JOBSHEET 16
Collection

Satria Wiguna/Ti 1D/ Absen 26

16.1. Tujuan Praktikum

Setelah melakukan praktikum ini, mahasiswa mampu:

1. memahami bentuk-bentuk collection dan hierarkinya;
2. menerapkan collection sesuai dengan fungsi dan jenisnya;
3. menyelesaikan kasus menggunakan collection yang sesuai.

16.2. Kegiatan Praktikum 1

16.2.1. Percobaan 1

Pada percobaan 1 ini akan dicontohkan penggunaan collection untuk menambahkan sebuah elemen, mengakses elemen, dan menghapus sebuah elemen.

1. Buatlah sebuah class ContohList yang main methode berisi kode program seperti di bawah ini

```
25          List l = new ArrayList();
26          l.add(1);
27          l.add(2);
28          l.add(3);
29          l.add("Cireng");
30          System.out.printf("Elemen 0: %d total elemen: %d elemen terakhir: %s\n",
31                  l.get(0), l.size(), l.get(l.size() - 1));
32
33          l.add(4);
34          l.remove(0);
35          System.out.printf("Elemen 0: %d total elemen: %d elemen terakhir: %s\n",
36                  l.get(0), l.size(), l.get(l.size() - 1));
```

```
1  import java.util.ArrayList;
2  import java.util.LinkedList;
3  import java.util.List;
4
5  public class ContohList {
       Run | Debug
6      public static void main(String[] args) {
7
8          List<Object> l = new ArrayList<>();
9          l.add(e:1);
10         l.add(e:2);
11         l.add(e:3);
12         l.add(e:"Cireng");
13         System.out.printf(format:"Elemen 0: %s total elemen: %d elemen terakhir: %s\n", l.get(index:0), l.size(), l.get(l.size() - 1));
14         l.add(e:4);
15         l.remove(index:0);
16         System.out.printf(format:"Elemen 0: %s total elemen: %d elemen terakhir: %s\n", l.get(index:0), l.size(), l.get(l.size() - 1));
17
```

2. Tambahkan kode program untuk menggunakan collection dengan aturan penulisan kode program seperti berikut

```
38              List<String> names = new LinkedList<>();
39              names.add("Noureen");
40              names.add("Akhleema");
41              names.add("Shannum");
42              names.add("Uwais");
43              names.add("Al-Qarni");
44
45              System.out.printf("Elemen 0: %s total elemen: %s elemen terakhir: %s\n",
46                      names.get(0), names.size(), names.get(names.size() - 1));
47              names.set(0, "My kid");
48              System.out.printf("Elemen 0: %s total elemen: %s elemen terakhir: %s\n",
49                      names.get(0), names.size(), names.get(names.size() - 1));
50              System.out.println("Names: " + names.toString());
```

```
18          List<String> names = new LinkedList<>();
19          names.add(e:"Noureen");
20          names.add(e:"Akhleema");
21          names.add(e:"Shannum");
22          names.add(e:"Uwais");
23          names.add(e:"Al-Qarni");
24          System.out.printf(format:"Elemen 0: %s total elemen: %d elemen terakhir: %s\n",
25          names.get(index:0), names.size(), names.get(names.size() - 1)); names.set(index:0, element:"My kid");
26          System.out.printf(format:"Elemen 0: %s total elemen: %d elemen terakhir: %s\n",
27          names.get(index:0), names.size(), names.get(names.size() - 1)); System.out.println("Names: " + names.toString());
28      }
29  }
30
```

16.2.2. Verifikasi Hasil Percobaan

Verifikasi hasil kompilasi kode program Anda dengan gambar berikut ini.

```
run:
Elemen 0: 1 total elemen: 4 elemen terakhir: Cireng
Elemen 0: 2 total elemen: 4 elemen terakhir: 4
Elemen 0: Noureen total elemen: 5 elemen terakhir: Al-Qarni
Elemen 0: My kid total elemen: 5 elemen terakhir: Al-Qarni
Names: [My kid, Akhleema, Shannum, Uwais, Al-Qarni]
```

```
Elemen 0: 1 total elemen: 4 elemen terakhir: Cireng
Elemen 0: 2 total elemen: 4 elemen terakhir: 4
Elemen 0: Noureen total elemen: 5 elemen terakhir: Al-Qarni
Elemen 0: My kid total elemen: 5 elemen terakhir: Al-Qarni
Names: [My kid, Akhleema, Shannum, Uwais, Al-Qarni]
```

16.2.3. Pertanyaan Percobaan

1. Perhatikan baris kode 25-36, mengapa semua jenis data bisa ditampung ke dalam sebuah Arraylist?

   ArrayList dideklarasikan sebagai List<Object>, yang berarti dapat menampung objek dari tipe data apapun karena Object adalah superclass dari semua tipe data non-primitif di Java

2. Modifikasi baris kode 25-36 seingga data yang ditampung hanya satu jenis atau spesifik tipe tertentu!

```
List<Integer> l = new ArrayList<>();
l.add(e:1);
l.add(e:2);
l.add(e:3);
//l.add("Cireng"); eror karena cireng bukan termasuk integer
System.out.printf(format:"Elemen 0: %s total elemen: %d elemen terakhir: %s\n", l.get(index:0), l.size(), l.get(l.size() - 1));
l.add(e:4);
l.remove(index:0);
System.out.printf(format:"Elemen 0: %s total elemen: %d elemen terakhir: %s\n", l.get(index:0), l.size(), l.get(l.size() - 1));
```

3. Ubah kode pada baris kode 38 menjadi seperti ini

```
LinkedList<String> names = new LinkedList<>();
```

```
LinkedList<String> names = new LinkedList<>();
```

4. Tambahkan juga baris berikut ini, untuk memberikan perbedaan dari tampilan yang sebelumnya

```
names.push("Mei-mei");
System.out.printf("Elemen 0: %s total elemen: %s elemen terakhir: %s\n",
        names.getFirst(), names.size(), names.getLast());
System.out.println("Names: " + names.toString());
```

```
    names.push(e:"Mei-mei");
    System.out.printf(format:"Elemen 0: %s total elemen: %d elemen terakhir: %s\n",
            names.getFirst(), names.size(), names.getLast());

    System.out.println("Names: " + names.toString());
}
```

5. Dari penambahan kode tersebut, silakan dijalankan dan apakah yang dapat Anda jelaskan!
   Kode di atas menunjukkan penggunaan ArrayList untuk menyimpan bilangan bulat dan LinkedList untuk menyimpan nama-nama. Menggunakan List<Integer> memastikan hanya bilangan bulat yang diterima, sementara LinkedList<String> memungkinkan penggunaan metode tambahan seperti push() untuk menambahkan elemen baru di depan. Outputnya mencakup cetakan nilai elemen pertama, jumlah elemen, dan elemen terakhir dari setiap struktur data.

16.3. Kegiatan Praktikum 2

16.3.1. Tahapan Percobaan

Pada praktikum 2 ini akan dibuat beberapa method untuk menampilkan beberapa cara yang dapat dilakukan untuk mengambil/menampilkan elemen pada sebuah collection. Silakan ikutilah Langkah-langkah di bawah ini

1. Buatlah class dengan nama LoopCollection serta tambahkan method main yang isinya adalah sebagai berikut.

```
25          Stack<String> fruits = new Stack<>();
26          fruits.push("Banana");
27          fruits.add("Orange");
28          fruits.add("Watermelon");
29          fruits.add("Leci");
30          fruits.push("Salak");
31
            for (String fruit : fruits) {
33              System.out.printf("%s ", fruit);
34          }
35
36          System.out.println("\n" + fruits.toString());
37
38          while (!fruits.empty()) {
39              System.out.printf("%s ", fruits.pop());
40          }
```

```
public static void main(String[] args) {
    Stack<String> fruits = new Stack<>();
    fruits.push(item:"Banana");
    fruits.add(e:"Orange");
    fruits.add(e:"Watermelon");
    fruits.add(e:"Leci");
    fruits.push(item:"Salak");

    for (String fruit : fruits) {
        System.out.printf(format:"%s ", fruit);
    }
    System.out.println("\n" + fruits.toString());
    while (!fruits.empty()) {
        System.out.printf(format:"%s ", fruits.pop());
```

2. Tambahkan potongan kode berikut ini dari yang sebelumnya agar proses menampilkan elemen pada sebuah stack bervariasi.

```
43              fruits.push("Melon");
44              fruits.push("Durian");
45              System.out.println("");
                for (Iterator<String> it = fruits.iterator(); it.hasNext();) {
47                  String fruit = it.next();
48                  System.out.printf("%s ", fruit);
49              }
50              System.out.println("");
51              fruits.stream().forEach(e -> {
52                  System.out.printf("%s ", e);
53              });
54              System.out.println("");
55              for (int i = 0; i < fruits.size(); i++) {
56                  System.out.printf("%s ", fruits.get(i));
57
58              }
```

```
        fruits.push(item:"Melon");
        fruits.push(item:"Durian");
        System.out.println(x:"");

        for (Iterator<String> it = fruits.iterator(); it.hasNext(); ) {
            String fruit = it.next();
            System.out.printf(format:"%s ", fruit);
        }
        System.out.println(x:"");
        fruits.stream().forEach(e -> {
            System.out.printf(format:"%s ", e);
        });
        System.out.println(x:"");
        for (int i = 0; i < fruits.size(); i++){
            System.out.printf(format:"%s ", fruits.get(i));
        }
    }
}
```

16.3.2. Verifikasi Hasil Percobaan

Verifikasi hasil kompilasi kode program Anda dengan gambar berikut ini.

```
Banana Orange Watermelon Leci Salak
[Banana, Orange, Watermelon, Leci, Salak]
Salak Leci Watermelon Orange Banana
Melon Durian
Melon Durian
Melon Durian BUILD SUCCESSFUL (total time: 0 seconds)
```

```
Banana Orange Watermelon Leci Salak
[Banana, Orange, Watermelon, Leci, Salak]
Salak Leci Watermelon Orange Banana
Melon Durian
Melon Durian
Melon Durian
```

16.3.3. Pertanyaan Percobaan

1. Apakah perbedaan fungsi push() dan add() pada objek *fruits*?

   push() adalah metode dari kelas Stack yang digunakan untuk menambahkan elemen ke atas stack.

   add() adalah metode dari antarmuka Collection (yang diimplementasikan oleh Stack) yang digunakan untuk menambahkan elemen ke dalam koleksi

2. Silakan hilangkan baris 43 dan 44, apakah yang akan terjadi? Mengapa bisa demikian?
   Jika baris 43 (System.out.println("")) dan baris 44 (for (Iterator<String> it = fruits.iterator(); it.hasNext();) {) dihapus, maka output program tidak akan mencetak elemen-elemen dari fruits setelah di-push kembali dengan "Melon" dan "Durian". Ini terjadi karena bagian kode yang mengiterasi dan mencetak elemen setelah push akan hilang.

3. Jelaskan fungsi dari baris 46-49?
   - Baris 46-49 menggunakan iterasi dengan Iterator untuk mencetak setiap elemen yang tersisa dalam fruits setelah elemen-elemen sebelumnya di-pop.
   - fruits.iterator() mengambil iterator dari stack fruits.
   - it.hasNext() memeriksa apakah masih ada elemen yang tersisa dalam stack.
   - it.next() mengambil dan mencetak elemen saat ini dari iterator.
4. Silakan ganti baris kode 25, *Stack<String>* menjadi *List<String>* dan apakah yang terjadi? Mengapa bisa demikian?
   atidak akan bisa langsung menggunakan metode push() karena List tidak memiliki metode push(). Sebagai gantinya, Anda akan menggunakan metode add() untuk menambahkan elemen ke dalam List

5. Ganti elemen terakhir dari dari objek fruits menjadi “Strawberry”!

```
fruits.set(fruits.size() - 1, element:"Strawberry");
```

```
Banana Orange Watermelon Leci Salak
[Banana, Orange, Watermelon, Leci, Salak]
Salak Leci Watermelon Orange Banana
Melon Strawberry
Melon Strawberry
Melon Strawberry
```

6. Tambahkan 3 buah seperti “Mango”,”guava”, dan “avocado” kemudian dilakukan sorting!

```
fruits.push(item:"Melon");
fruits.push(item:"Durian");
// fruits.set(fruits.size() - 1, "Strawberry");
fruits.add(e:"Mango");
fruits.add(e:"Guava");
fruits.add(e:"Avocado");
Collections.sort(fruits);
System.out.println(x:"");
```

```
Banana Orange Watermelon Leci Salak
[Banana, Orange, Watermelon, Leci, Salak]
Salak Leci Watermelon Orange Banana
Avocado Durian Guava Mango Melon
Avocado Durian Guava Mango Melon
Avocado Durian Guava Mango Melon
```

16.4. Kegiatan Praktikum 3

16.4.1. Tahapan Percobaan

Pada praktikum 3 ini dilakukan uji coba untuk mengimplementasikan sebuah collection untuk menampung objek yang dibuat sesuai kebutuhan. Objek tersebut adalah sebuah objek mahasiswa dengan fungsi-fungsi umum seperti menambahkan, menghapus, mengubah, dan mencari.

1. Buatlah sebuah class Mahasiswa dengan attribute, kontruktor, dan fungsi sebagai berikut.

```
String nim;
String nama;
String notelp;

public Mahasiswa() {
}

public Mahasiswa(String nim, String nama, String notelp) {
    this.nim = nim;
    this.nama = nama;
    this.notelp = notelp;
}

@Override
public String toString() {
    return "Mahasiswa{" + "nim=" + nim + ", nama=" + nama + ", notelp=" + notelp + '}';
}
```

```
package Praktikum3;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

public class Mahasiswa {
    String nim,nama,notelp;

    Mahasiswa(){

    }
    Mahasiswa(String nim, String nama, String notelp){
        this.nim = nim;
        this.nama = nama;
        this.notelp = notelp;
    }
    @Override
    public String toString() {
        return "Mahasiswa{" + "nim=" + nim + ", nama=" + nama + ", notelp=" + notelp + '}';
    }
    public String getNim() {
        return nim;
    }
}
```

2. Selanjutnya, buatlah sebuah class ListMahasiswa yang memiliki attribute seperti di bawah ini

```
List<Mahasiswa> mahasiswas = new ArrayList<>();
```

3. Method **tambah(), hapus(), update(), dan tampil()** secara berurut dibuat agar bisa melakukan operasi-operasi seperti yang telah disebutkan.

```
public void tambah(Mahasiswa... mahasiswa) {
    mahasiswas.addAll(Arrays.asList(mahasiswa));
}

public void hapus(int index) {
    mahasiswas.remove(index);
}

public void update(int index, Mahasiswa mhs) {
    mahasiswas.set(index, mhs);
}

public void tampil() {
    mahasiswas.stream().forEach(mhs -> {
        System.out.println("" + mhs.toString());
    });
}
```

```
public class ListMahasiswa {
    List<Mahasiswa> mahasiswas = new ArrayList<>();

    void tambah(Mahasiswa... mahasiswa) {
        mahasiswas.addAll(Arrays.asList(mahasiswa));
    }

    void hapus(int index) {
        mahasiswas.remove(index);
    }

    void update(int index, Mahasiswa mhs) {
        mahasiswas.set(index, mhs);
    }

    void tampil() {
        mahasiswas.forEach(mhs -> {
            System.out.println(mhs.toString());
        });
    }
```

4. Untuk proses hapus, update membutuhkan fungsi pencarian terlebih dahulu yang potongan kode programnya adalah sebagai berikut

```
int linearSearch(String nim) {
    for (int i = 0; i < mahasiswas.size(); i++) {
        if (nim.equals(mahasiswas.get(i).nim)) {
            return i;
        }
    }
    return -1;
}
```

```
int linearSearch(String nim){
    for(int i = 0; i<mahasiswas.size(); i++){
        if(nim.equals(mahasiswas.get(i).nim)){
            return i;
        }
    }
    return -1;
}
```

5. Pada class yang sama, tambahkan main method seperti potongan program berikut dan amati hasilnya!

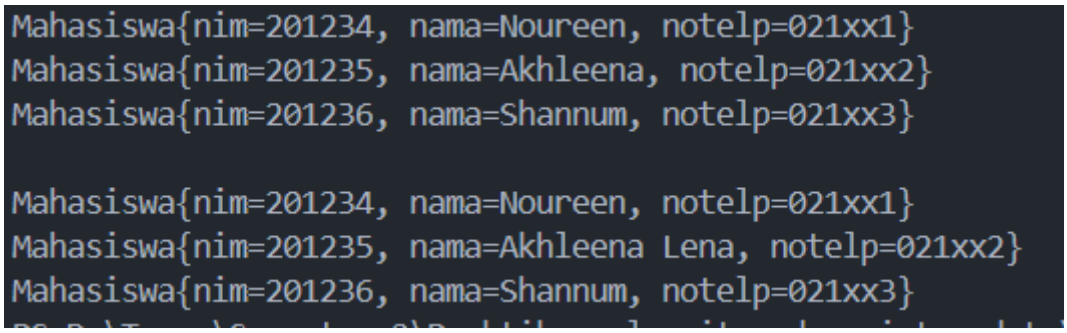
```
ListMahasiswa lm = new ListMahasiswa();
Mahasiswa m = new Mahasiswa("201234", "Noureen", "021xx1");
Mahasiswa m1 = new Mahasiswa("201235", "Akhleema", "021xx2");
Mahasiswa m2 = new Mahasiswa("201236", "Shannum", "021xx3");
  menambahkan objek mahasiswa
lm.tambah(m, m1, m2);
  menampilkan list mahasiswa
lm.tampil();
  update mahasiswa
lm.update(lm.linearSearch("201235"), new Mahasiswa("201235", "Akhleema Lela", "021xx2"));
System.out.println("");
lm.tampil();
```

```
public static void main(String[] args) {
    ListMahasiswa lm = new ListMahasiswa();
    Mahasiswa m1 = new Mahasiswa(nim:"201234", nama:"Noureen", notelp:"021xx1");
    Mahasiswa m2 = new Mahasiswa(nim:"201235", nama:"Akhleena", notelp:"021xx2");
    Mahasiswa m3 = new Mahasiswa(nim:"201236", nama:"Shannum", notelp:"021xx3");

    lm.tambah(m1, m2, m3);
    lm.tampil();

    int index = lm.binarySearch(nim:"201235");
    if (index != -1) {
        // Perbarui Mahasiswa dengan NIM yang ditemukan
        lm.update(index, new Mahasiswa(nim:"201235", nama:"Akhleena Lena", notelp:"021xx2"));
    } else {
        System.out.println(x:"Mahasiswa with NIM 201235 not found.");
    }

    System.out.println();
    lm.tampil();
  }
}
```

16.4.2. Verifikasi Hasil Percobaan

Verifikasi hasil kompilasi kode program Anda dengan gambar berikut ini.

```
Mahasiswa{nim=201234, nama=Noureen, notelp=021xx1}
Mahasiswa{nim=201235, nama=Akhleema, notelp=021xx2}
Mahasiswa{nim=201236, nama=Shannum, notelp=021xx3}

Mahasiswa{nim=201234, nama=Noureen, notelp=021xx1}
Mahasiswa{nim=201235, nama=Akhleema Lela, notelp=021xx2}
Mahasiswa{nim=201236, nama=Shannum, notelp=021xx3}
BUILD SUCCESSFUL (total time: 0 seconds)
```



16.4.3. Pertanyaan Percobaan

1. Pada fungsi tambah() yang menggunakan unlimited argument itu menggunakan konsep apa? Dan kelebihannya apa?

   Konsep pada tambah() dengan unlimited argument: Menggunakan konsep varargs. Kelebihannya adalah memberikan fleksibilitas dalam menerima jumlah argumen yang bervariasi saat memanggil metode, tanpa perlu mendefinisikan array secara eksplisit.

2. Pada fungsi linearSearch() di atas, silakan diganti dengan fungsi binarySearch() dari collection!

```
int binarySearch(String nim) {
    sortByNim(ascending:true);
    Mahasiswa key = new Mahasiswa(nim, nama:"", notelp:""); // Buat objek Mahasiswa dengan NIM sebagai kunci pencarian
    return Collections.binarySearch(mahasiswas, key, Comparator.comparing(Mahasiswa::getNim));
}
```

3. Tambahkan fungsi sorting baik secara ascending ataupun descending pada class tersebut!

```
void sortByNim(boolean ascending) {
    if (ascending) {
        Collections.sort(mahasiswas, Comparator.comparing(Mahasiswa::getNim));
    } else {
        Collections.sort(mahasiswas, Comparator.comparing(Mahasiswa::getNim).reversed());
    }
}
```

16.5. Tugas Praktikum

1. Buatlah implementasi program daftar nilai mahasiswa semester, minimal memiliki 3 class yaitu Mahasiswa, Nilai, dan Mata Kuliah. Data Mahasiswa dan Mata Kuliah perlu melalui penginputan data terlebih dahulu.

   Ilustrasi Program

   Menu Awal dan Penambahan Data

```
**********************************************
SISTEM PENGOLAHAN DATA NILAI MAHASISWA SEMESTER
**********************************************

1. Input Nilai
2. Tampil Nilai
3. Mencari Nilai Mahasiswa
4. Urut Data Nilai
5. Keluar
**********************************************
Pilih     :
```

```
Pilih     : 1
Masukan data
Kode      : 0001
Nilai     : 80.75

DAFTAR MAHASISWA
*********************************************
NIM            Nama           Telf
20001          Thalhah        021xxx
20002          Zubair         021xxx
20003          Abdur-Rahman   021xxx
20004          Sa'ad          021xxx
20005          Sa'id          021xxx
20006          Ubaidah        021xxx
Pilih mahasiswa by nim: 20001
DAFTAR MATA KULIAH
*********************************************
Kode      Mata Kuliah                          SKS
00001     Internet of Things                   3
00002     Algoritma dan Struktur Data          2
00003     Algoritma dan Pemrograman            2
00004     Praktikum Algoritma dan Struktur Data   3
00005     Praktikum Algoritma dan Pemrograman     3
Pilih MK by kode: 00001
```

Tampil Nilai

```
*********************************************
SISTEM PENGOLAHAN DATA NILAI MAHASISWA SEMESTER
*********************************************

1. Input Nilai
2. Tampil Nilai
3. Mencari Nilai Mahasiswa
4. Urut Data Nilai
5. Keluar
*********************************************
Pilih     : 2

DAFTAR NILAI MAHASISWA
*********************************************
Nim            Nama           Mata Kuliah                          SKS       Nilai
20001          Thalhah        Internet of Things                   3         80.75
```

Pencarian Data Mahasiswa

```
*********************************************
SISTEM PENGOLAHAN DATA NILAI MAHASISWA SEMESTER
*********************************************

1. Input Nilai
2. Tampil Nilai
3. Mencari Nilai Mahasiswa
4. Urut Data Nilai
5. Keluar
*********************************************
Pilih     : 3

DAFTAR NILAI MAHASISWA
*********************************************
Nim            Nama           Mata Kuliah                          SKS       Nilai
20001          Thalhah        Internet of Things                   3         90.00
20002          Zubair         Praktikum Algoritma dan Pemrograman  3         80.75
Masukkan data mahasiswa[nim] :20002
Nim            Nama           Mata Kuliah                          SKS       Nilai
20002          Zubair         Praktikum Algoritma dan Pemrograman  3         80.75
Total SKS 3 telah diambil.
```

Pengurutan Data Nilai

```
************************************************
SISTEM PENGOLAHAN DATA NILAI MAHASISWA SEMESTER
************************************************

1. Input Nilai
2. Tampil Nilai
3. Mencari Nilai Mahasiswa
4. Urut Data Nilai
5. Keluar
************************************************
Pilih     : 4

DAFTAR NILAI MAHASISWA
************************************************
Nim             Nama            Mata Kuliah                             SKS         Nilai
20002           Zubair          Praktikum Algoritma dan Pemrograman     3           80.75
20001           Thalhah         Internet of Things                      3           90.00
```

2. Tambahkan prosedur hapus data mahasiswa melalui implementasi Queue pada collections Tugas nomor 1!

Nilai:

```java
package TugasPraktikum;

public class Nilai {
    private Mahasiswa mahasiswa;
    private MataKuliah mataKuliah;
    private double nilai;

    public Nilai(Mahasiswa mahasiswa, MataKuliah mataKuliah, double nilai) {
        this.mahasiswa = mahasiswa;
        this.mataKuliah = mataKuliah;
        this.nilai = nilai;
    }

    public Mahasiswa getMahasiswa() {
        return mahasiswa;
    }

    public MataKuliah getMataKuliah() {
        return mataKuliah;
    }

    public double getNilai() {
        return nilai;
    }

    @Override
    public String toString() {
        return "Mahasiswa: " + mahasiswa.getNama() + ", Mata Kuliah: " + mataKuliah.getNama() + ", Nilai: " + nilai;
    }
}
```

Mata kuliah:

```java
package TugasPraktikum;

public class MataKuliah {
    private String kode;
    private String nama;
    private int sks;

    public MataKuliah(String kode, String nama, int sks) {
        this.kode = kode;
        this.nama = nama;
        this.sks = sks;
    }

    public String getKode() {
        return kode;
    }

    public String getNama() {
        return nama;
    }

    public int getSks() {
        return sks;
    }

    @Override
    public String toString() {
        return kode + "\t" + nama + "\t" + sks;
    }
}
```

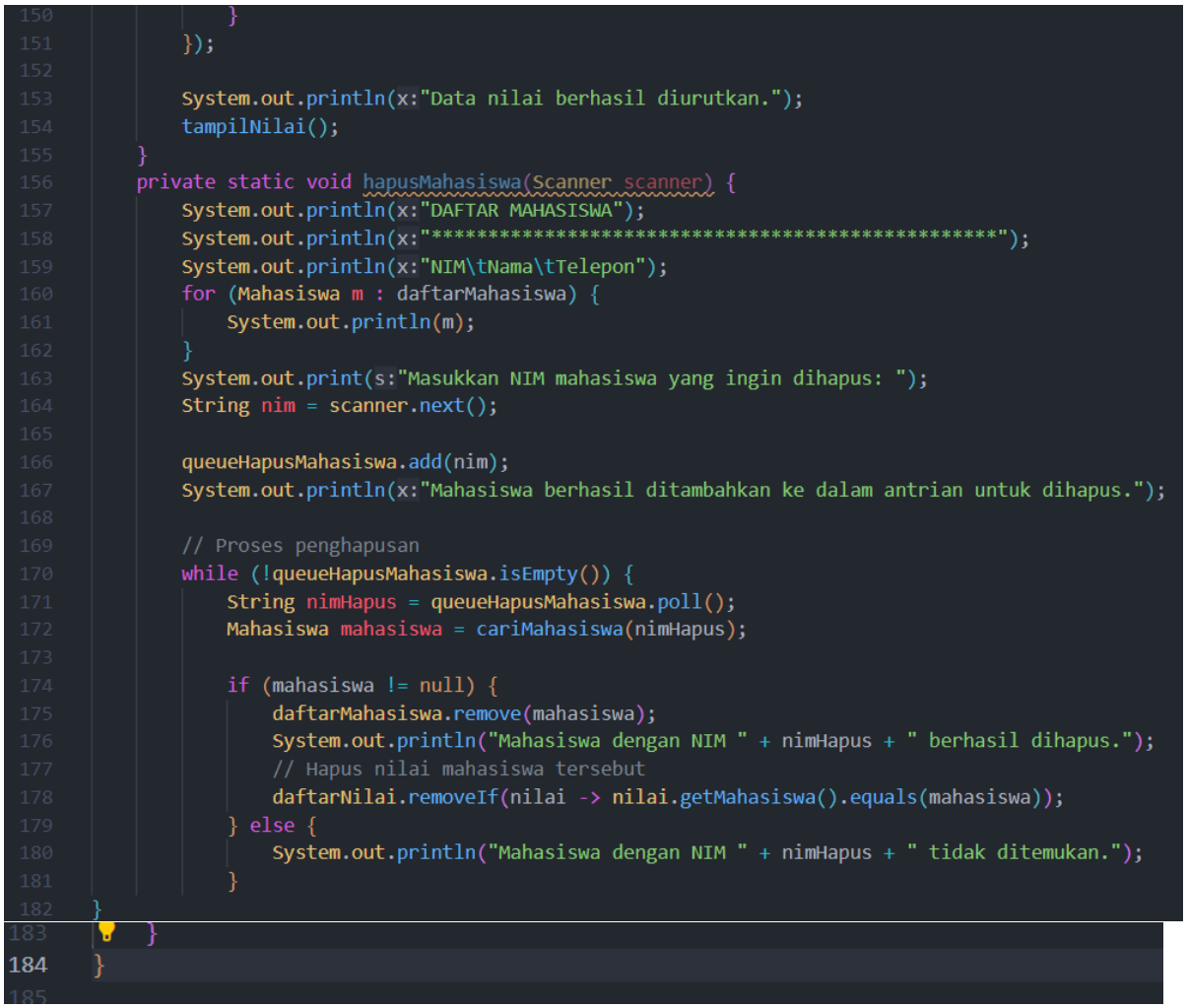
Mahasiswa:

```java
package TugasPraktikum;

public class Mahasiswa {
    private String nim;
    private String nama;
    private String telepon;

    public Mahasiswa(String nim, String nama, String telepon) {
        this.nim = nim;
        this.nama = nama;
        this.telepon = telepon;
    }

    public String getNim() {
        return nim;
    }

    public String getNama() {
        return nama;
    }

    public String getTelepon() {
        return telepon;
    }

    @Override
    public String toString() {
        return nim + "\t" + nama + "\t" + telepon;
    }
}
```

Main:

```java
package TugasPraktikum;
import java.util.*;

public class Main {
    private static List<Mahasiswa> daftarMahasiswa = new ArrayList<>();
    private static List<MataKuliah> daftarMataKuliah = new ArrayList<>();
    private static List<Nilai> daftarNilai = new ArrayList<>();
    private static Queue<String> queueHapusMahasiswa = new LinkedList<>();
    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Data Mahasiswa
        daftarMahasiswa.add(new Mahasiswa(nim:"20001", nama:"Thalhah", telepon:"021xxx"));
        daftarMahasiswa.add(new Mahasiswa(nim:"20002", nama:"Zubair", telepon:"021xxx"));
        daftarMahasiswa.add(new Mahasiswa(nim:"20003", nama:"Abdur-Rahman", telepon:"021xxx"));
        daftarMahasiswa.add(new Mahasiswa(nim:"20004", nama:"Sa'ad", telepon:"021xxx"));
        daftarMahasiswa.add(new Mahasiswa(nim:"20005", nama:"Sa'id", telepon:"021xxx"));
        daftarMahasiswa.add(new Mahasiswa(nim:"20006", nama:"Ubaidah", telepon:"021xxx"));

        // Data Mata Kuliah
        daftarMataKuliah.add(new MataKuliah(kode:"0001", nama:"Internet of Things", sks:3));
        daftarMataKuliah.add(new MataKuliah(kode:"0002", nama:"Algoritma dan Struktur Data", sks:2));
        daftarMataKuliah.add(new MataKuliah(kode:"0003", nama:"Algoritma dan Pemrograman", sks:2));
        daftarMataKuliah.add(new MataKuliah(kode:"0004", nama:"Praktikum Algoritma dan Struktur Data", sks:3));
        daftarMataKuliah.add(new MataKuliah(kode:"0005", nama:"Praktikum Algoritma dan Pemrograman", sks:3));

        int pilihan;

        do {
            System.out.println(x:"Menu:");
            System.out.println(x:"1. Input Nilai");
            System.out.println(x:"2. Tampil Nilai");
            System.out.println(x:"3. Mencari Nilai Mahasiswa");
            System.out.println(x:"4. Urut Data Nilai");
            System.out.println(x:"5. Keluar");
            System.out.print(s:"Pilih menu: ");
            pilihan = scanner.nextInt();

            switch (pilihan) {
                case 1:
                    inputNilai(scanner);
                    break;
                case 2:
                    tampilNilai();
                    break;
                case 3:
                    cariNilaiMahasiswa(scanner);
                    break;
                case 4:
                    urutDataNilai();
                    break;
                case 5:
                    System.out.println(x:"Keluar program.");
                    break;
                default:
                    System.out.println(x:"Pilihan tidak valid.");
            }
        } while (pilihan != 5);

        scanner.close();
    }

    private static void inputNilai(Scanner scanner) {
        System.out.println(x:"DAFTAR MAHASISWA");
        System.out.println(x:"************************************************");
        System.out.println(x:"NIM\tNama\tTelepon");
        for (Mahasiswa m : daftarMahasiswa) {
            System.out.println(m);
        }
        System.out.print(s:"Pilih mahasiswa by nim: ");
        String nim = scanner.next();
        Mahasiswa mahasiswa = cariMahasiswa(nim);

        if (mahasiswa == null) {
            System.out.println(x:"Mahasiswa tidak ditemukan.");
```

```
            return;
        }

        System.out.println(x:"DAFTAR MATA KULIAH");
        System.out.println(x:"*********************************************");
        System.out.println(x:"Kode\tMata Kuliah\tSKS");
        for (MataKuliah mk : daftarMataKuliah) {
            System.out.println(mk);
        }
        System.out.print(s:"Pilih MK by kode: ");
        String kode = scanner.next();
        MataKuliah mataKuliah = cariMataKuliah(kode);

        if (mataKuliah == null) {
            System.out.println(x:"Mata Kuliah tidak ditemukan.");
            return;
        }

        System.out.print(s:"Masukkan Nilai: ");
        double nilai = scanner.nextDouble();

        Nilai nilaiMahasiswa = new Nilai(mahasiswa, mataKuliah, (int) nilai);
        daftarNilai.add(nilaiMahasiswa);

        System.out.println(x:"Nilai berhasil ditambahkan.");
    }

    private static Mahasiswa cariMahasiswa(String nim) {
        for (Mahasiswa m : daftarMahasiswa) {
            if (m.getNim().equals(nim)) {
                return m;
            }
        }
        return null;
    }

    private static MataKuliah cariMataKuliah(String kode) {
        for (MataKuliah mk : daftarMataKuliah) {
            if (mk.getKode().equals(kode)) {
                return mk;
            }
        }
        return null;
    }

    private static void tampilNilai() {
        System.out.println(x:"DAFTAR NILAI");
        System.out.println(x:"************************************************");
        for (Nilai n : daftarNilai) {
            System.out.println(n);
        }
    }

    private static void cariNilaiMahasiswa(Scanner scanner) {
        System.out.print(s:"Masukkan NIM Mahasiswa: ");
        String nim = scanner.next();
        Mahasiswa mahasiswa = cariMahasiswa(nim);

        if (mahasiswa != null) {
            for (Nilai n : daftarNilai) {
                if (n.getMahasiswa().equals(mahasiswa)) {
                    System.out.println(n);
                }
            }
        } else {
            System.out.println(x:"Mahasiswa tidak ditemukan.");
        }
    }

    private static void urutDataNilai() {
        Collections.sort(daftarNilai, new Comparator<Nilai>() {
            @Override
            public int compare(Nilai n1, Nilai n2) {
                return Double.compare(n1.getNilai(), n2.getNilai());
```



```
150                 }
151             });
152
153             System.out.println(x:"Data nilai berhasil diurutkan.");
154             tampilNilai();
155         }
156         private static void hapusMahasiswa(Scanner scanner) {
157             System.out.println(x:"DAFTAR MAHASISWA");
158             System.out.println(x:"*********************************************");
159             System.out.println(x:"NIM\tNama\tTelepon");
160             for (Mahasiswa m : daftarMahasiswa) {
161                 System.out.println(m);
162             }
163             System.out.print(s:"Masukkan NIM mahasiswa yang ingin dihapus: ");
164             String nim = scanner.next();
165
166             queueHapusMahasiswa.add(nim);
167             System.out.println(x:"Mahasiswa berhasil ditambahkan ke dalam antrian untuk dihapus.");
168
169             // Proses penghapusan
170             while (!queueHapusMahasiswa.isEmpty()) {
171                 String nimHapus = queueHapusMahasiswa.poll();
172                 Mahasiswa mahasiswa = cariMahasiswa(nimHapus);
173
174                 if (mahasiswa != null) {
175                     daftarMahasiswa.remove(mahasiswa);
176                     System.out.println("Mahasiswa dengan NIM " + nimHapus + " berhasil dihapus.");
177                     // Hapus nilai mahasiswa tersebut
178                     daftarNilai.removeIf(nilai -> nilai.getMahasiswa().equals(mahasiswa));
179                 } else {
180                     System.out.println("Mahasiswa dengan NIM " + nimHapus + " tidak ditemukan.");
181                 }
182     }
183     }
184 }
185
```

--- *** ---