Nama : Muhammad Hikmal Al-Ghifary
Kelas : TI - 1B
Matkul : Praktikum Algoritma dan Struktur Data

LINKED LIST

Percobaan 1

1. Implementasikan Class Mahasiswa15

```java
public class Mahasiswa15{
    String nim, nama, kelas;
    double ipk;

    Mahasiswa15(){}

    Mahasiswa15(String nm, String name, String kls, double ip){
        nim = nm;
        nama = name;
        kelas = kls;
        ipk = ip;
    }

    public void tampilInformasi() {
        System.out.println(nama + " | " + nim + " | " + kelas + " | " + ipk);
    }
}
```

2. Implementasi class NodeMahasiswa15

```java
public class NodeMahasiswa15 {
    Mahasiswa15 data;
    NodeMahasiswa15 next;

    public NodeMahasiswa15(Mahasiswa15 data, NodeMahasiswa15 next){
        this.data = data;
        this.next = next;
    }
}
```

3. Implementasi class SingleLinkedList15

```java
public class SingleLinkedList15 {
    NodeMahasiswa15 head;
    NodeMahasiswa15 tail;

    boolean isEmpty() {
        return (head == null);
    }

    public void print() {
        if (!isEmpty()) {
            NodeMahasiswa15 tmp = head;
            System.out.println(x:"Isi Linked List:\t");
            while (tmp != null) {
                tmp.data.tampilInformasi();
                tmp = tmp.next;
            }
            System.out.println(x:"");
        } else {
            System.out.println(x:"Linked list masih kosong");
            System.out.println();
        }
    }

    public void addFirst(Mahasiswa15 input) {
        NodeMahasiswa15 ndInput = new NodeMahasiswa15(input, next:null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
        } else {
            ndInput.next = head;
            head = ndInput;
        }
    }
```

```
    public void addLast(Mahasiswa15 input) {
        NodeMahasiswa15 ndInput = new NodeMahasiswa15(input, next:null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
        } else {
            tail.next = ndInput;
            tail = ndInput;
        }
    }

    public void insertAfter(String key, Mahasiswa15 input) {
        NodeMahasiswa15 ndInput = new NodeMahasiswa15(input, next:null);
        NodeMahasiswa15 temp = head;
        do {
            if (temp.data.nama.equalsIgnoreCase(key)) {
                ndInput.next = temp.next;
                temp.next = ndInput;
                if (ndInput.next == null) {
                    tail = ndInput;
                }
                break;
            }
            temp = temp.next;
        } while (temp != null);
    }
```

```
     public void insertAt(int index, Mahasiswa15 input) {
         if (index < 0) {
             System.out.println(x:"indeks salah");
         } else if(index == 0) {
             addFirst(input);
         } else {
             NodeMahasiswa15 temp = head;
             for (int i = 0; i < index - 1; i++) {
                 temp = temp.next;
             }
             temp.next = new NodeMahasiswa15(input, temp.next);
             if (temp.next.next == null) {
                 tail = temp.next;
             }
         }
     }
 }
```

4. Implementasi class SLLMain15

```
public class SLLMain15 {
    Run | Debug
    public static void main(String[] args) {
        SingleLinkedList15 sll = new SingleLinkedList15();

        Mahasiswa15 mhs1 = new Mahasiswa15(nm:"2201", name:"Alvaro", kls:"TI-1A", ip:3.8);
        Mahasiswa15 mhs2 = new Mahasiswa15(nm:"2202", name:"Bimon", kls:"TI-1B", ip:3.5);
        Mahasiswa15 mhs3 = new Mahasiswa15(nm:"2203", name:"Cintia", kls:"TI-1A", ip:3.9);
        Mahasiswa15 mhs4 = new Mahasiswa15(nm:"2204", name:"Dirga", kls:"TI-1B", ip:3.6);

        sll.print();

        sll.addFirst(mhs4);
        sll.print();

        sll.addLast(mhs1);
        sll.print();

        sll.insertAfter(key:"Dirga", mhs3);
        sll.print();

        sll.insertAt(index:2, mhs2);
        sll.print();
    }
}
```

5. Verifikasi Percobaan

```
Linked list masih kosong

Isi Linked List:
Dirga | 2204 | TI-1B | 3.6

Isi Linked List:
Dirga | 2204 | TI-1B | 3.6
Alvaro | 2201 | TI-1A | 3.8

Isi Linked List:
Dirga | 2204 | TI-1B | 3.6
Cintia | 2203 | TI-1A | 3.9
Alvaro | 2201 | TI-1A | 3.8

Isi Linked List:
Dirga | 2204 | TI-1B | 3.6
Cintia | 2203 | TI-1A | 3.9
Bimon | 2202 | TI-1B | 3.5
Alvaro | 2201 | TI-1A | 3.8

PS C:\Users\Muhammad Hikmal AG
```

PERTANYAAN

1. Hasil compile kode program di baris pertama menghasilkan “Linked List Kosong” dikarenakan pada bagian tersebut memang belum ada satupun data yang dimasukkan, sehingga yang dieksekusi oleh method print adalah bagian else yang menunjukkan kondisi saat Linked List masih kosong
2. Kegunaan variable temp secara umum pada setiap method adalah sebagai variabel yang digunakan untuk pengalihan agar tidak mengubah head. Variabel ini digunakan untuk melakukan pengecekan pada nilai lain di indeks berikutnya (yang dijalankan). Jika mengubah nilai head dikarenakan tidak ada variabel pengalihan ini, maka data yang berada di depan akan dianggap hilang oleh programnya.
3. Modifikasi
   Menambahkan import java.util.Scanner dan Scanner di bagian atas untuk menerima input

```
29          while (true) {
30              System.out.print(s:"Tambah data baru? (y/n): ");
31              String jawab = sc15.nextLine();
32
33              if (jawab.equalsIgnoreCase(anotherString:"y")) {
34                  System.out.println(x:"----- INPUT DATA -----");
35                  System.out.print(s:"Masukkan nama: ");
36                  String nama = sc15.nextLine();
37                  System.out.print(s:"Masukkan NIM: ");
38                  String nim = sc15.nextLine();
39                  System.out.print(s:"Masukkan kelas: ");
40                  String kelas = sc15.nextLine();
41                  System.out.print(s:"Masukkan IPK: ");
42                  double ipk = sc15.nextDouble();
43                  Mahasiswa15 mhs = new Mahasiswa15(nim, nama, kelas, ipk);
44
45                  System.out.println();
46
47                  System.out.println(x:"Silahkan pilih menu: ");
48                  System.out.println(x:"1. Menambahkan data Diawal");
49                  System.out.println(x:"2. Menambahkan data Setelah Nama Tertentu");
50                  System.out.println(x:"3. Menambahkan data di index tertentu");
51                  System.out.println(x:"4. Menambahkan data di Akhir");
52                  System.out.print(s:"Pilihan : ");
53                  int pilih = sc15.nextInt();
54                  sc15.nextLine();
```

```
55                  switch (pilih) {
56                  case 1:
57                      sll.addFirst(mhs);
58                      sll.print();
59                      break;
60                  case 2:
61                      System.out.print(s:"Input data setelah: ");
62                      String dicari = sc15.nextLine();
63                      sll.insertAfter(dicari, mhs);
64                      sll.print();
65                      break;
66                  case 3:
67                      System.out.print(s:"Posisi index yang diinginkan: ");
68                      int index = sc15.nextInt();
69                      sc15.nextLine();
70                      sll.insertAt(index, mhs);
71                      sll.print();
72                      break;
73                      case 4:
74                      sll.addLast(mhs);
75                      sll.print();
76                      break;
77                   default:
78                      break;
79                  }
80              } else if(jawab.equalsIgnoreCase(anotherString:"n")) {
81                  break;
82              } else {
83                  System.out.println(x:"Tidak Valid, jawab dengan Y/N");
84              }
85          }
```

Percobaan 2

1. Tambahkan method untuk mendapatkan data, indexOf, removeFirst, removeLast, remove, dan menghapus node dengan menggunakan index pada class SingleLinkedList15

```java
//PERCOBAAN 2
public void getData(int index) {
    NodeMahasiswa15 tmp = head;
    for (int i = 0; i < index; i++) {
        tmp = tmp.next;
    }
    tmp.data.tampilInformasi();
}

public int indexOf(String key) {
    NodeMahasiswa15 tmp = head;
    int index = 0;
    while (tmp != null && !tmp.data.nama.equalsIgnoreCase(key)) {
        tmp = tmp.next;
        index++;
    }

    if (tmp == null) {
        return -1;
    } else {
        return index;
    }
}
```

```java
public void removeFirst() {
    if(isEmpty()) {
        System.out.println("Linked List masih kosong, tidak dapat dihapus!");
    } else if (head == tail) {
        head = tail = null;
    } else {
        head = head.next;
    }
}

public void removeLast() {
    if (isEmpty()) {
        System.out.println("Linked List masih kosong. Tidak dapat dihapus!");
    } else if(head == tail) {
        head = tail= null;
    } else {
        NodeMahasiswa15 temp = head;
        while (temp.next != tail) {
            temp = temp.next;
        }
        temp.next = null;
        tail = temp;
    }
}
```

```java
public void remove(String key) {
    if(isEmpty()) {
        System.out.println("Linked List masih kosong, tidak dapat dihapus");
    } else {
        NodeMahasiswa15 tmp = head;
        while(tmp != null) {
            if((tmp.data.nama.equalsIgnoreCase(key)) && (tmp == head)) {
                tmp.next = tmp.next.next;
                if(tmp.next == null) {
                    tail = tmp;
                }
                break;
            }
            tmp = tmp.next;
        }
    }
}

public void removeAt(int index) {
    if(index == 0) {
        removeFirst();
    } else {
        NodeMahasiswa15 tmp = head;
        for(int i = 0; i < index - 1; i++) {
            tmp = tmp.next;
        }
        tmp.next = tmp.next.next;
        if(tmp.next == null) {
            tail = tmp;
        }
```

2. Akses dan hapus data di method main

```
87          System.out.println(x:"Data index 1: ");
88          sll.getData(index:1);
89
90          System.out.println("Data mahasiswa Bimon berada pada index: " + sll.indexOf(key:"Bimon"));
91          System.out.println();
92
93          sll.removeFirst();
94          sll.removeLast();
95          sll.print();
96          sll.removeAt(index:0);
97          sll.print();
```

3. Verifikasi Percobaan

```
Data index 1:
Cintia | 2203 | TI-1A | 3.9
Data mahasiswa Bimon berada pada index: 2

Isi Linked List:
Cintia | 2203 | TI-1A | 3.9
Bimon | 2202 | TI-1B | 3.5

Isi Linked List:
Bimon | 2202 | TI-1B | 3.5
```

PERTANYAAN

1. Keyword break pada fungsi remove berguna untuk menghentikan perulangan while disaat kondisi yang ditentukan sudah tidak memenuhi lagi. Jika tidak ada break, maka perulangan akan terus berlanjut tanpa henti walaupun sudah melebihi kondisi yang ditentukan.
2. Kegunaan kode pada method remove tersebut adalah untuk menghapus node berdasarkan key dan memperbarui tail jika node yang dihapus adalah node terakhir.