

LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR DATA JOBSHEET 15



**OLEH :
BOBY ROZAK SAPUTRA
2341760162
SIB-1F/05
D-IV SISTEM INFORMASI BISNIS
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG**

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

1. Buatlah sebuah class ContohList dan main method

```
4 public class ContohList {
5     public static void main(String[] args) {
6
7         List l = new ArrayList<>();
8         l.add(e:1);
9         l.add(e:2);
10        l.add(e:3);
11        l.add(e:"Cireng");
12        System.out.printf(format:"Elemen 0: %d total elemen: %d elemen terakhir: %s\n", l.get(index:0), l.size(),
13                            l.get(l.size() - 1));
14
15        l.add(e:4);
16        l.remove(index:0);
17        System.out.printf(format:"Elemen 0: %d total elemen: %d elemen terakhir: %s\n",l.get(index:0), l.size(),
18                            l.get(l.size() - 1));
19    }
20 }
```

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

```
list<String> names = new LinkedList<>();
names.add("Noureen");
names.add("Akh leena");
names.add("Shannun");
names.add("Uwais");
names.add("A1-0arni");

System.out.printf(format:"Elemen 0: %s total elenen: %s elenen terakhir: %s\n", names.get(0), names.size(),
                    names.get(names.size() - 1));

names.set(0, "My kid");

System.out.printf(format:"Elemen 0: %s total elemen: %s elemen terakhir: %s\n", names.get(0), names.size(),
                    names.get(names.size() - 1));

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

16.2.2. Verifikasi Hasil Percobaan

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

```
Elemen 0: 1 total elemen: 4 elemen terakhir: Cireng
Elemen 0: 2 total elemen: 4 elemen terakhir: 4
Elemen 0: Noreen total elemen: 5 elemen terakhir: A1-Qarni
Elemen 0: My kid total elemen: 5 elemen terakhir: A1-Qarni
Names: [My kid, Akh leena, Shannun, Uwais, A1-Qarni]
```

16.2.3. Pertanyaan Percobaan

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

Karena tidak ada definisi type dari arraylist pada saat insansiasi

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

```
List<Integer> l = new ArrayList<>();
```

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:"Hei-hei");
System.out.printf(format:"Elemen 0: %s total elemen: %s 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!

Dengan menggunakan linkedlist kita dapat melakukan fungsi push, get first dan get last

16.3. Kegiatan Praktikum 2

16.3.1. Tahapan Percobaan

1. Buatlah class dengan nama LoopCollection serta tambahkan method main

```
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.

```
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

```
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
PS C:\Users\bobyr\OneDrive\Documents\Semester Genap\Algoritma dan Struktur Data\Jobsheet15>
```

16.3.3. Pertanyaan Percobaan

1. Apakah perbedaan fungsi `push()` dan `add()` pada objek *fruits*?
`add()` menambahkan elemen baru ke ujung (rear) dari objek Stack
`push()`, sama dengan `add()` namun menggunakan konsep lifo
2. Silakan hilangkan baris 43 dan 44, apakah yang akan terjadi? Mengapa bisa demikian?
Ketika kode program dijalankan tidak mengeluarkan output "Melon Durian, karena pada perulangan akhir method `get` tidak menjalankan statement apapun
3. Jelaskan fungsi dari baris 46-49?
Menampilkan seluruh elemen pada stack
4. Silakan ganti baris kode 25, `Stack<String>` menjadi `List<String>` dan apakah yang terjadi?
Mengapa bisa demikian?
Errorr dikarenakan tidak terdapat "`import java.util.list`"
5. Ganti elemen terakhir dari dari objek *fruits* menjadi "Strawberry"!

```
fruits.pop(); // Remove the last element
fruits.push(item:"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!

```
// Add new fruits (Mango, Guava, Avocado)
fruits.addAll(java.util.Arrays.asList(...a:"Mango", "Guava", "Avocado"));

// Sort the fruits (ascending order)
fruits.sort(java.util.Comparator.naturalOrder());
```

16.4. Kegiatan Praktikum 3

16.4.1. Tahapan Percobaan

1. Buat sebuah class Mahasiswa dengan attribute, kontruktur, dan fungsi

```
public class Mahasiswa {  
    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 + "}";  
    }  
}
```

2. Selanjutnya, buat sebuah class ListMahasiswa yang memiliki attribute

```
public class ListMahasiswa {  
    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());  
    });  
}
```

4. Untuk proses hapus, update membutuhkan fungsi pencarian terlebih dahulu

```

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

```

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

```

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

    // menambahkan objek mahasiswa
    lm.tambah(m, m1, m2);

    // enanpikan list mahasiswa
    lm.tampil();

    // Update mahasiswa
    lm.update(lm.linearSearch(nim:"201235"), new Mahasiswa(nim:"201235", nama:"Akhleena Lela", notelp:"021xx2"));
    System.out.println(x:"");
}

```

16.4.2. Verifikasi Hasil Percobaan

```

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

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

```

```

C:\Din> java ListMahasiswa
Mahasiswa{nim=201234, nama=Noureen, notelp=021xx1}
Mahasiswa{nim=201235, nama=Akhleena, notelp=021xx2}
Mahasiswa{nim=201236, nama=Shannun, notelp=021xx3}

Mahasiswa{nim=201234, nama=Noureen, notelp=021xx1}
Mahasiswa{nim=201235, nama=Akhleena Lela, notelp=021xx2}
Mahasiswa{nim=201236, nama=Shannun, notelp=021xx3}

```

16.4.3. Pertanyaan Percobaan

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

Menggunakan konsep Variable Arguments yang memungkinkan menerima jumlah argumen yang tidak terdefinisi. Kelebihan konsep ini memungkinkan fungsi menerima jumlah argumen yang tidak terdefinisi, dan meningkatkan fleksibilitas kode

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

```
int binarySearch(String nim) {
    int low = 0;
    int high = mahasiswa.size() - 1;

    while (low <= high) {
        int mid = low + (high - low) / 2;
        Mahasiswa mhs = mahasiswa.get(mid);

        if (mhs.nim.equals(nim)) {
            return mid;
        } else if (mhs.nim.compareTo(nim) < 0) {
            low = mid + 1;
        } else {
            high = mid - 1;
        }
    }

    return -1;
}
```

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

```
public void sortByNimAsc() {
    Collections.sort(mahasiswa, (mhs1, mhs2) -> mhs1.nim.compareTo(mhs2.nim));
}

public void sortByNimDesc() {
    Collections.sort(mahasiswa, (mhs1, mhs2) -> mhs2.nim.compareTo(mhs1.nim));
}
```

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.

▼ Tugas	
✓ Mahasiswa05.j...	U
✓ Main05.java	1, U
✓ MataKuliah05.j...	U
✓ Nilai05.java	U
✓ Sistem05.java	U

```
PS C:\Users\boby\OneDrive\Documents\Semester Gen...
'-XX:+ShowCodeDetailsInExceptionMessages' '-cp' '
a3\redhat.java\jdt_ws\Jobsheet15_fba9c8fc\bin' 'Tu
=====
SISTEM PENGOLAHAN DATA NILAI MAHASISWA SEMESTER
=====
1. Input Nilai
2. Tampil Nilai
3. Mencari Nilai Mahasiswa
4. Urut Data Nilai
5. Keluar
Pilih:
```

```
=====
NILAI MAHASISWA NIM: 001
=====
Nama      : BOBY
Mata Kuliah : Internet of Things
SKS       : 3
Nilai     : 90.0
=====
```

```
=====
SISTEM PENGOLAHAN DATA NILAI MAHASISWA SEMESTER
=====
1. Input Nilai
2. Tampil Nilai
3. Mencari Nilai Mahasiswa
4. Urut Data Nilai
5. Keluar
Pilih: 1
Masukkan NIM: 002
Masukkan Kode Mata Kuliah: 02
Masukkan Nilai: 80
```

```
Pilih: 4
=====
DAFTAR NILAI MAHASISWA
=====
NIM       : 001
Nama      : BOBY
Mata Kuliah : Internet of Things
SKS       : 3
Nilai     : 90.0
=====
```

```
Pilih: 2
=====
DAFTAR NILAI MAHASISWA
=====
NIM       : 001
Nama      : BOBY
Mata Kuliah : Internet of Things
SKS       : 3
Nilai     : 90.0
=====
```

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!

```
void antrianHapusMahasiswa(String nim) {  
    try {  
        Mahasiswa05 mahasiswa = daftarMahasiswa.stream()  
            .filter(m -> m.nim.equals(nim)).findFirst().orElse(null);  
        if (mahasiswa != null) {  
            antrianHapus.add(mahasiswa);  
            System.out.println("Mahasiswa dengan NIM " + nim + " ditambahkan ke dalam antrian penghapusan.");  
        } else {  
            System.out.println("Mahasiswa dengan NIM " + nim + " tidak ditemukan.");  
        }  
    } catch (NullPointerException e) {  
        System.out.println(x:"Terjadi kesalahan saat mencari mahasiswa. Silakan periksa input NIM.");  
    }  
}  
  
void hapusMahasiswa() {  
    Mahasiswa05 mahasiswa = antrianHapus.poll();  
    if (mahasiswa != null) {  
        daftarMahasiswa.remove(mahasiswa);  
        daftarNilai.removeIf(nilai -> nilai.mahasiswa.equals(mahasiswa));  
        System.out.println("Mahasiswa dengan NIM " + mahasiswa.nim + " telah dihapus.");  
    } else {  
        System.out.println(x:"Tidak ada mahasiswa dalam antrian penghapusan.");  
    }  
}
```

3. Mencari Nilai Mahasiswa

4. Urut Data Nilai

5. Tambah Mahasiswa ke Antrian Penghapusan

6. Hapus Mahasiswa dari Antrian

7. Keluar

Pilih: 5

Masukkan NIM Mahasiswa yang akan ditambahkan ke antrian penghapusan: 002

Mahasiswa dengan NIM 002 ditambahkan ke dalam antrian penghapusan.

```
=====
SISTEM PENGOLAHAN DATA NILAI MAHASISWA SEMESTER
=====
1. Input Nilai
2. Tampil Nilai
3. Mencari Nilai Mahasiswa
4. Urut Data Nilai
5. Tambah Mahasiswa ke Antrian Penghapusan
6. Hapus Mahasiswa dari Antrian
7. Keluar
Pilih: 6
Mahasiswa dengan NIM 002 telah dihapus.
```