

**LAPORAN TUGAS**  
**MATA KULIAH ALGORITMA DAN STRUKTUR DATA**

Dosen Pengampu : Triana Fatmawati, S.T, M.T

**PERTEMUAN - 6 Sorting**



**Nama : M. Zidna Billah Faza**  
**NIM : 2341760030**  
**Prodi : D-IV Sistem Informasi Bisnis**

**JURUSAN TEKNOLOGI INFORMASI**  
**POLITEKNIK NEGERI MALANG**  
**2023**

## PERCOBAAN 1 BUBBLE SORT

1. Buatlah sebuah class dengan nama Mahasiswa

 Mahasiswa\_18.java

2. 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 Mahasiswa_18 {
2
3      String nama;
4      int thnMasuk, umur;
5      double ipk;
6
7      Mahasiswa_18(String nama, int thnMasuk, int umur, double ipk) {
8          this.nama = nama;
9          this.thnMasuk = thnMasuk;
10         this.umur = umur;
11         this.ipk = ipk;
12     }
13
14     public void tampil() {
15         System.out.println("Nama      : " + nama);
16         System.out.println("Tahun Masuk : " + thnMasuk);
17         System.out.println("Umur      : " + umur);
18         System.out.println("IPK       : " + ipk);
19     }
20 }
```

3. Buat class DaftarMahasiswaBerprestasi seperti di bawah ini!

```
1  public class DaftarMahasiswaBerprestasi_18 {
2
3      Mahasiswa_18 listMhs[] = new Mahasiswa_18[5];
4      int idx;
5  }
```

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

```
1 void tambah(Mahasiswa_18 m) {  
2  
3     if (idx < listMhs.length) {  
4  
5         listMhs[idx] = m;  
6         idx++;  
7  
8     } else {  
9  
10        System.out.println("Data sudah penuh!!!");  
11  
12    }  
13  
14 }
```

5. 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.

```
1 void tampil() {  
2  
3     for (Mahasiswa_18 m : listMhs) {  
4         m.tampil();  
5         System.out.println("=====");  
6     }  
7  
8 }
```

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

```
1 void bubbleSort() {
2
3     for (int i = 0; i < listMhs.length; i++) {
4         for (int j = 1; j < listMhs.length; j++) {
5             if (listMhs[j].ipk > listMhs[j-1].ipk) {
6
7                 // Proses Swap atau Penukaran
8                 Mahasiswa_18 tmp = listMhs[j];
9                 listMhs[j] = listMhs[j-1];
10                listMhs[j-1] = tmp;
11            }
12        }
13    }
14 }
15
16 }
```

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

```
1 public class Main_18 {
2     public static void main(String[] args) {
3
```

8. 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.

```
DaftarMahasiswaBerprestasi_18 list = new DaftarMahasiswaBerprestasi_18();

Mahasiswa_18 m1 = new Mahasiswa_18("Nusa", 2017, 25, 3);
Mahasiswa_18 m2 = new Mahasiswa_18("Rara", 2012, 19, 4);
Mahasiswa_18 m3 = new Mahasiswa_18("Dompur", 2018, 19, 3.5);
Mahasiswa_18 m4 = new Mahasiswa_18("Abdul", 2017, 23, 2);
Mahasiswa_18 m5 = new Mahasiswa_18("Ummi", 2019, 21, 3.75);

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

System.out.println("Data Mahasiswa Sebelum Sorting");
System.out.println("=====");
list.tampil();

System.out.println("Data Mahasiswa Setelah Sorting Desc Berdasarkan IPK");
list.bubbleSort();
list.tampil();
```

## 9. Verifikasi Hasil Percobaan

### 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 : Dampu  
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 : 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

## Pertanyaan Percobaan 1

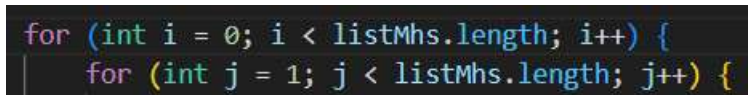
1. Terdapat di method apakah proses bubble sort?



```
1 void bubbleSort() {
2
3     for (int i = 0; i < listMhs.length; i++) {
4         for (int j = 1; j < listMhs.length; j++) {
5             if (listMhs[j].ipk > listMhs[j-1].ipk) {
6
7                 // Proses Swap atau Penukaran
8                 Mahasiswa_18 tmp = listMhs[j];
9                 listMhs[j] = listMhs[j-1];
10                listMhs[j-1] = tmp;
11            }
12        }
13    }
14 }
15
16 }
```

Terdapat pada class DaftarMahasiswaBerprestasi\_18.java

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



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

Untuk apakah proses tersebut?

Proses yang dilakukan dalam loop tersebut adalah iterasi melalui array listMhs untuk melakukan bubble sort dengan tujuan membandingkan setiap pasangan elemen berurutan dalam array listMhs dan menukar posisi mereka jika urutannya tidak benar.

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

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

- Perulangan i digunakan untuk mengontrol elemen. Ini berarti setiap langkah iterasi i mengurangi jumlah elemen yang harus diperiksa pada setiap langkahnya.
- Perulangan j digunakan untuk membandingkan setiap pasangan elemen berurutan dalam array dan **menukar posisi** mereka jika urutannya tidak benar. Ini adalah langkah yang terjadi di dalam satu iterasi dari perulangan i.

b. Mengapa syarat dari perulangan i adalah  $i < \text{listMhs.length} - 1$  ?

karena pada iterasi terakhir, elemen terakhir sudah pasti berada di posisi yang tepat. Oleh karena itu, tidak perlu lagi dilakukan iterasi setelah itu.

c. Mengapa syarat dari perulangan j adalah  $j < \text{listMhs.length} - i$  ?

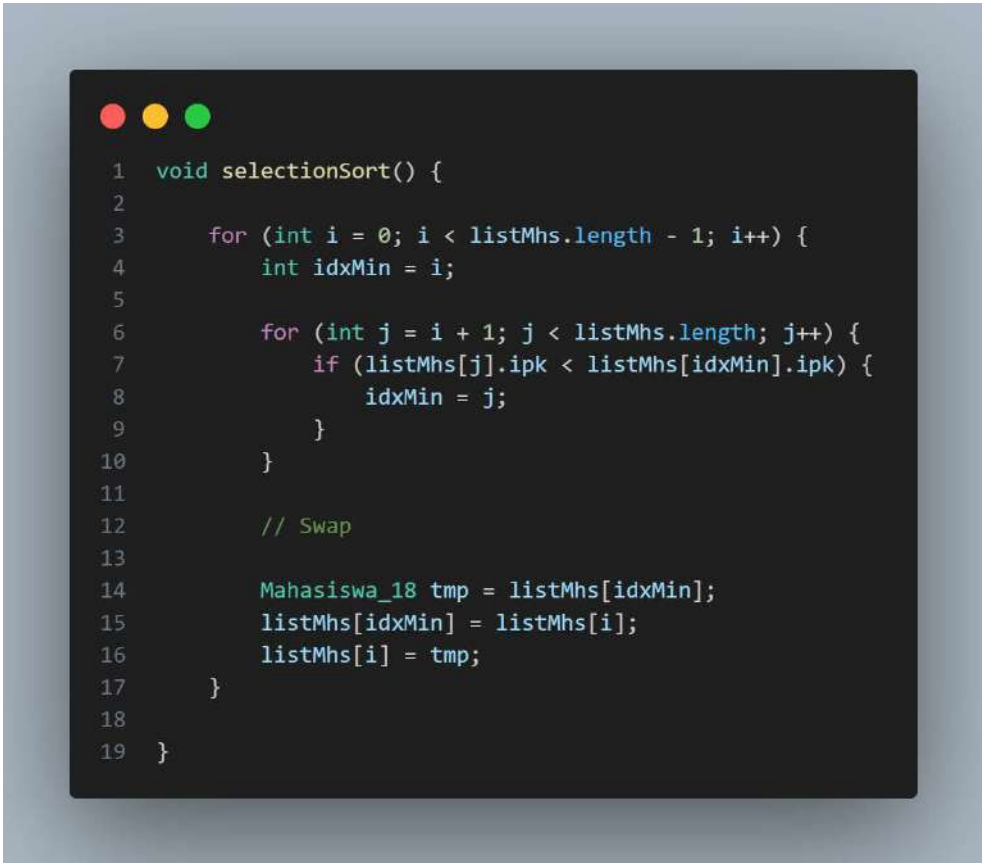
karena pada setiap iterasi i, elemen terakhir sudah pasti berada di posisi yang tepat. Oleh karena itu, tidak perlu membandingkan elemen yang sudah berada di posisi yang tepat.

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

Jika banyak data di dalam listMhs adalah 50, maka perulangan i akan berlangsung sebanyak 49 kali, karena i dimulai dari 0. Tahap bubble sort yang ditempuh juga sebanyak 49 tahap karena satu elemen teratas sudah pasti berada pada posisi yang tepat setelah iterasi sebelumnya.

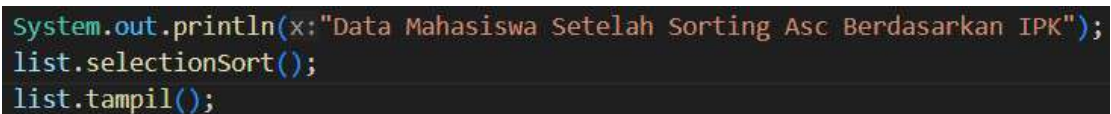
## PERCOBAAN 2 SELECTION SORT

1. Lihat kembali class `DaftarMahasiswaBerprestasi`, dan tambahkan method `selectionSort()` di dalamnya! Method ini juga akan melakukan proses sorting secara ascending, tetapi menggunakan pendekatan selection sort.



```
1 void selectionSort() {
2
3     for (int i = 0; i < listMhs.length - 1; i++) {
4         int idxMin = i;
5
6         for (int j = i + 1; j < listMhs.length; j++) {
7             if (listMhs[j].ipk < listMhs[idxMin].ipk) {
8                 idxMin = j;
9             }
10        }
11
12        // Swap
13
14        Mahasiswa_18 tmp = listMhs[idxMin];
15        listMhs[idxMin] = listMhs[i];
16        listMhs[i] = tmp;
17    }
18
19 }
```

2. 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");
list.selectionSort();
list.tampil();
```

3. Coba jalankan kembali class `Main`, dan amati hasilnya! Apakah kini data mahasiswa telah tampil urut menaik berdasar ipk?



#### 4. Verifikasi Hasil Percobaan

```
=====
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 Asc Berdasarkan IPK
=====
Nama      : Abdul
Tahun Masuk : 2017
Umur      : 23
IPK       : 2.0
=====
Nama      : Nusa
Tahun Masuk : 2017
Umur      : 25
IPK       : 3.0
=====
Nama      : Dompur
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
=====
```

## Pertanyaan Percobaan 2

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!

Proses tersebut digunakan untuk mengurutkan array dari elemen-elemen berdasarkan nilai IPK dari objek Mahasiswa

- **Inisialisasi idxMin** : Variabel idxMin diinisialisasi dengan i, yang merupakan indeks saat ini dari iterasi luar.
- **For** : Di dalam loop (for loop pertama), setiap elemen dari array akan dibandingkan dengan elemen-elemen setelahnya (idxMin) untuk mencari nilai IPK terendah
- **If** : Setiap kali ditemukan elemen dengan nilai IPK yang lebih rendah dari idxMin maka idxMin akan berubah menjadi nilai IPK yang lebih rendah tersebut.
- **Swap** : Elemen dengan nilai IPK terendah akan ditukar posisinya dengan elemen pada indeks saat ini (i). Hal ini dilakukan untuk menggeser elemen dengan nilai IPK terendah ke posisi yang tepat sesuai dengan iterasi saat ini.
- **Iterasi** : Proses ini akan terus berlanjut untuk setiap iterasi hingga seluruh array diurutkan berdasarkan nilai IPK

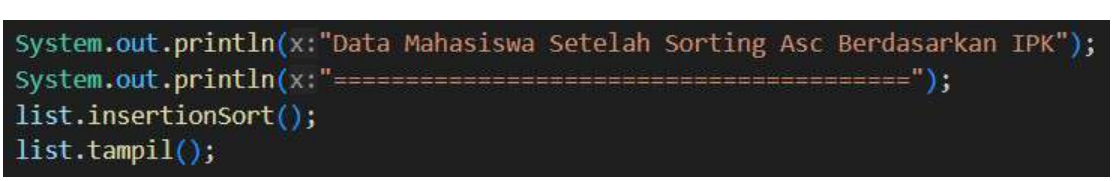
## PERCOBAAN 3 INSERTION SORT

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.



```
1 void insertionSort() {  
2  
3     for (int i = 1; i < listMhs.length; i++) {  
4         Mahasiswa_18 temp = listMhs[i];  
5         int j = i;  
6  
7         while (j > 0 && listMhs[j - 1].ipk > temp.ipk) {  
8             listMhs[j] = listMhs[j - 1];  
9             j--;  
10        }  
11  
12        listMhs[j] = temp;  
13    }  
14 }  
15  
16 }
```

2. Setelah itu, buka kembali class `Main`, dan di dalam method `main()` tambahkan baris program untuk memanggil method `insertionSort()` tersebut!



```
System.out.println(x:"Data Mahasiswa Setelah Sorting Asc Berdasarkan IPK");  
System.out.println(x:"=====");  
list.insertionSort();  
list.tampil();
```

3. Coba jalankan kembali class Main, dan amati hasilnya! Apakah kini data mahasiswa telah tampil urut menaik berdasar ipk?

#### 4. Verifikasi Hasil Percobaan

```
=====
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      : Dompui
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      : Dompui
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
=====
```

### Pertanyaan Percobaan 3

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

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

Perubahan terjadi pada tanda > yang diubah menjadi <

```
=====
Data Mahasiswa Setelah Sorting Asc Berdasarkan IPK
=====
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
=====
```

## Latihan Praktikum

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.

## Hasil Program

```
1  public class Penginapan_18 {
2
3      String nama;
4      int harga;
5      double rating;
6      String alamat;
7
8      Penginapan_18(String nama, int harga, double rating, String alamat) {
9          this.nama = nama;
10         this.harga = harga;
11         this.rating = rating;
12         this.alamat = alamat;
13     }
14
15     void tampil() {
16         System.out.println("Nama Penginapan : " + nama);
17         System.out.println("Harga          : " + harga);
18         System.out.println("Rating         : " + rating);
19         System.out.println("Alamat        : " + alamat);
20     }
21
22 }
23
```

```

1 public class DaftarPenginapan_18 {
2
3     Penginapan_18 listPenginapan[] = new Penginapan_18[5];
4     int idx;
5
6     void tambahPenginapan(Penginapan_18 Penginapan) {
7
8         if (idx < listPenginapan.length) {
9
10             listPenginapan[idx] = Penginapan;
11             idx++;
12
13         } else {
14
15             System.out.println("Data Sudah Penuh");
16
17         }
18     }
19
20
21     void tampil() {
22
23         for (Penginapan_18 Penginapan : listPenginapan) {
24
25             Penginapan.tampil();
26             System.out.println("-----");
27
28         }
29     }
30
31
32     void bubbleSortHargaDesc() {
33
34         for (int i = 0; i < listPenginapan.length; i++) {
35             for (int j = 1; j < listPenginapan.length; j++) {
36                 if (listPenginapan[j].harga > listPenginapan[j-1].harga) {
37
38                     // Proses Swap atau Penukaran
39
40                     Penginapan_18 mahal = listPenginapan[j];
41                     listPenginapan[j] = listPenginapan[j-1];
42                     listPenginapan[j-1] = mahal;
43
44                 }
45             }
46         }
47     }
48
49
50     void bubbleSortHargaAsc() {
51
52         for (int i = 0; i < listPenginapan.length; i++) {
53             for (int j = 1; j < listPenginapan.length; j++) {
54                 if (listPenginapan[j].harga < listPenginapan[j-1].harga) {
55
56                     // Proses Swap atau Penukaran
57
58                     Penginapan_18 mahal = listPenginapan[j];
59                     listPenginapan[j] = listPenginapan[j-1];
60                     listPenginapan[j-1] = mahal;
61
62                 }
63             }
64         }
65     }
66
67
68     void selectionSortRatingAsc() {
69
70         for (int i = 0; i < listPenginapan.length - 1; i++) {
71             int idxMin = i;
72
73             for (int j = i + 1; j < listPenginapan.length; j++) {
74                 if (listPenginapan[j].rating < listPenginapan[idxMin].rating) {
75                     idxMin = j;
76                 }
77             }
78
79             // Swap
80
81             Penginapan_18 terendah = listPenginapan[idxMin];
82             listPenginapan[idxMin] = listPenginapan[i];
83             listPenginapan[i] = terendah;
84
85         }
86     }
87
88
89     void selectionSortRatingDesc() {
90
91         for (int i = 0; i < listPenginapan.length - 1; i++) {
92             int idxMin = i;
93
94             for (int j = i + 1; j < listPenginapan.length; j++) {
95                 if (listPenginapan[j].rating > listPenginapan[idxMin].rating) {
96                     idxMin = j;
97                 }
98             }
99
100             // Swap
101
102             Penginapan_18 tertinggi = listPenginapan[idxMin];
103             listPenginapan[idxMin] = listPenginapan[i];
104             listPenginapan[i] = tertinggi;
105
106         }
107     }
108 }
109

```



```

1  import java.util.Scanner;
2
3  public class MainPenginapan_18 {
4
5      public static void main(String[] args) {
6          Scanner input18 = new Scanner(System.in);
7
8          int pilihan;
9
10         DaftarPenginapan_18 listPenginapan = new DaftarPenginapan_18();
11
12         Penginapan_18 Penginapan1 = new Penginapan_18("DeResort", 585000, 4.2, "Bali");
13         Penginapan_18 Penginapan2 = new Penginapan_18("TeVilla", 665000, 3.8, "Bandung");
14         Penginapan_18 Penginapan3 = new Penginapan_18("Mutiara Hotel", 360000, 3.2, "Semarang");
15         Penginapan_18 Penginapan4 = new Penginapan_18("Bobo Kabin", 405000, 4.7, "Monosobo");
16         Penginapan_18 Penginapan5 = new Penginapan_18("Simple Room", 350000, 4.0, "Bali");
17
18         listPenginapan.tambahPenginapan(Penginapan1);
19         listPenginapan.tambahPenginapan(Penginapan2);
20         listPenginapan.tambahPenginapan(Penginapan3);
21         listPenginapan.tambahPenginapan(Penginapan4);
22         listPenginapan.tambahPenginapan(Penginapan5);
23
24         header();
25         System.out.println("          Daftar Penginapan");
26         header();
27
28         do {
29             System.out.println("1. Tampilkan penginapan sebelum sorting");
30             System.out.println("2. Tampil penginapan berdasarkan harga (Asc)");
31             System.out.println("3. Tampil penginapan berdasarkan harga (Desc)");
32             System.out.println("4. Tampil penginapan berdasarkan rating (Asc)");
33             System.out.println("5. Tampil penginapan berdasarkan rating (Desc)");
34             System.out.println("6. Keluar");
35             header();
36             System.out.print("Masukkan Pilihan   : ");
37             pilihan = input18.nextInt();
38
39             switch (pilihan) {
40                 case 1:
41                     header();
42                     listPenginapan.tampil();
43                     header();
44                     break;
45                 case 2:
46                     header();
47                     System.out.println("  Daftar Penginapan Berdasarkan Harga Termurah");
48                     header();
49                     listPenginapan.bubbleSortHargaAsc();
50                     listPenginapan.tampil();
51                     break;
52                 case 3:
53                     header();
54                     System.out.println("  Daftar Penginapan Berdasarkan Harga Termahal");
55                     header();
56                     listPenginapan.bubbleSortHargaDesc();
57                     listPenginapan.tampil();
58                     break;
59                 case 4:
60                     header();
61                     System.out.println("  Daftar Penginapan Berdasarkan Rating Terendah");
62                     header();
63                     listPenginapan.selectionSortRatingAsc();
64                     listPenginapan.tampil();
65                     break;
66                 case 5:
67                     header();
68                     System.out.println("  Daftar Penginapan Berdasarkan Rating Teratas");
69                     header();
70                     listPenginapan.selectionSortRatingDesc();
71                     listPenginapan.tampil();
72                     break;
73                 case 6:
74                     header();
75                     System.out.println("          Terima Kasih");
76                     header();
77                     break;
78                 default:
79                     System.out.println("          Pilihan tidak valid!");
80                     break;
81             }
82             } while (pilihan != 6);
83
84     }
85
86     static void header() {
87         System.out.println("=====");
88     }
89
90 }
91

```



## Output

```
=====
                        Daftar Penginapan
=====
1. Tampilkan penginapan sebelum sorting
2. Tampilkan penginapan berdasarkan harga (Asc)
3. Tampilkan penginapan berdasarkan harga (Desc)
4. Tampilkan penginapan berdasarkan rating (Asc)
5. Tampilkan penginapan berdasarkan rating (Desc)
6. Keluar
=====

=====
                        Daftar Penginapan Sebelum Sorting
=====
Nama Penginapan : DeResort
Harga           : 585000
Rating          : 4.2
Alamat          : Bali
=====
Nama Penginapan : TeVilla
Harga           : 665000
Rating          : 3.8
Alamat          : Bandung
=====
Nama Penginapan : Mutiara Hotel
Harga           : 360000
Rating          : 3.2
Alamat          : Semarang
=====
Nama Penginapan : Bobo Kabin
Harga           : 405000
Rating          : 4.7
Alamat          : Wonosobo
=====
Nama Penginapan : Simple Room
Harga           : 350000
Rating          : 4.0
Alamat          : Bali
=====
```

=====

Daftar Penginapan Berdasarkan Harga Termurah

=====

Nama Penginapan : Simple Room  
Harga : 350000  
Rating : 4.0  
Alamat : Bali

=====

Nama Penginapan : Mutiara Hotel  
Harga : 360000  
Rating : 3.2  
Alamat : Semarang

=====

Nama Penginapan : Bobo Kabin  
Harga : 405000  
Rating : 4.7  
Alamat : Wonosobo

=====

Nama Penginapan : DeResort  
Harga : 585000  
Rating : 4.2  
Alamat : Bali

=====

Nama Penginapan : TeVilla  
Harga : 665000  
Rating : 3.8  
Alamat : Bandung

=====

=====

Daftar Penginapan Berdasarkan Harga Termahal

=====

Nama Penginapan : TeVilla  
Harga : 665000  
Rating : 3.8  
Alamat : Bandung

=====

Nama Penginapan : DeResort  
Harga : 585000  
Rating : 4.2  
Alamat : Bali

=====

Nama Penginapan : Bobo Kabin  
Harga : 405000  
Rating : 4.7  
Alamat : Wonosobo

=====

Nama Penginapan : Mutiara Hotel  
Harga : 360000  
Rating : 3.2  
Alamat : Semarang

=====

Nama Penginapan : Simple Room  
Harga : 350000  
Rating : 4.0  
Alamat : Bali

=====

=====

Daftar Penginapan Berdasarkan Rating Terendah

=====

Nama Penginapan : Mutiara Hotel  
Harga : 360000  
Rating : 3.2  
Alamat : Semarang

=====

Nama Penginapan : TeVilla  
Harga : 665000  
Rating : 3.8  
Alamat : Bandung

=====

Nama Penginapan : Simple Room  
Harga : 350000  
Rating : 4.0  
Alamat : Bali

=====

Nama Penginapan : DeResort  
Harga : 585000  
Rating : 4.2  
Alamat : Bali

=====

Nama Penginapan : Bobo Kabin  
Harga : 405000  
Rating : 4.7  
Alamat : Wonosobo

=====

=====

Nama Penginapan : Bobo Kabin  
Harga : 405000  
Rating : 4.7  
Alamat : Wonosobo

=====

Nama Penginapan : DeResort  
Harga : 585000  
Rating : 4.2  
Alamat : Bali

=====

Nama Penginapan : Simple Room  
Harga : 350000  
Rating : 4.0  
Alamat : Bali

=====

Nama Penginapan : TeVilla  
Harga : 665000  
Rating : 3.8  
Alamat : Bandung

=====

Nama Penginapan : Mutiara Hotel  
Harga : 360000  
Rating : 3.2  
Alamat : Semarang

=====

Repository GitHub : <https://github.com/zidnafaz/Praktikum-Algoritma-Struktur-Data>