LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN STRUKTUR DATA
PRAKTIKUM 15

Oleh:

RANDA HERU KUSUMA

NIM. 2341760009

SIB-1F / 25

D-IV SISTEM INFORMASI BISNIS JURUSAN

TEKNOLOGI INFORMASI POLITEKNIK

NEGERI MALANG

Link github: https://github.com/randaheru/Praktikum15.git

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.

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

```
List<Object> l = new ArrayList<>();
l.add(e:1);
l.add(e:2);
l.add(e:3);
l.add(e:"Cireng");

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

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

```
List<String> names = new LinkedList<>();
names.add(e:"Noureen");
names.add(e:"Akhleema");
names.add(e:"Shannum");
names.add(e:"Uwais");
names.add(e:"Al-Qarni");
```

16.2.2. Verifikasi Hasil Percobaan

Verifikasi hasil kompilasi kode program Anda dengan gambar berikut ini.

```
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?
   Dalam baris kode 25-36, ArrayList dideklarasikan tanpa tipe parameter (List l = new ArrayList();), yang berarti ArrayList tersebut menggunakan tipe Object secara default. Karena Object adalah tipe dasar dari semua kelas di Java, ArrayList ini dapat menampung semua jenis data (primitif atau objek).
2. Modifikasi baris kode 25-36 seingga data yang ditampung hanya satu jenis atau spesifik tipe tertentu!

```
List<Object> l = new ArrayList<>();
l.add(e:1);
l.add(e:2);
l.add(e:3);
l.add(e:"Cireng");

System.out.printf(format:"Elemen 0: %d 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: %d 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<>();
names.add(e:"Noureen");
```

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());
```

5. Dari penambahan kode tersebut, silakan dijalankan dan apakah yang dapat Anda jelaskan!

Metode push() menambahkan elemen ke depan LinkedList.

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

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.

```
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");

// Printing all elements using enhanced for-loop
for (String fruit : fruits) {
    System.out.printf(format:"%s ", fruit);
}

System.out.println("\n" + fruits.toString());

// Pop elements until the stack is empty
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();
// Print the stack using an iterator
for (Iterator<String> it = fruits.iterator(); it.hasNext(); ) {
    String fruit = it.next();
    System.out.printf(format:"%s ", fruit);
}

System.out.println();
// Print the stack using a stream
fruits.stream().forEach(e -> System.out.printf(format:"%s ", e));
System.out.println();

// Print the stack using a for-loop with index access
for (int i = 0; i < fruits.size(); i++) {
    System.out.printf(format:"%s ", fruits.get(i));
}
```

16.3.2.

n gambar berikut ini.

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

16.3.3. Pertanyaan Percobaan

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

push(): Menunjukkan operasi stack yaitu menempatkan elemen di atas stack, yang membuat maksud lebih jelas saat membaca kode.
add(): Menunjukkan penambahan elemen ke daftar atau vektor, yang mungkin tidak segera menunjukkan operasi khusus stack.

2. Silakan hilangkan baris 43 dan 44, apakah yang akan terjadi? Mengapa bisa demikian?

Menghapus baris 43 dan 44 menghilangkan penambahan "Melon" dan "Durian" ke stack setelah setiap operasi pop. Akibatnya, stack menjadi kosong setelah semua elemen di-pop, dan tidak ada elemen yang tersisa untuk dicetak oleh iterator, stream, atau for-loop yang mengikuti operasi pop tersebut.

3. Jelaskan fungsi dari baris 46-49?

Iterator: Baris 46-49 menggunakan Iterator untuk mengiterasi elemen-elemen dalam fruits.
fruits.iterator() membuat iterator untuk stack fruits.
it.hasNext() memeriksa apakah masih ada elemen berikutnya dalam stack.
it.next() mengembalikan elemen berikutnya dalam iterasi.

4. Silakan ganti baris kode 25, *Stack<String>* menjadi *List<String>* dan apakah yang terjadi? Mengapa bisa demikian?

```
List<String> fruits = new Stack<>();
```

Dengan mendeklarasikan fruits sebagai List, Anda bisa mengganti implementasi stack ke yang lain (seperti ArrayList, LinkedList) tanpa mengubah kode lainnya.

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

```
// Mengganti elemen terakhir menjadi "Strawberry"
fruits.set(fruits.size() - 1, element:"Strawberry");
```

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

```
// Menambahkan 3 buah baru
fruits.add(e:"Mango");
fruits.add(e:"Guava");
fruits.add(e:"Avocado");
```

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.

```java
public class Mahasiswa {
    private String nim;
    private String nama;
    private String notelp;

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

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

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

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

public void hapus(int index) {
    if (index >= 0 && index < mahasiswas.size()) {
        mahasiswas.remove(index);
    } else {
        System.out.println(x:"Index out of bounds");
    }
}

public void update(int index, Mahasiswa mhs) {
    if (index >= 0 && index < mahasiswas.size()) {
        mahasiswas.set(index, mhs);
    } else {
        System.out.println(x:"Index out of bounds");
    }
}

public void tampil() {
    for (Mahasiswa mhs : mahasiswas) {
        System.out.println(mhs);
    }
}
```

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

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

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

```
Run main | Debug main | Run | Debug
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:"Akhleema", notelp:"021xx2");
    Mahasiswa m2 = new Mahasiswa(nim:"201236", nama:"Shannum", notelp:"021xx3");

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

    // Menampilkan list mahasiswa
    System.out.println(x:"Daftar Mahasiswa:");
    lm.tampil();

    // Update mahasiswa
    int index = lm.linearSearch(nim:"201235");
    if (index != -1) {
        lm.update(index, new Mahasiswa(nim:"201235", nama:"Akhleema Lela", notelp:"021xx2"));
    } else {
        System.out.println(x:"Mahasiswa dengan NIM 201235 tidak ditemukan.");
    }

    System.out.println(x:"\nSetelah update:");
    lm.tampil();
}
```

16.4.2. Verifikasi Hasil Percobaan

Verifikasi hasil kompilasi kode program Anda dengan gambar berikut ini.

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

16.4.3. Pertanyaan Percobaan

1. Pada fungsi tambah() yang menggunakan unlimited argument itu menggunakan konsep apa? Dan kelebihannya apa?
   Pada fungsi tambah() yang menggunakan unlimited argument (Mahasiswa... mahasiswa), digunakan konsep varargs (variadic arguments)
2. Pada fungsi linearSearch() di atas, silakan diganti dengan fungsi binarySearch() dari collection!

```
public int binarySearch(String nim) {
    // Sort list based on nim before performing binary search
    Collections.sort(mahasiswas, (a, b) -> a.getNim().compareTo(b.getNim()));
    // Perform binary search
    int index = Collections.binarySearch(mahasiswas, new Mahasiswa(nim, nama:"", notelp:""), (a, b) -> a.getNim().compareTo(b.getNim()));
    return index;
}
```

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

```
public void sortAscending() {
    Collections.sort(mahasiswas, (a, b) -> a.getNim().compareTo(b.getNim()));
}

public void sortDescending() {
    Collections.sort(mahasiswas, (a, b) -> b.getNim().compareTo(a.getNim()));
}
```

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
```

1. Buat class Mahasiswa

```java
public class Mahasiswa {
    String nim;
    String nama;
    String telf;

    Mahasiswa(String nim, String nama, String telf) {
        this.nim = nim;
        this.nama = nama;
        this.telf = telf;
    }
```

2. Buat class MataKuliah

```java
public class MataKuliah {
    String kode;
    String nama;
    int sks;

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

3. Buat class Nilai

```java
public class Nilai {
    Mahasiswa mahasiswa;
    MataKuliah matakuliah;
    double nilai;

    Nilai(Mahasiswa mahasiswa, MataKuliah matakuliah, double nilai) {
        this.mahasiswa = mahasiswa;
        this.matakuliah = matakuliah;
        this.nilai = nilai;
    }
}
```

4. Buat class SistemNilai

```java
public class SistemNilai {
    List<Mahasiswa> daftarMahasiswa = new ArrayList<>();
    List<MataKuliah> daftarMataKuliah = new ArrayList<>();
    List<Nilai> daftarNilai = new ArrayList<>();
    Queue<Mahasiswa> antrianHapus = new LinkedList<>();

    void tambahMahasiswa(String nim, String nama, String telf)
        Mahasiswa mahasiswa = new Mahasiswa(nim, nama, telf);
        daftarMahasiswa.add(mahasiswa);
    }
```

5. Buat class Main

```java
public class Main {
    Run main | Debug main | Run | Debug
    public static void main(String[] args) {
        SistemNilai sistem = new SistemNilai();
        Scanner scanner = new Scanner(System.in);
        int pilihan;

        // Menambahkan beberapa data mahasiswa dan mata kuliah
        sistem.tambahMahasiswa(nim:"20001", nama:"Tha lhah", telf:"021xxx");
        sistem.tambahMahasiswa(nim:"20002", nama:"Zubair", telf:"021xxx");
        sistem.tambahMahasiswa(nim:"20003", nama:"Abdur-Rahman", telf:"021xxx");
        sistem.tambahMahasiswa(nim:"20004", nama:"Sa'ad", telf:"021xxx");
        sistem.tambahMahasiswa(nim:"20005", nama:"Sa'id", telf:"021xxx");
        sistem.tambahMahasiswa(nim:"20006", nama:"Ubaidah", telf:"021xxx");
```

2. Tambahkan prosedur hapus data mahasiswa melalui implementasi Queue pada collections

Tugas nomor 1!

```java
    void antrianHapusMahasiswa(String nim) {
        Mahasiswa mahasiswa = daftarMahasiswa.stream().filter(m -> m.nim.equals(nim)).findFirst().orElse(other:null);
        if (mahasiswa != null) {
            antrianHapus.add(mahasiswa);
            System.out.println("Mahasiswa dengan NIM " + nim + " ditambahkan ke dalam antrian penghapusan.");
        } else {
            System.out.println(x:"Mahasiswa tidak ditemukan.");
        }
    }

    void hapusMahasiswa() {
        Mahasiswa 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.");
        }
    }
}
```

```
=====================================
 SISTEM PENGOLAHAN DATA NILAI MAHASISWA
=====================================
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:
```