

LAPORAN HASIL PRAKTIKUM

ALGORITMA DAN STRUKTUR DATA

JOBSHEET 6



OLEH :

IVAN RIZAL AHMADI

NIM. 2341760128


SIB-1F/13

D-IV SISTEM INFORMASI BISNIS

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

5.2.1 Langkah-langkah Percobaan

1. Buat project baru dengan nama “bubble-selection-insertion”, kemudian buat package dengan nama “jobsheet6”.
2. Buatlah sebuah class dengan nama Mahasiswa  Mahasiswa13.java
3. Sesuaikan class Mahasiswa dengan melihat class diagram di atas dengan menambahkan attribute, konstruktor, dan fungsi atau method. Untuk lebih jelasnya class tersebut dapat dilihat pada potongan kode di bawah ini

```
1 public class Mahasiswa13 {
2     String nama;
3     int thnMasuk, umur;
4     double ipk;
5
6     Mahasiswa13(String n, int t, int u, double i) {
7         nama = n;
8         thnMasuk = t;
9         umur = u;
10        ipk = i;
11    }
12
13    void tampil() {
14        System.out.println("Nama      = " + nama);
15        System.out.println("Tahun Masuk = " + thnMasuk);
16        System.out.println("Umur      = " + umur);
17        System.out.println("IPK       = " + ipk);
18    }
19 }
```

4. Buat class DaftarMahasiswaBerprestasi seperti di bawah ini!

```
public class DaftarMahasiswaBerprestasi {
    Mahasiswa13 listMhs[] = new Mahasiswa13[5];
    int idx;
```

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

```
// method tambah()
void tambah(Mahasiswa13 m) {
    if (idx < listMhs.length) {
        listMhs[idx] = m; // selama id belum mencapai lengt, maka object mahasiswa dapat ters dimasukkan
        // | | | | | // kedalam array list mhs
        idx++;
    } else {
        System.out.println(x:"Data Sudah Penuh!!");
    }
}
```

6. 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.

```
// method tampil()
void tampil() {
    for (Mahasiswa13 m : listMhs) { // selama jumlah mahasiswa sesuai dengan elemen yang diset maka akan terus
        // ditampilkan
        m.tampil();
        System.out.println(x: "-----");
    }
}
```

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

```
// method bubbleSort()
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) {
                // proses dibawah ini adalah proses swap atau pertukaran (bubble sort)
                Mahasiswa13 tmp = listMhs[j];
                listMhs[j] = listMhs[j - 1];
                listMhs[j - 1] = tmp;
            }
        }
    }
}
```

8. Buat class Main dan didalamnya buat method main() seperti di bawah ini!

```
public class MahasiswaMain13 {
    Run | Debug
    public static void main(String[] args) {
```

9. **Di dalam method main(),** buatlah sebuah objek DaftarMahasiswaBerprestasi dan buatlah 5 objek mahasiswa kemudian tambahkan semua objek mahasiswa tersebut dengan memanggil fungsi tambah pada objek DaftarMahasiswaBerprestasi. 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 MahasiswaMain13 {
    Run | Debug
    public static void main(String[] args) {
        DaftarMahasiswaBerprestasi list = new DaftarMahasiswaBerprestasi();
        Mahasiswa13 m1 = new Mahasiswa13(n:"Nusa", t:2017, u:25, i:3);
        Mahasiswa13 m2 = new Mahasiswa13(n:"Rara", t:2012, u:19, i:4);
        Mahasiswa13 m3 = new Mahasiswa13(n:"Dompur", t:2018, u:19, i:3.5);
        Mahasiswa13 m4 = new Mahasiswa13(n:"Abdul", t:2017, u:23, i:2);
        Mahasiswa13 m5 = new Mahasiswa13(n:"Ummi", t:2019, u:21, i:3.75);

        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();
        System.out.println(x:"Data mahasiswa setelah sorting desc berdasarkan ipk (menggunakan Bobble Short)");
        list.bubbleSort();
        list.tampil();
    }
}

```

5.2.2 Verifikasi Hasil Percobaan

Cocokan hasilnya dengan yang terdapat pada tampilan di bawah ini

```

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      = Dompur
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 desc berdasarkan ipk (menggunakan Bobble Short)
Nama      = Rara
Tahun Masuk = 2012
Umur      = 19
IPK       = 4.0
-----
Nama      = Ummi
Tahun Masuk = 2019
Umur      = 21
IPK       = 3.75
-----
Nama      = Dampu
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
-----
PS C:\Users\ivanr\OneDrive\Desktop\Jobsheet6> c++; cd 'c:\Users\ivanr\OneDrive\Desktop\Jobsheet6

```

5.2.3 Pertanyaan

1. Terdapat di method apakah proses bubble sort?

Terdapat pada method bubblesort();, untuk lebih lengkapnya adalah seperti ss an saya berikut

```

// method bubbleSort()
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) {
                // proses dibawah ini adalah proses swap atau pertukaran (bubble short)
                Mahasiswa13 tmp = listMhs[j];
                listMhs[j] = listMhs[j - 1];
                listMhs[j - 1] = tmp;
            }
        }
    }
}

```

2. Di dalam method bubbleSort(), terdapat baris program seperti di bawah ini:

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			}		

Untuk apakah proses tersebut? = Proses tersebut adalah proses "pemilihan / if" dimana terdapat sebuah kondisi jika (listMhs[j].ipk > listMhs[j-1].ipk) terpenuhi, maka akan dilakukan proses swap atau pertukaran nilai dengan syarat yang telah diberikan. Yang mana syarat tersebut adalah (>)

lebih dari, maka nilai yang akan diurutkan akan keluar descending (dari angka terbesar ke angka yang terkecil)

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++){
```

- Apakah perbedaan antara kegunaan perulangan i dan perulangan j? = Perulangan i disebut sebagai perulangan luar yang bertujuan agar proses pertukaran (swap) pada perulangan j tetap berlanjut hingga semua bilangan sudah diurutkan. Perulangan j disebut sebagai perulangan dalam yang bertugas melakukan pertukaran (swap) nilai secara terus menerus hingga bilangan sudah terurut sesuai dengan syarat kondisi (ascending atau descending)
- Mengapa syarat dari perulangan i adalah $i < \text{listMhs.length} - 1$? = Agar batas dari perulangan i (perulangan dalam) memiliki rentang panjang sejumlah panjang array, listMhs dikurangi 1 ketika melakukan perulangan dari perulangan swapping yang dilakukan oleh perulangan j
- Mengapa syarat dari perulangan j adalah $j < \text{listMhs.length} - i$? = Agar batas dari perulangan j (perulangan dalam) memiliki rentang panjang sejumlah panjang array, listMhs dikurangi i ketika melakukan swapping / penukaran nilai dari array listMhs agar bisa urut
- Jika banyak data di dalam listMhs adalah 50, maka berapakah perulangan i akan berlangsung? Dan ada berapa **Tahap** bubble sort yang ditempuh? =
 - perulangan yang akan berlangsung pada i apabila length listMhs 50 adalah sebanyak length dari array listMhs - 1 ($\text{listMhs.length} - 1$) sampai proses swapping atau tahap bubbleshot pada perulangan j habis / sudah memenuhi kondisi terurut(jika belum perulangan i akan terus dilakukan). dan jika dalam bentuk angka sesungguhnya perulangan i akan berlangsung sebanyak 49 kali.
 - tahap bubbleshot akan ditempuh jika length listMhs 50 adalah sebanyak length dari array listMhs - i ($\text{listMhs.length} - i$) sampai proses swapping atau penukaran nilai dari elemen - elemen array listMhs sudah memenuhi kondisi terurut(jika belum maka swapping akan terus dilakukan). dan jika dalam bentuk angka sesungguhnya perulangan j(tahap bubblesort) akan ditempuh sebanyak 1273 kali

5.3.1. Langkah-langkah Percobaan.

- Lihat kembali class DaftarMahasiswaBerprestasi, 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;
            }
        }
        // proses dibawah ini adalah proses swap atau pertukaran (selection sort)
        Mahasiswa13 tmp = listMhs[idxMin];
        listMhs[idxMin] = listMhs[i];
        listMhs[i] = tmp;
    }
}

```

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

```

System.out.println(x:"Data mahasiswa setelah sorting asc berdasarkan ipk (menggunakan Selection Short)");
list.selectionSort();
list.tampil();

```

- Coba jalankan kembali class Main, dan amati hasilnya! Apakah kini data mahasiswa telah tampil urut menaik berdasar ipk? = iya, setelah saya menambahkan method selectionSort(), output yang ditampilkan program menjadi lebih tersusun ipk nya.

```

Data mahasiswa setelah sorting asc berdasarkan ipk (menggunakan Bobble Short)
Nama      = Rara
Tahun Masuk = 2012
Umur      = 19
IPK       = 4.0
-----
Nama      = Ummi
Tahun Masuk = 2019
Umur      = 21
IPK       = 3.75
-----
Nama      = Dompur
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
-----

```

5.3.2. Verifikasi Hasil Percobaan

Pastikan output yang ditampilkan sudah benar seperti di bawah ini

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 (menggunakan Bobble Short)

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

5.3.3. 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     }
```

1. Untuk apakah proses tersebut, jelaskan! =

- Proses tersebut bertujuan untuk mencari nilai elemen min pada semua nilai elemen pada array yang seharusnya (minimal pada pertama). kemudian elemen array tersebut ditetapkan
- menemukan sebuah elemen array yang memiliki nilai kecil dari index kedua dari elemen awal jika terkecil, setelah itu melakukan penukaran elemen tersebut dengan elemen array pada posisi index kedua, kemudian untuk menetapkan elemen array tersebut ditambah dengan elemen array yang sebelumnya
- dari program diatas nilai terkecil atau min dideklarasikan dengan idxmin, lalu masuk perulangan kedua untuk memberikan syarat pemilihan if (listMhs[j].ipk < listMhs[idxMin].ipk) pada nilai tersebut dan jika memenuhi kondisi maka nilai terkecil sudah ditemukan

5.4 Mengurutkan Data Mahasiswa Berdasarkan IPK Menggunakan Insertion Sort Waktu : 30 menit

Yang terakhir akan diimplementasikan Teknik sorting menggunakan Insertion Sort, dengan mengurutkan IPK mahasiswa secara ascending.

5.4.1 Langkah-langkah Percobaan

1. Lihat kembali class DaftarMahasiswaBerprestasi, dan tambahkan method insertionSort() di dalamnya. Method ini juga akan melakukan proses sorting secara **ascending**, tetapi menggunakan pendekatan Insertion Sort.

```
// method insertionSort()
void insertionSort() {
    for (int i = 1; i < listMhs.length; i++) {
        Mahasiswa13 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 Main, dan di dalam method main() tambahkan baris program untuk memanggil method insertionSort() tersebut!

```
System.out.println();
System.out.println(x:"Data mahasiswa setelah sorting asc berdasarkan ipk (menggunakan Insertion Short)");
list.insertionSort();
list.tampil();
```

- Coba jalankan kembali class Main, dan amati hasilnya! Apakah kini data mahasiswa telah tampil urut menaik berdasar ipk? Saya memiliki hasil program seperti yang ada dibawah ini =

5.4.2 Verifikasi Hasil Percobaan

Pastikan output yang ditampilkan sudah benar seperti di bawah ini

```
Data mahasiswa setelah sorting asc berdasarkan ipk (menggunakan Insertion Short)
Nama      = Abdul
Tahun Masuk = 2017
Umur       = 23
IPK        = 2.0
-----
Nama      = Nusa
Tahun Masuk = 2017
Umur       = 25
IPK        = 3.0
-----
Nama      = Dampu
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.

disini saya melakukan modifikasi di class daftarMahasiswaBerprestasi dan di main class, untuk perubahan pada class daftarMahasiswaBerprestasi menambahkan parameter sebagai berikut\

```
// method insertionSort()
void insertionSort(boolean asc) {
    for (int i = 1; i < listMhs.length; i++) {
        Mahasiswa13 temp = listMhs[i];
        int j = i;
        if(asc){
            while(j > 0 && listMhs[j - 1].ipk > temp.ipk){
                listMhs[j] = listMhs[j-1];
                j--;
            }
        }else{
            while(j > 0 && listMhs[j - 1].ipk < temp.ipk){
                listMhs[j] = listMhs[j-1];
                j--;
            }
        }
        listMhs[j] = temp;
    }
}
```

Dan pada main class nya adalah sebagai berikut

```
System.out.println();
System.out.println(x:"Data mahasiswa setelah sorting asc berdasarkan ipk (menggunakan Insertion Short)");
list.insertionSort(asc:true);
list.tampil();

System.out.println();
System.out.println(x:"Data mahasiswa setelah sorting dsc berdasarkan ipk (menggunakan Insertion Short)");
list.insertionSort(asc:false);
list.tampil();
```

Sehingga diperoleh hasil sebagai berikut =

```
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      = Dompus
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 (menggunakan Insertion Short)

Nama = Abdul
Tahun Masuk = 2017
Umur = 23
IPK = 2.0

Nama = Nusa
Tahun Masuk = 2017
Umur = 25
IPK = 3.0

Nama = Dampu
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

Data mahasiswa setelah sorting dsc berdasarkan ipk (menggunakan Insertion Short)

Nama = Rara
Tahun Masuk = 2012
Umur = 19
IPK = 4.0

Nama = Ummi
Tahun Masuk = 2019
Umur = 21
IPK = 3.75

Nama = Dampu
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

PS C:\Users\ivanr\OneDrive\Desktop\Jobsheet6>