

Laporan Hasil Pratikum
Algoritma Dan Struktur Data

Jobsheet 11



Nama : Zacky Rio Orlando

NIM : 244107020086

Kelas : 1E

Program Studi D-IV Teknik Informatika

Jurusan Teknologi Informasi

Praktikum

2025

2.1 Pembuatan Single Linked List

Class Mahasiswa27

```
public class Mahasiswa27 {
    String nim;
    String nama;
    String kelas;
    double ipk;

    public Mahasiswa27(){

    }

    public Mahasiswa27(String nim, String nama, String kelas, double ipk) {
        this.nim = nim;
        this.nama = nama;
        this.kelas = kelas;
        this.ipk = ipk;
    }

    public void tampilkanInformasi() {
        System.out.println(nim + "\t\t" + nama + "\t\t" + kelas + "\t\t" +
ipk);
    }
}
```

Class NodeMahasiswa27

```
public class NodeMahasiswa27 {
    Mahasiswa27 data;
    NodeMahasiswa27 next;

    public NodeMahasiswa27(Mahasiswa27 data, NodeMahasiswa27 next) {
        this.data = data;
        this.next = next;
    }
}
```

Class SingleLinkedList27

```
public class SingleLinkedList27 {
    NodeMahasiswa27 head;
    NodeMahasiswa27 tail;

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

    public void print() {
        if (!isEmpty()) {
            NodeMahasiswa27 tmp = head;
            System.out.print("Isi Linked List:\t");
            System.out.println();
            while (tmp != null) {
                tmp.data.tampilkanInformasi();
                tmp = tmp.next;
            }
            System.out.println("");
        } else {
            System.out.println("Linked list kosong");
        }
    }

    public void addFirst (Mahasiswa27 input) {
        NodeMahasiswa27 ndInput = new NodeMahasiswa27 (input, null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
        } else {
            ndInput.next = head;
            head = ndInput;
        }
    }
}
```

```

public void addLast (Mahasiswa27 input) {
    NodeMahasiswa27 ndInput = new NodeMahasiswa27 (input, null);
    if (isEmpty()) {
        head = ndInput;
        tail = ndInput;
    } else {
        tail.next = ndInput;
        tail = ndInput;
    }
}

public void insertAfter (String key, Mahasiswa27 input) {
    NodeMahasiswa27 ndInput = new NodeMahasiswa27(input, null);
    NodeMahasiswa27 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, Mahasiswa27 input) {
    if (index < 0) {
        System.out.println("indeks salah");
    } else if (index == 0) {
        addFirst(input);
    } else {
        NodeMahasiswa27 temp = head;
        for (int i = 0; i < index - 1; i++) {
            temp = temp.next;
        }
        temp.next = new NodeMahasiswa27(input, temp.next);
        if (temp.next.next == null) {
            tail = temp.next;
        }
    }
}
}

```

Class SLLMain27

```

public class SLLMain27 {
    public static void main(String[] args) {
        SingleLinkedList27 sll = new SingleLinkedList27();

        Mahasiswa27 mhs1 = new Mahasiswa27("21212203", "Dirga", "4D", 3.6);
        Mahasiswa27 mhs2 = new Mahasiswa27("22212202", "Cintia", "3C",
3.5);
        Mahasiswa27 mhs3 = new Mahasiswa27("23212201", "Bimon", "2B", 3.8);
        Mahasiswa27 mhs4 = new Mahasiswa27("24212200", "Alvaro", "1A",
4.0);

        sll.print();
        sll.addFirst(mhs4);
        sll.print();
        sll.addLast(mhs1);
        sll.print();
        sll.insertAfter("Dirga", mhs3);
        sll.insertAt(2, mhs2);
        sll.print();
    }
}

```

Hasil Output dari kode program

```

Linked list kosong
Isi Linked List:
24212200      Alvaro      1A      4.0

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

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

```

2.1.2 Pertanyaan

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

Jawaban :

Karena saat perintah `sll.print()`; pertama kali dijalankan, linked list belum punya data apa pun. Objek `sll` baru dibuat dan belum ada mahasiswa yang dimasukkan, sehingga method `isEmpty()` mengembalikan nilai `true` karena head masih kosong. Akibatnya, program langsung mencetak "Linked list kosong".

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

Jawaban :

Digunakan sebagai penunjuk sementara untuk membantu menelusuri node dalam linked list, mulai dari head hingga ke tail. Dalam method print(), temp dipakai untuk membaca dan menampilkan data setiap node satu per satu. Pada method insertAfter(), temp digunakan untuk mencari node dengan nama tertentu agar bisa menyisipkan data baru. Sedangkan pada method insertAt(), temp membantu menemukan posisi berdasarkan indeks agar data bisa disisipkan di lokasi yang tepat.

3. Lakukan modifikasi agar data dapat ditambahkan dari keyboard!

Jawaban :

Modifikasi pada Class SLLMain27

```
import java.util.Scanner;

public class SLLMain27 {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        SingleLinkedList27 sll = new SingleLinkedList27();
        System.out.print("Masukkan jumlah mahasiswa: ");
        int n = input.nextInt();
        input.nextLine();

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

            Mahasiswa27 mhs = new Mahasiswa27(nim, nama, kelas, ipk);
            sll.addLast(mhs);
        }

        System.out.println("\nData mahasiswa dalam linked list:");
        sll.print();
    }
}
```

Hasil Output

```
Masukkan jumlah mahasiswa: 3
Data mahasiswa ke-1
NIM: 244107020086
Nama: Zacky
Kelas: 1E
IPK: 3,75
Data mahasiswa ke-2
NIM: 244107020090
Nama: Rio
Kelas: 1A
IPK: 3,9
Data mahasiswa ke-3
NIM: 244107020095
Nama: Orlando
Kelas: 1B
IPK: 3,5

Data mahasiswa dalam linked list:
Isi Linked List:
244107020086      Zacky      1E      3.75
244107020090      Rio      1A      3.9
244107020095      Orlando  1B      3.5
```

2.2 Modifikasi Elemen pada Single Linked List

Class Mahasiswa27

```
public class Mahasiswa27 {
    String nim;
    String nama;
    String kelas;
    double ipk;

    public Mahasiswa27(){

    }

    public Mahasiswa27(String nim, String nama, String kelas, double ipk) {
        this.nim = nim;
        this.nama = nama;
        this.kelas = kelas;
        this.ipk = ipk;
    }

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

Class NodeMahasiswa27

```
public class NodeMahasiswa27 {
    Mahasiswa27 data;
    NodeMahasiswa27 next;

    public NodeMahasiswa27(Mahasiswa27 data, NodeMahasiswa27 next) {
        this.data = data;
        this.next = next;
    }
}
```

Class SingleLinkedList27


```

public class SingleLinkedList27 {
    NodeMahasiswa27 head;
    NodeMahasiswa27 tail;

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

    public void print() {
        if (!isEmpty()) {
            NodeMahasiswa27 tmp = head;
            System.out.print("Isi Linked List:\t");
            System.out.println();
            while (tmp != null) {
                tmp.data.tampilInformasi();
                tmp = tmp.next;
            }
            System.out.println("");
        } else {
            System.out.println("Linked list kosong");
        }
    }

    public void addFirst (Mahasiswa27 input) {
        NodeMahasiswa27 ndInput = new NodeMahasiswa27 (input, null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
        } else {
            ndInput.next = head;
            head = ndInput;
        }
    }

    public void addLast (Mahasiswa27 input) {
        NodeMahasiswa27 ndInput = new NodeMahasiswa27 (input, null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
        } else {
            tail.next = ndInput;
            tail = ndInput;
        }
    }
}

```

```

public void insertAfter (String key, Mahasiswa27 input) {
    NodeMahasiswa27 ndInput = new NodeMahasiswa27(input, null);
    NodeMahasiswa27 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, Mahasiswa27 input) {
    if (index < 0) {
        System.out.println("indeks salah");
    } else if (index == 0) {
        addFirst(input);
    } else {
        NodeMahasiswa27 temp = head;
        for (int i = 0; i < index - 1; i++) {
            temp = temp.next;
        }
        temp.next = new NodeMahasiswa27(input, temp.next);
        if (temp.next.next == null) {
            tail = temp.next;
        }
    }
}

public void getData (int index) {
    NodeMahasiswa27 tmp = head;
    for (int i = 0; i < index; i++) {
        tmp = tmp.next;
    }
    tmp.data.tampilInformasi();
}

```

```

public int indexOf(String key) {
    NodeMahasiswa27 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;
    }
}

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 {
        NodeMahasiswa27 temp = head;
        while (temp.next != tail) {
            temp = temp.next;
        }
        temp.next = null;
        tail = temp;
    }
}

```

```

        public void remove(String key) {
            if (isEmpty()) {
                System.out.println("Linked List masih Kosong, tidak dapat
dihapus!");
            } else {
                NodeMahasiswa27 temp = head;
                while (temp != null) {
                    if ((temp.data.nama.equalsIgnoreCase(key)) && (temp ==
head)) {
                        this.removeFirst();
                        break;
                    } else if (temp.data.nama.equalsIgnoreCase(key)) {
                        temp.next = temp.next.next;
                        if (temp.next == null) {
                            tail = temp;
                        }
                        break;
                    }
                    temp = temp.next;
                }
            }
        }

        public void removeAt (int index) {
            if (index == 0) {
                removeFirst();
            } else {
                NodeMahasiswa27 temp = head;
                for (int i = 0; i < index - 1; i++) {
                    temp = temp.next;
                }
                temp.next = temp.next.next;
                if (temp.next == null) {
                    tail = temp;
                }
            }
        }
    }
}

```

Class SLLMain

```

public class SLLMain27 {
    public static void main(String[] args) {
        SingleLinkedList27 sll = new SingleLinkedList27();

        Mahasiswa27 mhs1 = new Mahasiswa27("21212203", "Dirga", "4D", 3.6);
        Mahasiswa27 mhs2 = new Mahasiswa27("22212202", "Cintia", "3C",
3.5);
        Mahasiswa27 mhs3 = new Mahasiswa27("23212201", "Bimon", "2B", 3.8);
        Mahasiswa27 mhs4 = new Mahasiswa27("24212200", "Alvaro", "1A",
4.0);

        sll.print();
        sll.addFirst(mhs1);
        sll.print();
        sll.addLast(mhs4);
        sll.print();
        sll.insertAfter("Dirga", mhs3);
        sll.insertAt(1, mhs2);
        sll.print();

        System.out.println("data index 1 : ");
        sll.getData(1);

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

        sll.removeFirst();
        sll.removeLast();
        sll.print();
        sll.removeAt(0);
        sll.print();
    }
}

```

Hasil Output kode Program diatas

```

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

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

Isi Linked List:
23212201          Bimon          2B          3.8

```

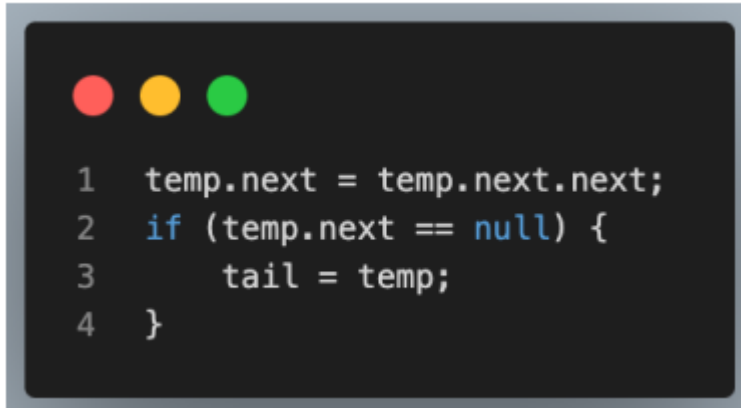
2.2.3 Pertanyaan

1. Mengapa digunakan keyword break pada fungsi remove? Jelaskan!

Jawaban :

Digunakan supaya setelah data mahasiswa yang ingin dihapus ketemu dan dihapus, perulangan langsung berhenti. Ini supaya program tidak terus mencari dan menghapus data yang sama berkali-kali atau melakukan pengecekan yang tidak perlu. Kalau break tidak dipakai, perulangan akan terus berjalan walaupun data sudah dihapus, yang bisa bikin program error dan lambat. Jadi, break membuat proses hapus data jadi lebih cepat dan tepat.

2. Jelaskan kegunaan kode dibawah pada method remove



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

Jawaban :

Kode `temp.next = temp.next.next`; digunakan untuk melewati satu node, yaitu node yang ingin dihapus, sehingga node itu tidak lagi terhubung ke linked list. Artinya, node sebelum yang dihapus langsung terhubung ke node setelahnya. Lalu, bagian `if (temp.next == null)` berarti kalau node setelahnya sudah tidak ada, maka node temp sekarang menjadi node terakhir, jadi tail harus diarahkan ke temp. Intinya, kode ini menjaga agar daftar tetap tersambung dengan benar dan ujung akhir (tail) tetap tepat setelah penghapusan data.

3. 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.

Jawaban :

Class MahasiswaTugas27

```
public class MahasiswaTugas27 {
    String nim, nama, prodi;

    public MahasiswaTugas27(String nim, String nama, String prodi) {
        this.nim = nim;
        this.nama = nama;
        this.prodi = prodi;
    }

    public void tampil() {
        System.out.println("NIM    : " + nim);
        System.out.println("Nama   : " + nama);
        System.out.println("Prodi  : " + prodi);
        System.out.println("-----");
    }
}
```

Class NodeTugas27

```
public class NodeTugas27 {
    MahasiswaTugas27 data;
    NodeTugas27 next;

    public NodeTugas27(MahasiswaTugas27 data) {
        this.data = data;
        this.next = null;
    }
}
```

Class QueueTugas27

```
public class QueueTugas27 {
    NodeTugas27 front, rear;
    int size, max;

    public QueueTugas27(int max) {
        this.max = max;
        this.size = 0;
        front = rear = null;
    }

    public boolean isEmpty() {
        return size == 0;
    }

    public boolean isFull() {
        return size == max;
    }

    public void enqueue(MahasiswaTugas27 mhs) {
        if (isFull()) {
            System.out.println("Antrian sudah penuh!");
            return;
        }
        NodeTugas27 newNode = new NodeTugas27(mhs);
        if (isEmpty()) {
            front = rear = newNode;
        } else {
            rear.next = newNode;
            rear = newNode;
        }
        size++;
        System.out.println("Mahasiswa berhasil ditambahkan ke antrian.");
    }

    public void dequeue() {
        if (isEmpty()) {
            System.out.println("Antrian kosong, tidak ada yang bisa
dipanggil.");
            return;
        }
        System.out.println("Memanggil mahasiswa:");
        front.data.tampil();
        front = front.next;
        size--;
        if (front == null) rear = null;
    }
}
```



```

public void peekFront() {
    if (isEmpty()) {
        System.out.println("Antrian kosong.");
    } else {
        System.out.println("Mahasiswa di depan antrian:");
        front.data.tampil();
    }
}

public void peekRear() {
    if (isEmpty()) {
        System.out.println("Antrian kosong.");
    } else {
        System.out.println("Mahasiswa di akhir antrian:");
        rear.data.tampil();
    }
}

public void printQueue() {
    if (isEmpty()) {
        System.out.println("Antrian kosong.");
        return;
    }
    System.out.println("Isi antrian:");
    NodeTugas27 temp = front;
    while (temp != null) {
        temp.data.tampil();
        temp = temp.next;
    }
}

public void clear() {
    front = rear = null;
    size = 0;
    System.out.println("Antrian dikosongkan.");
}

public int getSize() {
    return size;
}
}

```

Class MainTugas27

```

import java.util.Scanner;

public class MainTugas27 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        QueueTugas27 antrian = new QueueTugas27(10);

        int pilihan;
        do {
            System.out.println("\n--- MENU LAYANAN UNIT KEMAHASISWAAN ---");
            System.out.println("1. Tambah Antrian Mahasiswa");
            System.out.println("2. Panggil Antrian");
            System.out.println("3. Cek Antrian Kosong");
            System.out.println("4. Cek Antrian Penuh");
            System.out.println("5. Tampilkan Antrian Terdepan");
            System.out.println("6. Tampilkan Antrian Terakhir");
            System.out.println("7. Tampilkan Jumlah Antrian");
            System.out.println("8. Tampilkan Semua Antrian");
            System.out.println("9. Kosongkan Antrian");
            System.out.println("0. Keluar");
            System.out.print("Pilih menu: ");
            pilihan = sc.nextInt(); sc.nextLine();

            switch (pilihan) {
                case 1:
                    System.out.print("NIM   : ");
                    String nim = sc.nextLine();
                    System.out.print("Nama   : ");
                    String nama = sc.nextLine();
                    System.out.print("Prodi  : ");
                    String prodi = sc.nextLine();
                    antrian.enqueue(new MahasiswaTugas27(nim, nama, prodi));
                    break;
                case 2:
                    antrian.dequeue();
                    break;
                case 3:
                    System.out.println("Antrian kosong: " + antrian.isEmpty());
                    break;
                case 4:
                    System.out.println("Antrian penuh: " + antrian.isFull());
                    break;
                case 5:
                    antrian.peekFront();
                    break;
                case 6:
                    antrian.peekRear();
                    break;
                case 7:
                    System.out.println("Jumlah mahasiswa dalam antrian: " +
antrian.getSize());
                    break;
                case 8:
                    antrian.printQueue();
                    break;
                case 9:
                    antrian.clear();
                    break;
                case 0:
                    System.out.println("Terima kasih!");
                    break;
                default:
                    System.out.println("Menu tidak tersedia.");
            }
        } while (pilihan != 0);

        sc.close();
    }
}

```

Hasil Outputnya

```
--- MENU LAYANAN UNIT KEMAHASISWAAN ---
1. Tambah Antrian Mahasiswa
2. Panggil Antrian
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Antrian Terdepan
6. Tampilkan Antrian Terakhir
7. Tampilkan Jumlah Antrian
8. Tampilkan Semua Antrian
9. Kosongkan Antrian
0. Keluar
Pilih menu: 1
NIM   : 12345
Nama  : Zacky
Prodi : Teknik Informatika
Mahasiswa berhasil ditambahkan ke antrian.

--- MENU LAYANAN UNIT KEMAHASISWAAN ---
1. Tambah Antrian Mahasiswa
2. Panggil Antrian
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Antrian Terdepan
6. Tampilkan Antrian Terakhir
7. Tampilkan Jumlah Antrian
8. Tampilkan Semua Antrian
9. Kosongkan Antrian
0. Keluar
Pilih menu: 1
NIM   : 54321
Nama  : Rio
Prodi : Teknik Informatika
Mahasiswa berhasil ditambahkan ke antrian.
```

--- MENU LAYANAN UNIT KEMAHASISWAAN ---

1. Tambah Antrian Mahasiswa
2. Panggil Antrian
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Antrian Terdepan
6. Tampilkan Antrian Terakhir
7. Tampilkan Jumlah Antrian
8. Tampilkan Semua Antrian
9. Kosongkan Antrian
0. Keluar

Pilih menu: 7

Jumlah mahasiswa dalam antrian: 2

--- MENU LAYANAN UNIT KEMAHASISWAAN ---

1. Tambah Antrian Mahasiswa
2. Panggil Antrian
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Antrian Terdepan
6. Tampilkan Antrian Terakhir
7. Tampilkan Jumlah Antrian
8. Tampilkan Semua Antrian
9. Kosongkan Antrian
0. Keluar

Pilih menu: 8

Isi antrian:

NIM : 12345

Nama : Zacky

Prodi : Teknik Informatika

NIM : 54321

Nama : Rio

Prodi : Teknik Informatika

--- MENU LAYANAN UNIT KEMAHASISWAAN ---

1. Tambah Antrian Mahasiswa
2. Panggil Antrian
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Antrian Terdepan
6. Tampilkan Antrian Terakhir
7. Tampilkan Jumlah Antrian
8. Tampilkan Semua Antrian
9. Kosongkan Antrian
0. Keluar

Pilih menu: 5

Mahasiswa di depan antrian:

NIM : 12345

Nama : Zacky

Prodi : Teknik Informatika

--- MENU LAYANAN UNIT KEMAHASISWAAN ---

1. Tambah Antrian Mahasiswa
2. Panggil Antrian
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Antrian Terdepan
6. Tampilkan Antrian Terakhir
7. Tampilkan Jumlah Antrian
8. Tampilkan Semua Antrian
9. Kosongkan Antrian
0. Keluar

Pilih menu: 6

Mahasiswa di akhir antrian:

NIM : 54321

Nama : Rio

Prodi : Teknik Informatika

--- MENU LAYANAN UNIT KEMAHASISWAAN ---

1. Tambah Antrian Mahasiswa
2. Panggil Antrian
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Antrian Terdepan
6. Tampilkan Antrian Terakhir
7. Tampilkan Jumlah Antrian
8. Tampilkan Semua Antrian
9. Kosongkan Antrian
0. Keluar

Pilih menu: 3

Antrian kosong: false

--- MENU LAYANAN UNIT KEMAHASISWAAN ---

1. Tambah Antrian Mahasiswa
2. Panggil Antrian
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Antrian Terdepan
6. Tampilkan Antrian Terakhir
7. Tampilkan Jumlah Antrian
8. Tampilkan Semua Antrian
9. Kosongkan Antrian
0. Keluar

Pilih menu: 4

Antrian penuh: false

--- MENU LAYANAN UNIT KEMAHASISWAAN ---

1. Tambah Antrian Mahasiswa
2. Panggil Antrian
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Antrian Terdepan
6. Tampilkan Antrian Terakhir
7. Tampilkan Jumlah Antrian
8. Tampilkan Semua Antrian
9. Kosongkan Antrian
0. Keluar

Pilih menu: 2

Memanggil mahasiswa:

NIM : 12345

Nama : Zacky

Prodi : Teknik Informatika

--- MENU LAYANAN UNIT KEMAHASISWAAN ---

1. Tambah Antrian Mahasiswa
2. Panggil Antrian
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Antrian Terdepan
6. Tampilkan Antrian Terakhir
7. Tampilkan Jumlah Antrian
8. Tampilkan Semua Antrian
9. Kosongkan Antrian
0. Keluar

Pilih menu: 7

Jumlah mahasiswa dalam antrian: 1

--- MENU LAYANAN UNIT KEMAHASISWAAN ---

1. Tambah Antrian Mahasiswa
2. Panggil Antrian
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Antrian Terdepan
6. Tampilkan Antrian Terakhir
7. Tampilkan Jumlah Antrian
8. Tampilkan Semua Antrian
9. Kosongkan Antrian
0. Keluar

Pilih menu: 8

Isi antrian:

NIM : 54321

Nama : Rio

Prodi : Teknik Informatika

--- MENU LAYANAN UNIT KEMAHASISWAAN ---

1. Tambah Antrian Mahasiswa
2. Panggil Antrian
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Antrian Terdepan
6. Tampilkan Antrian Terakhir
7. Tampilkan Jumlah Antrian
8. Tampilkan Semua Antrian
9. Kosongkan Antrian
0. Keluar

Pilih menu: 9

Antrian dikosongkan.

--- MENU LAYANAN UNIT KEMAHASISWAAN ---

1. Tambah Antrian Mahasiswa
2. Panggil Antrian
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Antrian Terdepan
6. Tampilkan Antrian Terakhir
7. Tampilkan Jumlah Antrian
8. Tampilkan Semua Antrian
9. Kosongkan Antrian
0. Keluar

Pilih menu: 7

Jumlah mahasiswa dalam antrian: 0

--- MENU LAYANAN UNIT KEMAHASISWAAN ---

1. Tambah Antrian Mahasiswa
2. Panggil Antrian
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Antrian Terdepan
6. Tampilkan Antrian Terakhir
7. Tampilkan Jumlah Antrian
8. Tampilkan Semua Antrian
9. Kosongkan Antrian
0. Keluar

Pilih menu: 8

Antrian kosong.