data[j-1] = temp;

}

}

}

}

1. 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();

}

1. Buat class **SortingMain09** kemudian deklarasikan array dengan nama a[] kemudian isi array tersebut

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

1. Buatlah objek baru dengan nama dataurut1 yang merupakan instansiasi dari class Sorting, kemudian isi parameternya. Lakukan pemanggilan method bubbleSort

Sorting09 dataurut1 = new Sorting09(a, a.length);

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

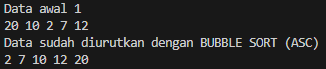
dataurut1.tampil();

dataurut1.bubbleSort();

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

dataurut1.tampil();

1. Run program



1. ***SORTING – SELECTION SORT***
2. Pada class **Sorting09** yang sudah dibuat di praktikum sebelumnya tambahkan method **SelectionSort09** yang mengimplementasikan pengurutan menggunakan algoritma selection sort

void SelectionSort() {

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

int min = i;

for (int j = 0; j < jumData; j++) {

if (data[j] < data[min]) {

min = j;

}

}

int temp = data[i];

data[i] = data [min];

data[min] = temp;

}

}

1. Deklarasikan array dengan nama b[] pada kelas **SortingMain09** kemudian isi array tersebut

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

1. Buatlah objek baru dengan nama dataurut2 yang merupakan instansiasi dari class Sorting, kemudian isi parameternya. Lakukan pemanggilan method SelectionSort

Sorting09 dataurut2 = new Sorting09(b, b.length);

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

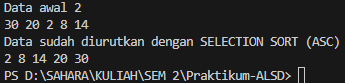
dataurut2.tampil();

dataurut2.bubbleSort();

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

dataurut2.tampil();

1. Run program



1. ***SORTING – INSERTION SORT***
2. Pada class **Sorting09** 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--;

}

}

}

1. Deklarasikan array dengan nama c[] pada kelas **SortingMain09** kemudian isi array tersebut

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

1. Buatlah objek baru dengan nama dataurut3 yang merupakan instansiasi dari class Sorting, kemudian isi parameternya. Lakukan pemanggilan method *insertionSort.*

Sorting09 dataurut3 = new Sorting09(c, c.length);

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

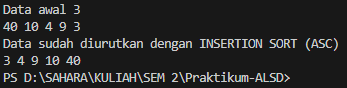
dataurut3.tampil();

dataurut3.bubbleSort();

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

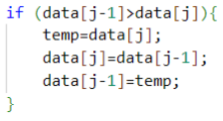
dataurut3.tampil();

1. Run program



* **Pertanyaan**

1. Jelaskan fungsi kode program berikut



* Berfungsi menukar posisi kedua elemen jika elemen sebelumnya

(data[j-1]) lebih besar dari elemen saat ini (data[j]). Maka kedua elemen harus ditukar agar berada dalam urutan yang benar (ascending).

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

void SelectionSort() {

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

int min = i;

for (int j = 0; j < jumData; j++) {

if (data[j] < data[min]) {

min = j;

}

}

int temp = data[i];

data[i] = data [min];

data[min] = temp;

}

}

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



* Kondisi ini memastikan **elemen lebih besar digeser ke kanan** sebelum elemen baru (temp) dimasukkan ke posisi yang benar. Membuat array tetap terurut selama proses penyisipan.

1. Pada Insertion sort, apakah tujuan dari perintah **data[ j + 1] = data[j];**

* Berfungsi menggeser elemen ke kanan, memberikan ruang bagi elemen yang lebih kecil untuk disisipkan. Dilakukan berulang kali sampai elemen yang lebih kecil berada di posisi yang benar dalam array.
  1. **Percobaan 2 – Mengurutkan Data Mahasiswa**

1. ***Bubble Sort***
2. Buatlah class **Mahasiswa09** dan masukkan kode programnya

public class Mahasiswa09 {

String nim;

String nama;

String kelas;

double ipk;

//Konstruktor default

Mahasiswa09() {

}

//Konstruktor berparameter

Mahasiswa09(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);

}

}

1. Buat class **MahasiswaBerprestasi09**

public class MahasiswaBerprestasi09 {

Mahasiswa09[] listMhs = new Mahasiswa09[5];

int idx;

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

void tambah (Mahasiswa09 m) {

if (idx < listMhs.length) {

listMhs[idx] = m;

idx++;

}else{

System.out.println("data sudah penuh");

}

}

1. Tambahkan method tampil() di dalam class tersebut! Method tampil() digunakan untuk menampilkan semua data mahasiswa-mahasiswa yang ada di dalam class

void tampil() {

for (Mahasiswa09 m : listMhs) {

m.tampilInformasi();

System.out.println("--------------------------");

}

}

1. 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) {

Mahasiswa09 tmp = listMhs[j];

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

listMhs[j-1] = tmp;

}

}

}

}

1. Buat class **MahasiswaDemo09**, kemudian buatlah sebuah objek **MahasiswaBerprestasi09** dan buatlah 5 objek mahasiswa kemudian tambahkan semua objek mahasiswa tersebut dengan memanggil fungsi tambah pada objek **MahasiswaBerprestasi09**. 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 MahasiswaDemo09 {

public static void main(String[] args) {

MahasiswaBerprestasi09 list = new MahasiswaBerprestasi09();

Mahasiswa09 m1 = new Mahasiswa09("123", "Zidan", "2A", 3.2);

Mahasiswa09 m2 = new Mahasiswa09("124", "Ayu", "2A", 3.5);

Mahasiswa09 m3 = new Mahasiswa09("125", "Sofi", "2A", 3.1);

Mahasiswa09 m4 = new Mahasiswa09("126", "Sita", "2A", 3.9);

Mahasiswa09 m5 = new Mahasiswa09("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): ");

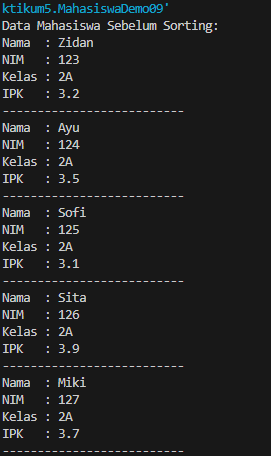
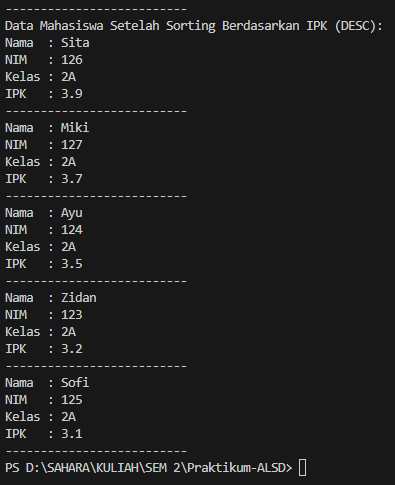
list.bubbleSort();

list.tampil();

}

}

1. Run program



* **Pertanyaan**

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

* Dalam bubble sort, element terakhir akan otomatis berada di posisi yang benar setelah iterasi luar. Maka, cukup melakukan iterasi sebanyak 1 kali untuk memastikan semua elemen telah terurut.

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

* Karena setiap iterasi luar **i** akan menempatkan elemen terbesar yang tersisa ke posisi akhir, bagian yang sudah urut tidak perlu dibandingkan lagi. Jumlah iterasi dalam **j** akan berkurang seiring bertambahnya **i**, untuk menghindari perbandingan yang tidak perlu.

1. Jika banyak data di dalam listMhs adalah 50, maka berapakali perulangan i akan berlangsung? Dan ada berapa Tahap bubble sort yang ditempuh?

* Perulangan luar **i** akan berjalan sebanyak 50 – 1 = 49 kali.
* Perulangan dalam **j** pada iterasi pertama berjalan 49 kali, lalu berkurang satu di setiap iterasi.

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

* Kode program

import java.util.Scanner;

public class MahasiswaDemo09 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

MahasiswaBerprestasi09 list = new MahasiswaBerprestasi09();

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();

Mahasiswa09 m = new Mahasiswa09(nim, nama, kelas, ipk);

list.tambah(m);

}

System.out.println("\nData mahasiswa sebelum Sorting: ");

list.tampil();

System.out.println("Data Mahasiswa Setelah Sorting Berdasarkan IPK (DESC): ");

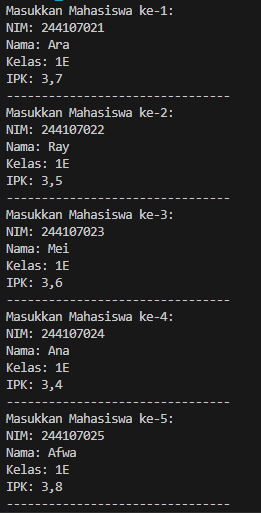
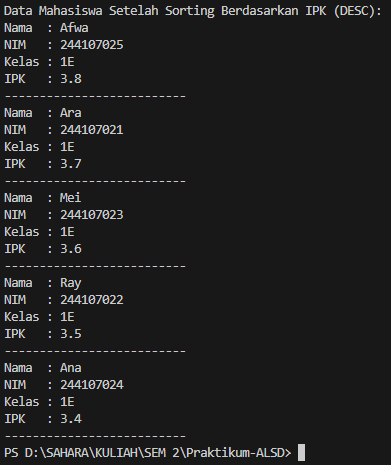
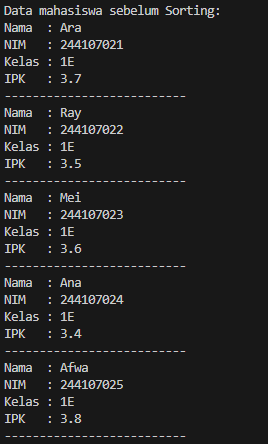
list.bubbleSort();

list.tampil();

}

}

* Run program



1. ***Selection Sort***
2. Lihat kembali class **MahasiswaBerprestasi09**, 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;

}

}

Mahasiswa09 tmp = listMhs[idxMin];

listMhs[idxMin] = listMhs[i];

listMhs[i] = tmp;

}

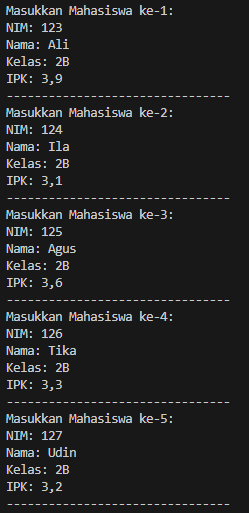
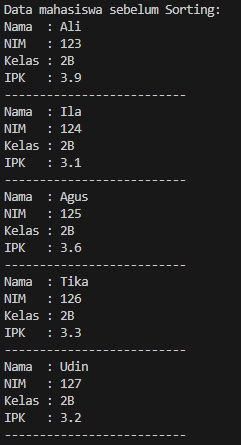
}

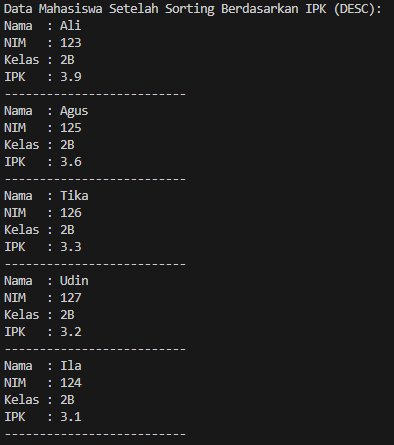
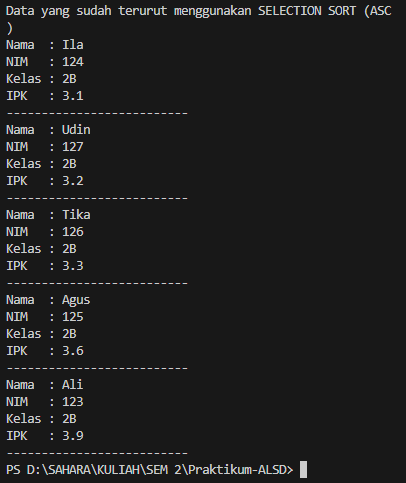
1. Buka kembali class **MahasiswaDemo09**, dan di dalam method main() tambahkan baris program untuk memanggil method selectionSort(), kemudian panggil method tampil() untuk menampilkan data yang sudah diurutkan!

System.out.println("Data yang sudah terurut menggunakan SELECTION SORT (ASC)");

list.selectionSort();

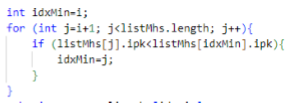
list.tampil();

1. Run program



* **Pertanyaan**

1. Dalam method selection sort, terdapat baris program. Untuk apakah proses tersebut?



* Berfungsi untuk mencari indeks elemen dengan nilai terkecil dalam array. Dalam selection sort digunakan untuk menemukan elemen terkecil sebelum dilakukan pertukaran. Dengan cara memilih elemen terkecil secara bertahap.

1. ***Insertion Sort***
2. Pada class **MahasiswaBerprestasi09**, 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++) {

Mahasiswa09 temp = listMhs[i];

int j=i;

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

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

j--;

}

listMhs[j] = temp;

}

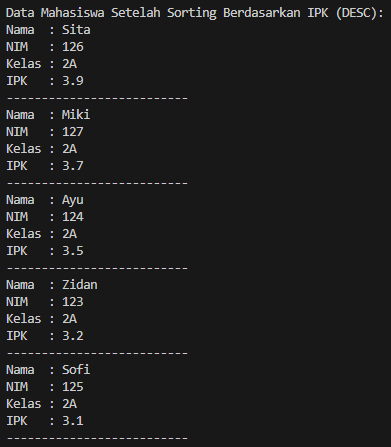
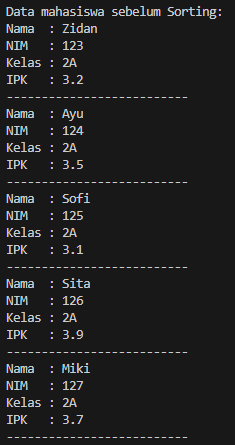
}

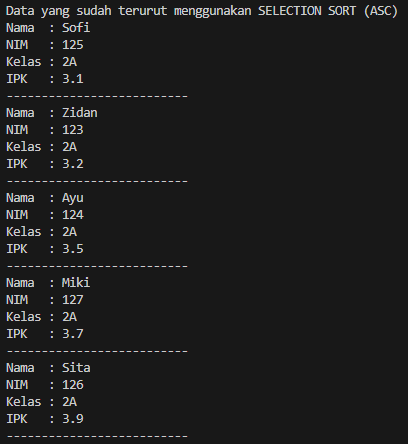
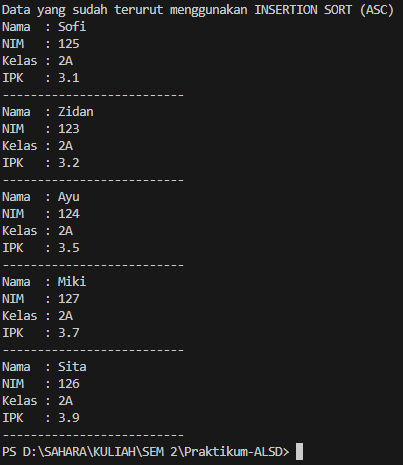
1. Buka kembali class **MahasiswaDemo09**, dan di dalam method main() tambahkan baris program untuk memanggil method insertionSort() dan tampil ()

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

list.insertionSort();

list.tampil();

1. Run program



* **Pertanyaan :**

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

* Kode program

void insertionSort() {

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

Mahasiswa09 temp = listMhs[i];

int j = i;

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

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

j--;

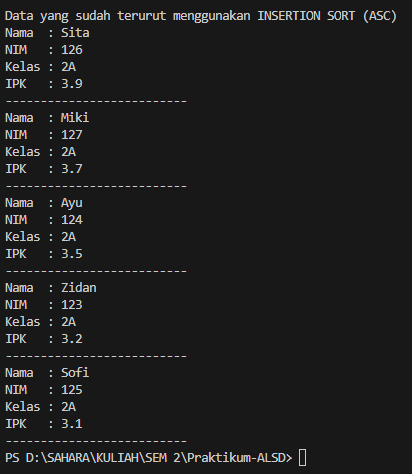
}

listMhs[j] = temp;

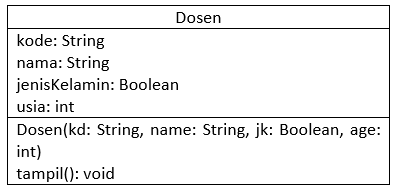
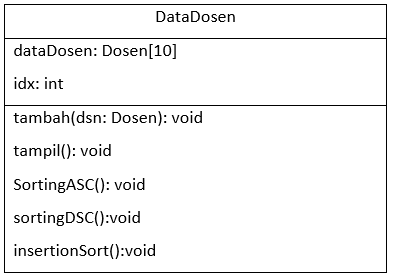
}

}

* Outputnya



* 1. **Latihan Praktikum**

1. 

* **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 **selection sort** atau **insertion sort**.
* **Kode programnya :**
* Buat class **dosen09.java**

package Praktikum5;

public class dosen09 {

String kode;

String nama;

boolean jenisKelamin;

int usia;

dosen09 (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);

System.out.println("Usia : " + usia);

}

}

* Buat class **dataDosen09.java**

package Praktikum5;

public class dataDosen09 {

dosen09[] dataDosen09 = new dosen09[10];

int idx;

void tambah(dosen09 dsn) {

if (idx < dataDosen09.length) {

dataDosen09[idx] = dsn;

idx++;

}else {

System.out.println("Data Dosen SUdah Banyak!");

}

}

//menampilkan data dosen

void tampil() {

if (idx == 0) {

System.out.println("Tidak Ada Data Dosen");

return;

}

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

dataDosen09[i].tampil();

System.out.println();

}

}

//mengurutkan data dosen secara ascending (asc)

void sortingASC() {

for (int i = 0; i < idx - 1; i++) {

for (int j = 0; j < idx - 1 - i; j++) {

if (dataDosen09[j].usia > dataDosen09[j+1].usia) {

dosen09 temp = dataDosen09[j];

dataDosen09[j] = dataDosen09[j+1];

dataDosen09[j+1] = temp;

}

}

}

}

//mengurutkan data dosen secara descending (dsc)

void sortingDSC() {

for (int i = 0; i < idx - 1; i++) {

for (int j = 0; j < idx - 1 - i; j++) {

if (dataDosen09[j].usia < dataDosen09[j+1].usia) {

dosen09 temp = dataDosen09[j];

dataDosen09[j] = dataDosen09[j+1];

dataDosen09[j+1] = temp;

}

}

}

}

}

* Buat class **dosenDemo09.java**

package Praktikum5;

import java.util.Scanner;

public class dosenDemo09 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

dataDosen09 datadsn = new dataDosen09();

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 (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:

for (int i = 0; i < 10; i++) {

System.out.println("Masukkan data dosen ke-" + (i+1) + ":");

System.out.print("Masukkan kode dosen: ");

String kode = sc.nextLine();

System.out.print("Masukkan nama dosen: ");

String nama = sc.nextLine();

System.out.print("Masukkan jenis Kelamin (Perempuan / Laki-laki) : ");

char jk = sc.next().charAt(0);

boolean jenisKelamin = (jk == '1' || jk == 'p');

System.out.print("Masukkan usia dosen: ");

int usia = sc.nextInt();

sc.nextLine();

System.out.println("-------------------------------------------------------------");

dosen09 dsn = new dosen09(kode, nama, jenisKelamin, usia);

datadsn.tambah(dsn);

}

break;

case 2:

datadsn.tampil();

break;

case 3:

datadsn.sortingASC();

System.out.println("Data Dosen Telah Diurutkan Secara Ascending");

datadsn.tampil();

break;

case 4:

datadsn.sortingDSC();

System.out.println("Data Dosen Telah Diurutkan Secara Descending");

datadsn.tampil();

break;

default:

System.out.println("Pilihan Tidak Valid. Silahkan Coba Lagi");

}

}while (pilihan != 5);

}

}

datadsn.tampil();

break;

default:

System.out.println("Pilihan Tidak Valid. Silahkan Coba Lagi");

}

}while (pilihan != 5);

}

}

* **Outputnya :**

