

LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR DATA
LINKED LIST



AHMAD SOFYAN BADAWI

244107020073

KELAS TI-1B

PRODI D-IV TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG

2025

Pembuatan Single Linked List:

Langkah-langkah:

1. Pada Project yang sudah dibuat pada Minggu sebelumnya. Buat folder atau package baru bernama Jobsheet11 di dalam repository Praktikum ASD.
2. Tambahkan class-class berikut:
3. Implementasikan Class Mahasiswa00 sesuai dengan diagram class berikut ini :
4. Implementasi class Node seperti gambar berikut ini

```
public class Node04 {  
    Mahasiswa04 data;  
    Node04 next;  
  
    Node04(Mahasiswa04 data, Node04 next) {  
        this.data = data;  
        this.next = next;  
    }  
}
```

- 5.
6. Tambahkan attribute head dan tail pada class SingleLinkedList

```
public class SingleLinkedList04 {  
    Node04 head;  
    Node04 tail;
```

7. Sebagai langkah berikutnya, akan diimplementasikan method-method yang terdapat pada SingleLinkedList.
8. Tambahkan method isEmpty().

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

9. Implementasi method untuk mencetak dengan menggunakan proses traverse.

```

public void print() {
    if (!isEmpty()) {
        Node04 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 Kosong!");
    }
}
}

```

10. Implementasikan method addFirst().

```

public void addFirst(Mahasiswa04 input) {
    Node04 ndInput = new Node04(input, next:null);
    if (isEmpty()) {
        head = ndInput;
        tail = ndInput;
    } else {
        ndInput.next = head;
        head = ndInput;
    }
}
}

```

11. Implementasikan method addLast().

```

public void addLast(Mahasiswa04 input) {
    Node04 ndInput = new Node04(input, next:null);
    if (isEmpty()) {
        head = ndInput;
        tail = ndInput;
    } else {
        tail.next = ndInput;
        tail = ndInput;
    }
}
}

```

12. Implementasikan method insertAfter, untuk memasukkan node yang memiliki data input setelah node yang memiliki data key.

```

public void insertAfter(String key, Mahasiswa04 input) {
    Node04 ndInput = new Node04(input, next:null);
    Node04 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);
}

```

13. Tambahkan method penambahan node pada indeks tertentu.

```

public void insertAt(int index, Mahasiswa04 input) {
    if (index < 0) {
        System.out.println(x:"Index salah");
    } else if (index == 0) {
        addFirst(input);
    } else {
        Node04 temp = head;
        for (int i = 0; i < index - 1; i++) {
            temp = temp.next;
        }
        temp.next = new Node04(input, temp.next);
        if (temp.next.next == null) {
            tail = temp.next;
        }
    }
}

```

14. Pada class SLLMain00, buatlah fungsi main, kemudian buat object dari class SingleLinkedList.

```

public static void main(String[] args) {
    SingleLinkedList04 sll = new SingleLinkedList04();
}

```

15. Buat empat object mahasiswa dengan nama mhs1, mhs2, mhs3, mhs4 kemudian isi data setiap object melalui konstruktor.

```

Mahasiswa04 mhs1 = new Mahasiswa04(nm:"24212200",name:"Alvaro", kls:"1A", ip:4.0);
Mahasiswa04 mhs2 = new Mahasiswa04(nm:"23212201",name:"Bimon", kls:"2B", ip:3.8);
Mahasiswa04 mhs3 = new Mahasiswa04(nm:"22212202",name:"Cintia", kls:"3C", ip:3.5);
Mahasiswa04 mhs4 = new Mahasiswa04(nm:"21212203",name:"Dirga", kls:"4D", ip:3.6);

```

16. Tambahkan Method penambahan data dan pencetakan data di setiap penambahannya agar terlihat perubahannya.

```
sll.print();
sll.addFirst(mhs4);
sll.print();
sll.addLast(mhs1);
sll.print();
sll.insertAfter(key:"Dirga", mhs3);
sll.insertAt(index:2, mhs2);
sll.print();
```

Verifikasi:

```
Linked List Kosong!
Isi Linked List:
Dirga  21212203      4D      3.6

Isi Linked List:
Dirga  21212203      4D      3.6
Alvaro 24212200      1A      4.0

Isi Linked List:
Dirga  21212203      4D      3.6
Cintia 22212202      3C      3.5
Bimon  23212201      2B      3.8
Alvaro 24212200      1A      4.0
```

Pertanyaan:

1. Mengapa hasil compile kode program di baris pertama menghasilkan “Linked List Kosong”?

Jawab: Karena belum ada data yang dimasukkan

2. Jelaskan kegunaan variable temp secara umum pada setiap method!

Jawab: Untuk mengakses data tanpa merusak data aslinya

3. Lakukan modifikasi agar data dapat ditambahkan dari keyboard!

Jawab:

Modifikasi Elemen pada Single Linked List:

Langkah-langkah:

1. Implementasikan method untuk mengakses data dan indeks pada linked list
2. Tambahkan method untuk mendapatkan data pada indeks tertentu pada class Single Linked List

```
public void getData(int index) {
    Node04 tmp = head;
    for (int i = 0; i < index; i++) {
        tmp = tmp.next;
    }
    tmp.data.tampilInformasi();
}
```

3. Implementasikan method indexOf.

```
public int indexOf(String key) {  
    Node04 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;  
    }  
}
```

4. Tambahkan method removeFirst pada class SingleLinkedList

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

5. Tambahkan method untuk menghapus data pada bagian belakang pada class SingleLinkedList

```
public void removeLast() {  
    if (isEmpty()) {  
        System.out.println(x:"Linked List masih kosong, tidak dapat dihapus!");  
    } else if (head == tail) {  
        head = tail = null;  
    } else {  
        Node04 tmp = head;  
        while (tmp.next != tail) {  
            tmp = tmp.next;  
        }  
        tmp.next = null;  
        tail = tmp;  
    }  
}
```

6. Sebagai langkah berikutnya, akan diimplementasikan method remove

```

public void removeBy(String key) {
    if (isEmpty()) {
        System.out.println(x:"Linked List masih kosong, tidak dapat dihapus!");
    } else {
        Node04 tmp = head;
        while (tmp != null) {
            if (tmp.data.nama.equalsIgnoreCase(key) && tmp == head) {
                this.removeFirst();
                break;
            } else if (tmp.data.nama.equalsIgnoreCase(key)) {
                tmp.next = tmp.next.next;
                if (tmp.next == null) {
                    tail = tmp;
                }
                break;
            }
            tmp = tmp.next;
        }
    }
}

```

7. Implementasi method untuk menghapus node dengan menggunakan index

```

public void removeAt(int index) {
    if (index == 0) {
        removeFirst();
    } else {
        Node04 tmp = head;
        for (int i = 0; i < index - 1; i++) {
            tmp = tmp.next;
        }
        tmp.next = tmp.next.next;
        if (tmp.next == null) {
            tail = tmp;
        }
    }
}

```

8. Kemudian, coba lakukan pengaksesan dan penghapusan data di method main pada class SLLMain dengan menambahkan kode berikut

```

//Percobaan 2
System.out.println(x:"data index 1 : ");
sll.getData(index:1);

System.out.println("data mahasiswa an Bimon berada pada index : " + sll.indexOf(key:"Bimon"));
System.out.println();

sll.removeFirst();
sll.removeLast();
sll.print();
sll.removeAt(index:0);
sll.print();

```

9. Jalankan class SLLMain

Verifikasi:

```

Percobaan 2!
data index 1 :
Cintia 22212202      3C      3.5
data mahasiswa an Bimon berada pada index : 2

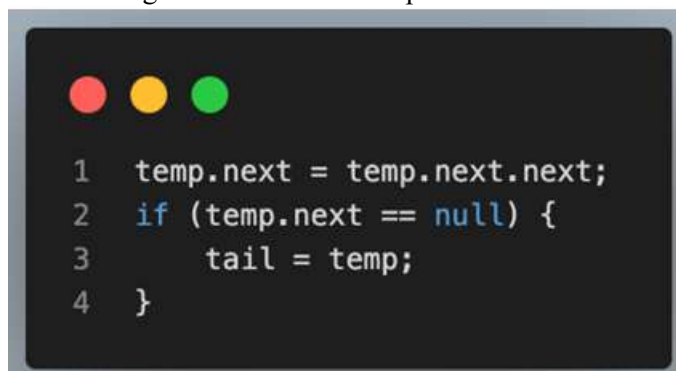
Isi Linked List:
Cintia 22212202      3C      3.5
Bimon 23212201      2B      3.8

Isi Linked List:
Bimon 23212201      2B      3.8

```

Pertanyaan:

1. Mengapa digunakan keyword break pada fungsi remove? Jelaskan!
Jawab: Supaya while tidak berjalan terus menerus
2. Jelaskan kegunaan kode dibawah pada method remove



```

1 temp.next = temp.next.next;
2 if (temp.next == null) {
3     tail = temp;
4 }

```

Jawab: temp.next = temp.next.next digunakan untuk traverse dan pada kode if digunakan untuk memastikan apakah data pada temp.next.next bernilai null, jika nilainya null maka data tersebut merupakan data paling akhir dan kemudian data tersebut diubah menjadi tail

Tugas:

Buatlah implementasi program antrian layanan unit kemahasiswaan sesuai dengan berikut ini :
Implementasi antrian menggunakan Queue berbasis Linked List! **(Hasil Ada Pada Git)**

- a. Implementasi antrian menggunakan Queue berbasis Linked List!
- b. Program merupakan proyek baru bukan modifikasi dari percobaan
- c. Ketika seorang mahasiswa akan mengantri, maka dia harus mendaftarkan datanya
- d. Cek antrian kosong, Cek antrian penuh, Mengosongkan antrian.
- e. Menambahkan antriana
- f. Memanggil antrian
- g. Menampilkan antrian terdepan dan antrian paling akhir
- h. Menampilkan jumlah mahasiswa yang masih mengantre.

