

**LAPORAN HASIL PRAKTIKUM  
ALGORITMA DAN STRUKTUR DATA  
JOBSHEET 6**



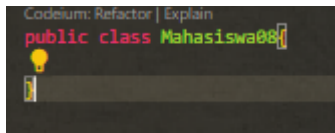
**Oleh:  
DZULFIKAR MUHAMMAD AL GHIFARI  
NIM. 2341760071  
SIB-1F / 08  
D-IV SISTEM INFORMASI BISNIS  
JURUSAN TEKNOLOGI INFORMASI  
POLITEKNIK NEGERI MALANG**

## PRAKTIKUM 5

### 5.2 PERCOBAAN

#### 5.2.1 LANGKAH LANGKAH PERCOBAAN

1. Membuat class Mahasiswa



```
Codeium: Refactor | Explain
public class Mahasiswa08 {
  }
```

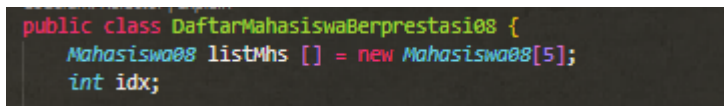
2. Menambahkan attribute, konstruktor, dan fungsi atau method



```
Mahasiswa08(String n, int t, int u, double i)
{
    nama = n;
    thnMasuk = t;
    umur = u;
    ipk = i;
}

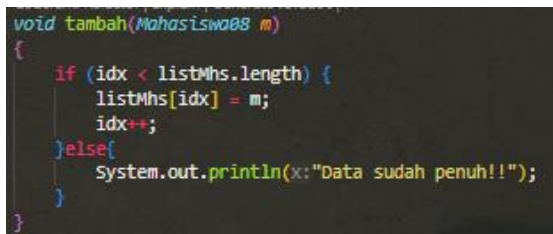
Codeium: Refactor | Explain | Generate Javadoc | X
void tampil()
{
    System.out.println("Nama = " + nama);
    System.out.println("Tahun Masuk = " + thnMasuk);
    System.out.println("Umur = " + umur);
    System.out.println("IPK = " + ipk);
}
```

3. Membuat class mahasiswa berprestasi



```
public class DaftarMahasiswaBerprestasi08 {
    Mahasiswa08 listMhs [] = new Mahasiswa08[5];
    int idx;
}
```

4. Menambahkan method tambah



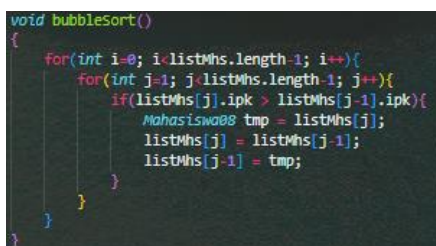
```
void tambah(Mahasiswa08 m)
{
    if (idx < listMhs.length) {
        listMhs[idx] = m;
        idx++;
    } else {
        System.out.println("Data sudah penuh!!");
    }
}
```

5. Menambahkan method tampil



```
void tampil()
{
    for (Mahasiswa08 m : listMhs) {
        m.tampil();
        System.out.println("-----");
    }
}
```

6. Menambahkan method bubbleSort



```
void bubbleSort()
{
    for (int i = 0; i < listMhs.length - 1; i++) {
        for (int j = 1; j < listMhs.length - 1; j++) {
            if (listMhs[j].ipk > listMhs[j-1].ipk) {
                Mahasiswa08 tmp = listMhs[j];
                listMhs[j] = listMhs[j-1];
                listMhs[j-1] = tmp;
            }
        }
    }
}
```

## 7. Menambahkan class main

```
package jobsheete;  
  
Codeium: Refactor | Explain  
public class Main {  
    Run | Debug | Codeium: Refactor | Explain | Generate Javadoc | X  
    public static void main(String[] args) {  
        DaftarMahasiswaBerprestasi08 list = new DaftarMahasiswaBerprestasi08();  
        Mahasiswa08 m1 = new Mahasiswa08(n:"Nusa", t:2017, u:25, i:3);  
        Mahasiswa08 m2 = new Mahasiswa08(n:"Rara", t:2012, u:19, i:4);  
        Mahasiswa08 m3 = new Mahasiswa08(n:"Dompur", t:2018, u:19, i:3.5);  
        Mahasiswa08 m4 = new Mahasiswa08(n:"Abdul", t:2017, u:23, i:2);  
        Mahasiswa08 m5 = new Mahasiswa08(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(x:"Data mahasiswa setelah sorting desc berdasarkan ipk");  
        list.bubbleSort();  
        list.tampil();  
    }  
}
```

## VERIFIKASI HASIL PERCOBAAN 5.2.2

```
f23d00e37f2fb157daf598aa59\redhat.java\jdt_ws\per te  
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
Nama = Rara
Tahun Masuk = 2012
Umur = 19
IPK = 4.0
-----
Nama = Dompus
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
-----
Nama = Ummi
Tahun Masuk = 2019
Umur = 21
IPK = 3.75
-----

```

### PERTANYAAN 3.2.3

1. Terdapat di method apakah proses bubble sort?
2. Di dalam method bubbleSort(), terdapat baris program seperti di bawah ini

```

    if(listMhs[j].ipk > listMhs[j-1].ipk){
        //di bawah ini proses swap atau penukaran
        Mahasiswa tmp = listMhs[j];
        listMhs[j] = listMhs[j-1];
        listMhs[j-1] = tmp;
    }
}

```

Untuk apakah proses tersebut?

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

```

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

```

- a. Apakah perbedaan antara kegunaan perulangan i dan perulangan j?
- b. Mengapa syarat dari perulangan i adalah `i<listMhs.length-1` ?
- c. Mengapa syarat dari perulangan j adalah `j<listMhs.length-i` ?
- d. Jika banyak data di dalam listMhs adalah 50, maka berapakah perulangan i akan berlangsung? Dan ada berapa **Tahap** bubble sort yang ditempuh?

Jawaban

1. Pada method bubbleSort

```
void bubbleSort()
{
    for(int i=0; i<listMhs.length-1; i++){
        for(int j=1; j<listMhs.length-1; j++){
            if(listMhs[j].ipk > listMhs[j-1].ipk){
                Mahasiswa088 tmp = listMhs[j];
                listMhs[j] = listMhs[j-1];
                listMhs[j-1] = tmp;
            }
        }
    }
}
```

2. Untuk menukar posisi angka atau swapping
3. A. perulangan i digunakan untuk menentukan elemen array yang akan di bandingkan, sedangkan perulangan j digunakan untuk membandingkan elemen array dengan elem di sampingnya  
B. Karena index perulangan dimulai dari 0, jika tidak dikurangi 1 maka literasi terakhir tidak akan menemukan index pada array  
C. Sebagai batas akhir adalah untuk menghindari pengecualian array out of bounds.  
D. perulangan akan berjalan sebanyak 50 kali pada perulangan luar, pada setiap perulangan luar akan melakukan perulangan dalam sebanyak 49 kali

#### LANGKAH LANGKAH PERCOBAAN 5.3.1

1. Menambahkan method selectionSort

```
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;
            }
        }

        Mahasiswa088 tmp = listMhs[idxMin];
        listMhs[idxMin] = listMhs[i];
        listMhs[i] = tmp;
    }
}
```

2. Modifikasi kode untuk memanggil method selectionSort

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

### VERIFIKASI HASIL PERCOBAAN 5.3.2

```
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 = Dampu
Tahun Masuk = 2018
Umur = 19
IPK = 3.5
-----
Nama = Ummi
Tahun Masuk = 2019
Umur = 21
IPK = 3.75
-----
Nama = Rara
Tahun Masuk = 2012
```

### PERTANYAAN 5.3.3

1. 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

1. Untuk menemukan indeks elemen dengan nilai IPK terkecil di listMhs  
Dimulai dengan inisialisasi idxMin; kemudian di dalam perulangan dilakukan pengecekan apakah pada ipk pada elemen j lebih kecil dari ipk elemen idxmin. Jika ya maka idx min akan di replace oleh index j

### LANGKAH LANGKAH PERCOBAAN 5.4.1

1. Menambahkan method insertionSort

```
void insertionSort() {
    for (int i = 1; i < listMhs.length; i++) {
        Mahasiswa08 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. Modifikasi kode untuk memanggil method insertionSort

#### VERIFIKASI HASIL PERCOBAAN 5.4.2

```
-----
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 = Dompus
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
-----
PS E:\1POLINEMA\2Genap 2023-2024\PraktikumAlgoritma\pertemuan6>
```

#### PERTANYAAN 5.4.3

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

Jawaban

1. Dengan mengubah kondisi pada while yang awalnya > menjadi <

```
listMhs[j - 1].ipk < temp.ipk)
```

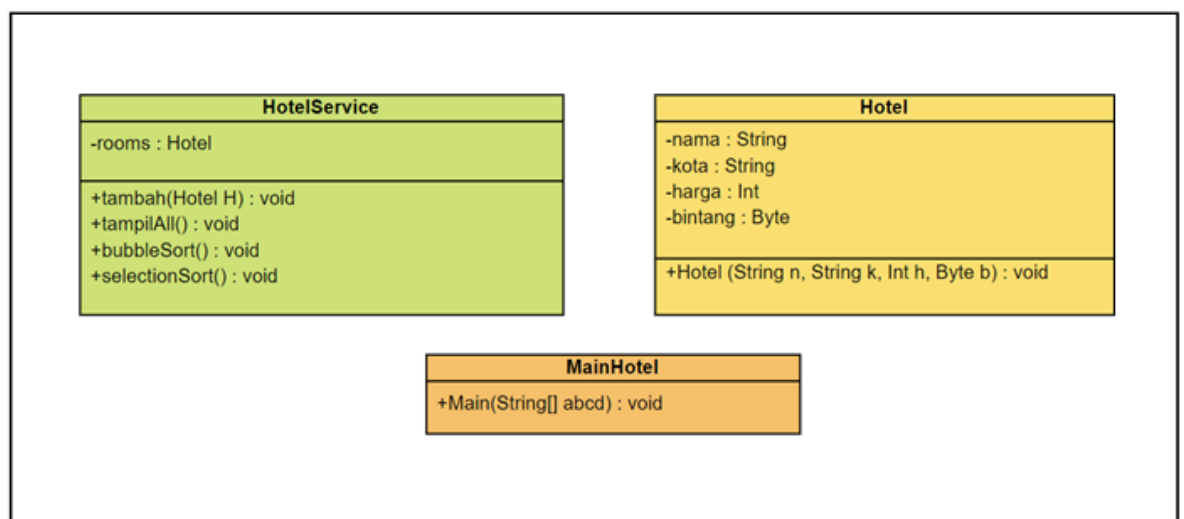
```
Coderum: Refactor | Explain | X
void insertionSort() {
    for (int i = 1; i < listMhs.length; i++) {
        Mahasiswa08 temp = listMhs[i];
        int j = i;
        while (j > 0 && listMhs[j - 1].ipk > temp.ipk) {
            listMhs[j] = listMhs[j - 1];
            j--;
        }
        listMhs[j] = temp;
    }
}
```

## 5.5 TUGAS

Sebuah platform travel yang menyediakan layanan pemesanan kebutuhan travelling sedang mengembangkan backend untuk sistem pemesanan/reservasi akomodasi (penginapan), salah satu fiturnya adalah menampilkan daftar penginapan yang tersedia berdasarkan pilihan filter yang diinginkan user. Daftar penginapan ini harus dapat disorting berdasarkan

1. Harga dimulai dari harga termurah ke harga tertinggi.
2. Rating bintang penginapan dari bintang tertinggi (5) ke terendah (1)

Buatlah proses sorting data untuk kedua filter tersebut dengan menggunakan algoritma bubble sort dan selection sort



Jawaban



## MainHotel

```
Codeium: Refactor | Explain
public class MainHotel08 {
    Run | Debug | Codeium: Refactor | Explain | Generate Javadoc | X
    public static void main(String[] args) {
        HotelService08 list = new HotelService08();
        Hotel08 m1 = new Hotel08(n:"RedDors", k:"Malang", h:25000, (byte)5);
        Hotel08 m5 = new Hotel08(n:"WhiteDors", k:"Batu", h:26000, (byte)3);
        Hotel08 m2 = new Hotel08(n:"BlakDors", k:"Surabaya", h:19000, (byte)4);
        Hotel08 m3 = new Hotel08(n:"BlueDors", k:"Kediri", h:20000, (byte)1);
        Hotel08 m4 = new Hotel08(n:"GreyDors", k:"Malang", h:22000, (byte)2);

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

        System.out.println(x:"Data hotel = ");
        list.tampilAll();

        System.out.println(x:"Data hotel dengan filter harga termurah = ");
        list.bubbleSort();
        list.tampilAll();

        System.out.println(x:"Data hotel dengan filter bintang tertinggi");
        list.selectionSort();
        list.tampilAll();
    }
}
```

## HotelService

```
Codeium: Refactor | Explain
public class HotelService08 {
    Hotel08 rooms[] = new Hotel08[5];
    int idx;

    Codeium: Refactor | Explain | Generate Javadoc | X
    void tambah(Hotel08 h) {
        if (idx < rooms.length) {
            rooms[idx] = h;
            idx++;
        } else {
            System.out.println(x:"Data sudah penuh!!");
        }
    }

    Codeium: Refactor | Explain | Generate Javadoc | X
    void tampilAll() {
        for (Hotel08 h : rooms) {
            h.tampil();
            System.out.println(x:"-----");
        }
    }

    Codeium: Refactor | Explain | Generate Javadoc | X
    void bubbleSort() {
        for (int i = 0; i < rooms.length - 1; i++) {
            for (int j = 1; j < rooms.length - i - 1; j++) {
                if (rooms[j].harga < rooms[j - 1].harga) {
                    Hotel08 tmp = rooms[j];
                    rooms[j] = rooms[j - 1];
                    rooms[j - 1] = tmp;
                }
            }
        }
    }

    Codeium: Refactor | Explain | Generate Javadoc | X
    void selectionSort() {
        for (int i = 0; i < rooms.length - 1; i++) {
            int idxMax = i;
            for (int j = i + 1; j < rooms.length; j++) {
                if (rooms[j].bintang > rooms[idxMax].bintang) {
                    idxMax = j;
                }
            }

            Hotel08 tmp = rooms[idxMax];
            rooms[idxMax] = rooms[i];
            rooms[i] = tmp;
        }
    }
}
```

## Hotel

```
Codeium: Refactor | Explain
public class Hotel08 {
    String nama, kota;
    int harga;
    byte bintang;

    Hotel08(String n, String k, int h, byte b)
    {
        nama = n;
        kota = k;
        harga = h;
        bintang = b;
    }

    Codeium: Refactor | Explain | Generate Javadoc | X
    void tampil()
    {
        System.out.println("Nama = "+ nama);
        System.out.println("Kota = "+ kota);
        System.out.println("Harga = "+ harga);
        System.out.println("Rating = "+ bintang);
    }
}
```

## Hasil compile

```
Data hotel =  
Nama = RedDors  
Kota = Malang  
Harga = 25000  
Rating = 5  
-----
```

```
Nama = BlakDors  
Kota = Surabaya  
Harga = 19000  
Rating = 4  
-----
```

```
Nama = BlueDors  
Kota = Kediri  
Harga = 20000  
Rating = 1  
-----
```

```
Nama = GreyDors  
Kota = Malang  
Harga = 22000  
Rating = 2  
-----
```

```
Nama = WhiteDors  
Kota = Batu  
Harga = 26000  
Rating = 3  
-----
```

```
Data hotel dengan filter bintang tertinggi  
Nama = RedDors  
Kota = Malang  
Harga = 25000  
Rating = 5  
-----
```

```
Nama = BlakDors  
Kota = Surabaya  
Harga = 19000  
Rating = 4  
-----
```

```
Nama = WhiteDors  
Kota = Batu  
Harga = 26000  
Rating = 3  
-----
```

```
Nama = GreyDors  
Kota = Malang  
Harga = 22000  
Rating = 2  
-----
```

```
Nama = BlueDors  
Kota = Kediri  
Harga = 20000  
Rating = 1  
-----
```

```
Data hotel dengan filter harga termurah =  
Nama = BlakDors  
Kota = Surabaya  
Harga = 19000  
Rating = 4  
-----
```

```
Nama = BlueDors  
Kota = Kediri  
Harga = 20000  
Rating = 1  
-----
```

```
Nama = GreyDors  
Kota = Malang  
Harga = 22000  
Rating = 2  
-----
```

```
Nama = RedDors  
Kota = Malang  
Harga = 25000  
Rating = 5  
-----
```

```
Nama = WhiteDors  
Kota = Batu  
Harga = 26000  
Rating = 3  
-----
```