LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN STRUKTUR DATA
PRAKTIKUM 6

Oleh:

RANDA HERU KUSUMA

NIM. 2341760009

SIB-1F / 25

D-IV SISTEM INFORMASI BISNIS

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

Link github: https://github.com/randaheru/Praktikum6.git

5.2.1 Langkah-langkah Percobaan

Class Mahasiswa

```
public class Mahasiswa {
    String nama;
    int thnMasuk, umur;
    double ipk;
```

Buat sebuah class dengan nama Mahasiswa dan mendeklasi nama, thnMasuk, umur, dan ipk

```
Mahasiswa(String n, int t, int u, double i) {
    nama = n;
    thnMasuk = t;
    umur = u;
    ipk = i;
}
```

Tambahkan atribut seperti nama, thnMasuk, umur, ipk dan ditambahkan conctructor berparameter n, t, u, i

```
void tampil() {
    System.out.println("Nama = " + nama);
    System.out.println("Tahun Masuk = " + thnMasuk);
    System.out.println("Umur = " + umur);
    System.out.println("IPK = " + ipk);
```

Langkah selajutnya dendefinisikan method publik void bernama tampil. Di dalam method tampil, kode digunakan untuk mencetak nilai variabel-variabel class.

Class MahasiswaBerprestasi

```
public class DaftarMahasiswaBerprestasi {
    Mahasiswa listMhs[] = new Mahasiswa[5];
    int idx;
```

Langkah awal buat class DaftarMahasiswafBerpestasi

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

Tambah method publik void bernama tambah. Method ini digunakan untuk menambahkan objek Mahasiswa baru ke dalam array listMhs.

```
void tampil() {
    for(int i = 0; i < idx; i++) {
        listMhs[i].tampil();
        System.out.println(x:"--------------------------------");
    }
}
```

Mendefinisikan method publik void bernama tampil. Method yang digunakan untuk menampilkan data mahasiswa berprestasi yang ada di dalam array listMhs.

```
void bubbleSort90() {
    for(int i = 0; i < idx - 1; i++) {
        for(int j = 1; j < idx - i; j++) {
            if(listMhs[j].ipk > listMhs[j - 1].ipk) {
                Mahasiswa tmp = listMhs[j];
                listMhs[j] = listMhs[j - 1];
                listMhs[j - 1] = tmp;

            }
```

Mendefinisikan method publik void bernama bubbleSort90. Method ini digunakan untuk mengurutkan data mahasiswa berdasarkan IPK secara descending (nilai IPK tertinggi di awal) menggunakan algoritma bubble sort.

Class Main

```
public class Main {
    Run | Debug
```

Lalu buat class Main

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

Mendefinisikan method publik statis void bernama main

```
DaftarMahasiswaBerprestasi list = new DaftarMahasiswaBerprestasi()
Mahasiswa m1 = new Mahasiswa(n:"Nusa", t:2017, u:25, i:3);
Mahasiswa m2 = new Mahasiswa(n:"Rara", t:2012, u:19, i:4);
Mahasiswa m3 = new Mahasiswa(n:"Dompu", t:2018, u:19, i:3.5);
Mahasiswa m4 = new Mahasiswa(n:"Abdul", t:2017, u:23, i:2);
Mahasiswa m5 = new Mahasiswa(n:"Ummi", t:2019, u:21, i:3.75);
```

Lalu membuat 5 objek mahasiswa

```
list.tambah(m1);
list.tambah(m2);
list.tambah(m3);
list.tambah(m4);
list.tambah(m5);
```

kemudian tambah semua objek mahasiswa tersebut dengan memanggil fungsi tambah pada objek DaftarMahasiswaBerprestasi.

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

Memanggil method tampil dari objek list untuk menampilkan data mahasiswa sebelum sorting.

```
System.out.println(x:"Data mahasiswa setelah sorting desc berda
list.bubbleSort90();
list.tampil();
```

Memanggil method bubbleSort90 dari objek list untuk mengurutkan data mahasiswa berdasarkan IPK secara descending.

```
Data mahasiswa sebelum sorting =
Nama = Nusa
Tahun Masuk = 2017
Umur = 25
IPK = 3.0
--------------------------------
Nama = Rara
Tahun Masuk = 2012
Umur = 19
IPK = 4.0
--------------------------------
Nama = Dompu
Tahun Masuk = 2018
Umur = 19
IPK = 3.5
--------------------------------
Nama = Abdul
Tahun Masuk = 2017
Umur = 23
IPK = 2.0
--------------------------------
Nama = Ummi
Tahun Masuk = 2019
Umur = 21
```

```
Data mahasiswa setelah sorting desc berdasarkan ipk
Nama = Rara
Tahun Masuk = 2012
Umur = 19
IPK = 4.0
--------------------------------
Nama = Ummi
Tahun Masuk = 2019
Umur = 21
IPK = 3.75
--------------------------------
Nama = Dompu
Tahun Masuk = 2018
Umur = 19
IPK = 3.5
--------------------------------
Nama = Nusa
Tahun Masuk = 2017
Umur = 25
IPK = 3.0
--------------------------------
Nama = Abdul
Tahun Masuk = 2017
Umur = 23
IPK = 2.0
--------------------------------
```

5.2.3 Pertanyaan

1. Terdapat di method apakah proses bubble sort?

terdapat dalam metode bubbleSort90(). Dalam metode tersebut, dilakukan iterasi untuk membandingkan dua elemen sekaligus dan menukar posisi jika diperlukan. Proses ini terus diulangi hingga seluruh elemen telah terurut secara ascending berdasarkan nilai IPK (Indeks Prestasi Kumulatif) mahasiswa.

2. Di dalam method bubbleSort(), terdapat baris program seperti di bawah ini: Untuk apakah proses tersebut?

```
29                  if(listMhs[j].ipk > listMhs[j-1].ipk){
30                      //di bawah ini proses swap atau penukaran
31                      Mahasiswa tmp = listMhs[j];
32                      listMhs[j] = listMhs[j-1];
33                      listMhs[j-1] = tmp;
34                  }
35              }
```

Kode di atas merupakan bagian dari algoritma bubble sort yang digunakan untuk menukar posisi dua elemen jika nilai IPK elemen saat ini lebih besar daripada nilai IPK elemen sebelumnya

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

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

a. Apakah perbedaan antara kegunaan perulangan i dan perulangan j?

Perulangan i digunakan untuk melacak iterasi utama di seluruh array. Sedangkan perulangan j digunakan sebagai iterasi dalam satu siklus perulangan i untuk membandingkan dan menukar posisi dua elemen yang berdekatan jika perlu.

b. Mengapa syarat dari perulangan i adalah i

Karena dalam algoritma bubble sort, kita tidak perlu membandingkan elemen terakhir dengan elemen setelahnya. Oleh karena itu, iterasi i hanya perlu berjalan hingga elemen kedua terakhir.

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

karena setiap iterasi i, elemen terakhir dalam array (setelah iterasi i ke-i) akan terurut dengan benar, sehingga kita tidak perlu membandingkannya lagi

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

Jika banyaknya data di dalam **listMhs** adalah 50, maka perulangan i akan berlangsung sebanyak 49 kali, karena perulangan i berjalan dari 0 hingga 48 (**listMhs.length - 1**).

5.3.1. Langkah-langkah Percobaan.

```
void selectionSort() {
    for (int i = 0; i < idx - 1; i++) {
        int idxMin = i;
        for (int j = i + 1; j < idx; j++) {
            if (listMhs[j].ipk < listMhs[idxMin].ipk) {
                idxMin = j;
            }
        }
        // swap
        Mahasiswa tmp = listMhs[idxMin];
        listMhs[idxMin] = listMhs[i];
        listMhs[i] = tmp;
    }
```

Ke class MahasiswaBeprestasi tambahkan method selectionSort() di dalamnya! Method ini juga akan melakukan proses sorting secara ascending, tetapi menggunakan pendekatan selection sort.

```
System.out.println(x:"Data mahasiswa setelah sorting asc berdasark
list.selectionSort();
list.tampil();
```

lalu buka class Main, dan ditambah method main() tambahkan baris program untuk memanggil method selectionSort()

```
Data mahasiswa sebelum sorting =
Nama = Nusa
Tahun Masuk = 2017
Umur = 25
IPK = 3.0
---------------------------------
Nama = Rara
Tahun Masuk = 2012
Umur = 19
IPK = 4.0
---------------------------------
Nama = Dompu
Tahun Masuk = 2018
Umur = 19
IPK = 3.5
---------------------------------
Nama = Abdul
Tahun Masuk = 2017
Umur = 23
IPK = 2.0
---------------------------------
Nama = Ummi
Tahun Masuk = 2019
Umur = 21
IPK = 3.75
---------------------------------
```

```
Data mahasiswa setelah sorting asc berdasarkan ipk
Nama = Abdul
Tahun Masuk = 2017
Umur = 23
IPK = 2.0
---------------------------------
Nama = Nusa
Tahun Masuk = 2017
Umur = 25
IPK = 3.0
---------------------------------
Nama = Dompu
Tahun Masuk = 2018
Umur = 19
IPK = 3.5
---------------------------------
Nama = Ummi
Tahun Masuk = 2019
Umur = 21
IPK = 3.75
---------------------------------
Nama = Rara
Tahun Masuk = 2012
Umur = 19
IPK = 4.0
```

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

```
42                  int idxMin = i;
43                  for(int j=i+1; j<listMhs.length; j++){
44                      if(listMhs[j].ipk < listMhs[idxMin].ipk){
45                          idxMin = j;
46                      }
47                  }
```

Untuk apakah proses tersebut, jelaskan!

5.4.1 Langkah-langkah Percobaan

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

Masuk ke class MahasiswaBerprestaasi menambahkan method insertionSort(). Method ini juga akan melakukan proses sorting secara ascending.

```
System.out.println(x:"Data mahasiswa setelah sorting asc berdasark
list. insertionSort();
list. tampil();
```

Lalu menabahkan di class Main, dan di dalam method main() tambahkan baris program untuk memanggil method insertionSort()

```
Data mahasiswa sebelum sorting =
Nama = Nusa
Tahun Masuk = 2017
Umur = 25
IPK = 3.0
---------------------------------
Nama = Rara
Tahun Masuk = 2012
Umur = 19
IPK = 4.0
---------------------------------
Nama = Dompu
Tahun Masuk = 2018
Umur = 19
IPK = 3.5
---------------------------------
Nama = Abdul
Tahun Masuk = 2017
Umur = 23
IPK = 2.0
---------------------------------
Nama = Ummi
Tahun Masuk = 2019
Umur = 21
IPK = 3.75
---------------------------------
```

```
Data mahasiswa setelah sorting asc berdasarkan ipk
Nama = Abdul
Tahun Masuk = 2017
Umur = 23
IPK = 2.0
--------------------------------
Nama = Nusa
Tahun Masuk = 2017
Umur = 25
IPK = 3.0
--------------------------------
Nama = Dompu
Tahun Masuk = 2018
Umur = 19
IPK = 3.5
--------------------------------
Nama = Ummi
Tahun Masuk = 2019
Umur = 21
IPK = 3.75
--------------------------------
Nama = Rara
Tahun Masuk = 2012
Umur = 19
IPK = 4.0
```

5.4.3 Pertanyaan

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

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

Mengubah kondisi **listMhs[j - 1].ipk > temp.ipk** menjadi **listMhs[j - 1].ipk < temp.ipk**, maka proses sorting akan dilakukan secara descending nilai IPK.