

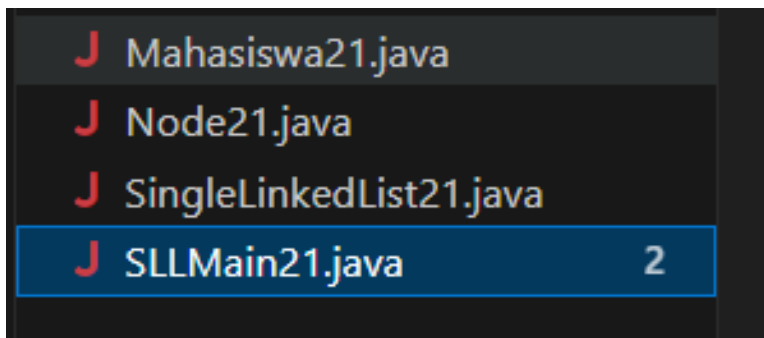
Nama : Siti Mutmainah  
Kelas : TI-1B/21  
NIM : 244107020143

## Praktikum Pembuatan Single Linked List

Didalam praktikum ini, kita akan mempraktikkan bagaimana membuat Single Linked List dengan representasi data berupa Node, pengaksesan linked list dan metode penambahan data.

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:
  - a. Mahasiswa00.java
  - b. Node00.java
  - c. SingleLinkedList00.java
  - d. SLLMain00.java

**Ganti 00 dengan nomer Absen Anda**



3. Implementasikan Class Mahasiswa00 sesuai dengan diagram class berikut ini :

Mahasiswa
nim: String nama: String kelas: String ipk: double
Mahasiswa() Mahasiswa(nm: String, name: String, kls: String, ip: double) tampilInformasi(): void

4. Implementasi class Node seperti gambar berikut ini

```
public class Node21 {  
    Mahasiswa21 data;  
    Node21 next;  
  
    public Node21(Mahasiswa21 data, Node21 next){  
        this.data = data;  
        this.next = next;  
    }  
}
```

5. Tambahkan attribute **head** dan **tail** pada class SingleLinkedList

```
public class SingleLinkedList21 {  
    Node21 head;  
    Node21 tail;
```

6. Sebagai langkah berikutnya, akan diimplementasikan method-method yang terdapat pada SingleLinkedList.

7. Tambahkan method **isEmpty()**.

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

8. Implementasi method untuk mencetak dengan menggunakan proses traverse.

```
public void print() {  
    if (!isEmpty()) {  
        Node21 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");  
    }  
}
```

9. Implementasikan method **addFirst()**.

```
public void addFirst(Mahasiswa21 input){  
    Node21 ndInput = new Node21(input,next:null);  
    if (isEmpty()){  
        head = ndInput;  
        tail = ndInput;  
    }else{  
        ndInput.next=head;  
        head=ndInput;  
    }  
}
```

10. Implementasikan method **addLast()**.

```
public void addLast(Mahasiswa21 input){
    Node21 ndInput = new Node21(input, next:null);
    if (isEmpty()){
        head = ndInput;
        tail = ndInput;
    }else{
        tail.next=ndInput;
        tail=ndInput;
    }
}
```

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

```
public void insertAfter(String key, Mahasiswa21 input){
    Node21 ndInput = new Node21(input, next:null);
    Node21 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);
}
```

12. Tambahkan method penambahan node pada indeks tertentu.

```
public void insertAt(int index, Mahasiswa21 input){
    if (index < 0){
        System.out.println(x:"indeks salah");
    } else if(index == 0){
        addFirst(input);
    }else{
        Node21 temp = head;
        for (int i=0; i<index - 1; i++){
            temp = temp.next;
        }
        temp.next = new Node21(input, temp.next);
        if (temp.next.next == null){
            tail = temp.next;
        }
    }
}
```

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

```
public class SLLMain21 {
    Run | Debug
    public static void main(String[] args) {
        SingleLinkedList21 sll = new SingleLinkedList21();
    }
}
```

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

```
Mahasiswa21 mhs1 = new Mahasiswa21(nim:"24212200", nama:"Alvaro", kelas:"1A", ipk:4.0);
Mahasiswa21 mhs2 = new Mahasiswa21(nim:"23212201", nama:"Bimon", kelas:"2B", ipk:3.8);
Mahasiswa21 mhs3 = new Mahasiswa21(nim:"22212202", nama:"Cintia", kelas:"3C", ipk:3.5);
Mahasiswa21 mhs4 = new Mahasiswa21(nim:"21212203", nama:"Dirga", kelas:"4D", ipk:3.6);
```

15. 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 Hasil Percobaan

```
'--enable-preview' '-XX:+Shor\workspaceStorage\c0
r\workspaceStorage\c09ccf44ea35e7e90b575e62c083926
t11_cab07d27\bin' 'SLLMain21'
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 saat baris pertama sll.print(); dipanggil, belum ada data apa pun yang ditambahkan ke dalam linked list. Objek SingleLinkedList21 memang sudah dibuat, tapi list-nya masih kosong alias head == null. Maka dari itu, method print() akan mendeteksi kondisi ini dan mencetak pesan:

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

Jawab: Secara umum temp bersifat sementara yang digunakan untuk menelusuri isi linked list. Karena kita tidak bisa langsung melompat ke node tertentu di linked list, kita perlu mulai dari head, lalu maju satu per satu menggunakan temp.

3. Lakukan modifikasi agar data dapat ditambahkan dari keyboard!

Jawab:

```
import java.util.Scanner;

public class SLLMain21 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        SingleLinkedList21 sll = new SingleLinkedList21();

        System.out.print(s:"Masukkan jumlah mahasiswa: ");
        int jumlah = sc.nextInt();
        sc.nextLine();

        for (int i = 0; i < jumlah; i++) {
            System.out.println("Data Mahasiswa ke-" + (i + 1));
            System.out.print(s:"NIM   : ");
            String nim = sc.nextLine();
            System.out.print(s:"Nama   : ");
            String nama = sc.nextLine();
            System.out.print(s:"Kelas : ");
            String kelas = sc.nextLine();
            System.out.print(s:"IPK   : ");
            double ipk = sc.nextDouble();
            sc.nextLine();

            Mahasiswa21 mhs = new Mahasiswa21(nim, nama, kelas, ipk);
            sll.addLast(mhs);
        }

        System.out.println(x:"\nData Mahasiswa dalam Linked List:");
        sll.print();
    }
}

Masukkan jumlah mahasiswa: 2
Data Mahasiswa ke-1
NIM   : 1234
Nama   : ray
Kelas : 1D
IPK   : 3.9
Data Mahasiswa ke-2
NIM   : 1235
Nama   : iin
Kelas : 1B
IPK   : 4.0

Data Mahasiswa dalam Linked List:
Isi Linked List:
ray 1234 1D 3.9
iin 1235 1B 4.0
```

## Modifikasi Elemen pada Single Linked List

Didalam praktikum ini, kita akan mempraktekkan bagaimana mengakses elemen, mendapatkan indeks dan melakukan penghapusan data pada Single Linked List.:

### Langkah-langkah Percobaan

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){
    Node21 tmp = head;
    for (int i = 0; i < index; i++) {
        tmp = tmp.next;
    }
    tmp.data.tampilInformasi();
}
```

3. Implementasikan method **indexOf**.

```
public int indexOf(String key){
    Node21 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{
        Node21 temp = head;
        while (temp.next != tail) {
            temp = temp.next;
        }
        temp.next=null;
        tail=temp;
    }
}
```

6. Sebagai langkah berikutnya, akan diimplementasikan method **remove**

```
public void remove(String key) {
    if (isEmpty()) {
        System.out.println(x:"Linked List masih kosong, tidak dapat dihapus!");
    } else {
        Node21 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 {
        Node21 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

```
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();

System.out.println(x:"\nData Mahasiswa dalam Linked List:");
sll.print();
}
```

9. Jalankan class SLLMain

## Verifikasi Hasil Percobaan


```
6c\redhat.java\jdt_ws\Jobsheet11_cab07d27\bin' 'SLMain21'
Masukkan jumlah mahasiswa: 4
Data Mahasiswa ke-1
NIM : 12
Nama : i
Kelas : 1B
IPK : 3.9
Data Mahasiswa ke-2
NIM : 13
Nama : a
Kelas : 1B
IPK : 4.0
Data Mahasiswa ke-3
NIM : 14
Nama : q
Kelas : 1B
IPK : 3.7
Data Mahasiswa ke-4
NIM : 15
Nama : j
Kelas : 1B
IPK : 4.0
data index 1 :
a 13 1B 4.0
data mahasiswa an bimon berada pada index : -1

Isi Linked List:
a 13 1B 4.0
q 14 1B 3.7

Isi Linked List:
q 14 1B 3.7
```

## Pertanyaan

1. Mengapa digunakan keyword break pada fungsi remove? Jelaskan!  
Jawab: Untuk menghentikan perulangan while setelah data yang dicari ditemukan dan berhasil dihapus dari linked list, menggunakan break bisa keluar dari loop menghindari iterasi yang tidak perlu.
2. Jelaskan kegunaan kode dibawah pada method remove



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

Jawab: Kode ini berfungsi untuk menghapus node setelah temp, dan jika node adalah yang terakhir, maka tail akan diperbarui agar linked list tetap konsisten.



## Tugas

**Waktu pengerjaan : 50 menit**

Buatlah implementasi program antrian layanan unit kemahasiswaan sesuai dengan berikut ini :

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

### TUGAS MAHASISWA

```
J TugasMahasiswa.java > TugasMahasiswa > TugasMahasiswa(String, String, String)
1 public class TugasMahasiswa {
2     String nim;
3     String nama;
4     String layanan;
5
6     public TugasMahasiswa(String name, String nim, String layanan){
7         this.nim = nim;
8         this.nama = name;
9         this.layanan = layanan;
10    }
11
12    public void tampilInformasi() {
13        System.out.println(nama + " - " + nim + " - " + layanan);
14    }
15 }
16
```

### TUGAS NODE

```
J TugasNode.java > TugasNode > TugasNode(TugasMahasiswa, TugasNode)
1 public class TugasNode {
2     TugasMahasiswa data;
3     TugasNode next;
4
5     public TugasNode(TugasMahasiswa data, TugasNode next) {
6         this.data = data;
7         this.next = next;
8     }
9 }
10
```

### TUGAS QUEUE

```

J TugasQueue.java > TugasQueue > AntrianTerakhir()
1 public class TugasQueue {
2     TugasNode head, tail;
3     int size = 0;
4
5     boolean isEmpty() {
6         return (head == null);
7     }
8
9     boolean isFull() {
10        return !isEmpty();
11    }
12
13    public void print() {
14        if (!isEmpty()) {
15            TugasNode tmp = head;
16            System.out.println(x:"Isi Linked List:\t");
17            while (tmp != null) {
18                tmp.data.tampilInformasi();
19                tmp = tmp.next;
20            }
21            System.out.println(x:"");
22        } else {
23            System.out.println(x:"Linked List kosong");
24        }
25    }
26
27    public void kosongkanAntrian() {
28        this.head = null;
29        this.tail = null;
30        this.size = 0;
31        System.out.println(x:"Semua antrian telah dikosongkan.");
32    }
33
34    public void tampilkanSemuaAntrian() {
35        if (isEmpty()) {
36            System.out.println(x:"Antrian saat ini kosong.");
37            return;
38        }
39        System.out.println(x:"daftar mahasiswa dalam antrian: ");
40        TugasNode tmp = head;
41
42        while (tmp != null) {
43            tmp.data.tampilInformasi();
44            tmp = tmp.next;
45        }
46        System.out.println(x:"-----");
47    }
48
49    public void TambahAntrian(TugasMahasiswa input) {
50        TugasNode ndInput = new TugasNode(input, next:null);
51        if (isEmpty()) {
52            head = ndInput;
53            tail = ndInput;
54        } else {
55            tail.next = ndInput;
56            tail = ndInput;
57        }
58        size++;
59        System.out.println(x:"Data masuk dalam antrian");
60    }
61 }

```

```

public TugasMahasiswa panggilAntrian() {
    if (isEmpty()) {
        System.out.println(x:"Linked List masih kosong, tidak dapat dihapus!");
        return null;
    }
    TugasMahasiswa pglmhs = head.data;
    head = head.next;
    size--;
    if (head == null)
        tail = null;
    System.out.println(x:"Antrian berikut adalah:");
    pglmhs.tampilInformasi();
    return pglmhs;
}

public TugasMahasiswa AntrianDepan() {
    if (isEmpty()) {
        System.out.println(x:"Antrian saat ini kosong.");
        return null;
    }
    System.out.println(x:"Antrian terdepan adalah : ");
    head.data.tampilInformasi();
    return head.data;
}
}

```

```

public void jumlahAntrian() {
    System.out.println("Jumlah mahasiswa dalam antrian: " + size);
}

public TugasMahasiswa AntrianTerakhir() {
    if (isEmpty()) {
        System.out.println(x:"Antrian saat ini kosong.");
        return null;
    }
    System.out.println(x:"Antrin terakhir adalah : ");
    tail.data.tampilInformasi();
    return tail.data;
}
}

```

## TUGAS MAIN

```

import java.util.Scanner;

public class TugasMain {

    public class SLIMain25 {
        public static void main(String[] args) {
            Scanner sc = new Scanner(System.in);
            int pilihan;
            TugasQueue sll = new TugasQueue();

            do {
                System.out.println(x:"\n==== MENU LAYANAN KEMAHASISWAAN ====");
                System.out.println(x:"1. Tambah Antrian Mahasiswa");
                System.out.println(x:"2. Panggil Mahasiswa");
                System.out.println(x:"3. Tampilkan Semua Antrian");
                System.out.println(x:"4. Tampilkan Antrian Terdepan");
                System.out.println(x:"5. Tampilkan Antrian Belakang");
                System.out.println(x:"6. Tampilkan Jumlah Mahasiswa dalam Antrian");
                System.out.println(x:"7. Kosongkan Antrian");
                System.out.println(x:"8. Cek Apakah Antrian Kosong");
                System.out.println(x:"9. Cek Apakah Antrian Penuh");
                System.out.println(x:"10. Keluar");
                System.out.print(s:"Pilih menu: ");
                pilihan = sc.nextInt();
                sc.nextLine();
            } while (pilihan != 10);
        }
    }
}

```

```

switch (pilihan) {
    case 1:
        System.out.println(x:"Isi data mahasiswa ");
        System.out.print(s:"Nama : ");
        String nama = sc.nextLine();
        System.out.print(s:"NIM : ");
        String nim = sc.nextLine();
        System.out.print(s:"Masukkan Layanan : ");
        String layanan = sc.nextLine();
        TugasMahasiswa mhsBaru = new TugasMahasiswa(nama, nim, layanan);
        sll.TambahAntrian(mhsBaru);
        break;
    case 2:
        sll.panggilAntrian();
        break;
    case 3:
        sll.tampilkanSemuaAntrian();
        break;
    case 4:
        sll.AntrianDepan();
        break;
    case 5:
        sll.AntrianTerakhir();
        break;
    case 6:
        sll.jumlahAntrian();
        break;
    case 7:
        sll.kosongkanAntrian();
        break;
    case 8:
        System.out.println(sll.isEmpty() ? "Antrian kosong." : "Antrian tidak kosong.");
        break;
    case 9:
        System.out.println(sll.isFull() ? "Antrian penuh." : "Antrian belum penuh.");
        break;
    case 10:
        System.out.println(x:"Terima kasih! Program selesai.");
        break;
    default:
        System.out.println(x:"Pilihan tidak valid!");
}
} while (pilihan != 10);
}

```

## HASIL RUNNING

```

===== MENU LAYANAN KEMAHASISWAAN =====
1. Tambah Antrian Mahasiswa
2. Panggil Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan Antrian Terdepan
5. Tampilkan Antrian Belakang
6. Tampilkan Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
8. Cek Apakah Antrian Kosong
9. Cek Apakah Antrian Penuh
10. Keluar
Pilih menu: 1
Isi data mahasiswa
Nama : iin
NIM : 123
Masukkan Layanan : mengumpulkan tugas
Data masuk dalam antrian

```

```
===== MENU LAYANAN KEMAHASISWAAN =====
1. Tambah Antrian Mahasiswa
2. Panggil Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan Antrian Terdepan
5. Tampilkan Antrian Belakang
6. Tampilkan Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
8. Cek Apakah Antrian Kosong
9. Cek Apakah Antrian Penuh
10. Keluar
Pilih menu: 1
Isi data mahasiswa
Nama : adel
NIM : 234
Masukkan Layanan : mengumpulkan tugas
Data masuk dalam antrian
```

```
===== MENU LAYANAN KEMAHASISWAAN =====
1. Tambah Antrian Mahasiswa
2. Panggil Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan Antrian Terdepan
5. Tampilkan Antrian Belakang
6. Tampilkan Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
8. Cek Apakah Antrian Kosong
9. Cek Apakah Antrian Penuh
10. Keluar
Pilih menu: 2
Antrian berikut adalah:
iin - 123 - mengumpulkan tugas
```

```
===== MENU LAYANAN KEMAHASISWAAN =====
1. Tambah Antrian Mahasiswa
2. Panggil Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan Antrian Terdepan
5. Tampilkan Antrian Belakang
6. Tampilkan Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
8. Cek Apakah Antrian Kosong
9. Cek Apakah Antrian Penuh
10. Keluar
Pilih menu: 3
daftar mahasiswa dalam antrian:
adel - 234 - mengumpulkan tugas
-----
```

```
===== MENU LAYANAN KEMAHASISWAAN =====
1. Tambah Antrian Mahasiswa
2. Panggil Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan Antrian Terdepan
5. Tampilkan Antrian Belakang
6. Tampilkan Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
8. Cek Apakah Antrian Kosong
9. Cek Apakah Antrian Penuh
10. Keluar
Pilih menu: 4
Antrian terdepan adalah :
adel - 234 - mengumpulkan tugas
```

```
===== MENU LAYANAN KEMAHASISWAAN =====
1. Tambah Antrian Mahasiswa
2. Panggil Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan Antrian Terdepan
5. Tampilkan Antrian Belakang
6. Tampilkan Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
8. Cek Apakah Antrian Kosong
9. Cek Apakah Antrian Penuh
10. Keluar
Pilih menu: 5
Antrin terakhir adalah :
adel - 234 - mengumpulkan tugas
```

```
===== MENU LAYANAN KEMAHASISWAAN =====
1. Tambah Antrian Mahasiswa
2. Panggil Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan Antrian Terdepan
5. Tampilkan Antrian Belakang
6. Tampilkan Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
8. Cek Apakah Antrian Kosong
9. Cek Apakah Antrian Penuh
10. Keluar
Pilih menu: 6
Jumlah mahasiswa dalam antrian: 1
```

===== MENU LAYANAN KEMAHASISWAAN =====

1. Tambah Antrian Mahasiswa
2. Panggil Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan Antrian Terdepan
5. Tampilkan Antrian Belakang
6. Tampilkan Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
8. Cek Apakah Antrian Kosong
9. Cek Apakah Antrian Penuh
10. Keluar

Pilih menu: 9

Antrian penuh.

===== MENU LAYANAN KEMAHASISWAAN =====

1. Tambah Antrian Mahasiswa
2. Panggil Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan Antrian Terdepan
5. Tampilkan Antrian Belakang
6. Tampilkan Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
8. Cek Apakah Antrian Kosong
9. Cek Apakah Antrian Penuh
10. Keluar

Pilih menu: 7

Semua antrian telah dikosongkan.

===== MENU LAYANAN KEMAHASISWAAN =====

1. Tambah Antrian Mahasiswa
2. Panggil Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan Antrian Terdepan
5. Tampilkan Antrian Belakang
6. Tampilkan Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
8. Cek Apakah Antrian Kosong
9. Cek Apakah Antrian Penuh
10. Keluar

Pilih menu: 8

Antrian kosong.