Nama : Dicky Darmawan
Kelas : TI 1B
No. Absen : 08
NIM : 244107020037

Percobaan 1: Operasi Dasar Queue

Langkah-langkah Percobaan

1. Buat folder baru bernama P1Jobsheet10 di dalam repository Praktikum ASD, kemudian buat class baru dengan nama Queue.

2. Tambahkan atribut-atribut Queue sesuai diagram class, kemudian tambahkan pula konstruktornya seperti berikut ini

```
int[] data;
int front, rear, size, max;

public Queue(int n) {
    max = n;
    data = new int[max];
    size = 0;
    front = rear = -1;
}
```

3. Buat method IsEmpty bertipe boolean yang digunakan untuk mengecek apakah queue kosong.

```
public boolean IsEmpty() {
    if (size == 0) {
        return true;
    } else {
        return false;
    }
}
```

4. Buat method IsFull bertipe boolean yang digunakan untuk mengecek apakah queue sudah penuh.

```
public boolean IsFull() {
    if (size == max) {
        return true;
    } else {
        return false;
    }
}
```

5. Buat method peek bertipe void untuk menampilkan elemen queue pada posisi paling depan.

```
public void peek() {
    if (!IsEmpty()) {
        System.out.println("Elemen terdepan: " + data[front]);
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}
```

6. Buat method print bertipe void untuk menampilkan seluruh elemen pada queue mulai dari posisi front sampai dengan posisi rear.

```
public void print() {
    if (IsEmpty()) {
        System.out.println(x:"Queue masih kosong");
    } else {
        int i = front;
        while (i != rear) {
            System.out.println(data[i] + " ");
            i = (i + 1) % 2;
        }
        System.out.println(data[i] + " ");
        System.out.println("Jumlah elemen= " + size);
    }
}
```

7. Buat method clear bertipe void untuk menghapus semua elemen pada queue

```
public void clear() {
    if (!IsEmpty()) {
        front = rear = -1;
        size = 0;
        System.out.println(x:"Queue berhasil dikosongkan");
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}
```

8. Buat method Enqueue bertipe void untuk menambahkan isi queue dengan parameter dt yang bertipe integer

```
public void enqueue(int dt) {
    if (IsFull()) {
        System.out.println(x:"Queue sudah penuh");
    } else {
        if (IsEmpty()) {
            front = rear = 0;
        } else {
            if (rear == max -1) {
                rear =  0;
            } else {
                rear++;
            }
        }
        data[rear] = dt;
        size++;
    }
}
```

9. Buat method Dequeue bertipe int untuk mengeluarkan data pada queue di posisi belakang

```
public int dequeue() {
    int dt = 0;
    if (IsEmpty()) {
        System.out.println(x:"Queue masih kosong");
    } else {
        dt = data[front];
        size--;
        if (IsEmpty()) {
            front = rear = -1;
        } else {
            if (front == max - 1) {
                front = 0;
            } else {
                front++;
            }
        }
    } return dt;
}
```

10. Selanjutnya, buat class baru dengan nama QueueMain tetap pada package Praktikum1. Buat method menu bertipe void untuk memilih menu program pada saat dijalankan.


```
public static void menu() {
    System.out.println(x:"Masukkan operasi yang diinginkan:");
    System.out.println(x:"1. Enqueue");
    System.out.println(x:"2. Dequeue");
    System.out.println(x:"3. Print");
    System.out.println(x:"4. Peek");
    System.out.println(x:"5. Clear");
    System.out.println(x:"------------------");

}
```

11. Buat fungsi main, kemudian deklarasikan Scanner dengan nama sc.

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
}
```

12. Buat variabel n untuk menampung masukan berupa jumlah maksimal elemen yang dapat disimpan pada queue.

```
System.out.print(s:"Masukkan kapasitas Queue");
int n = sc.nextInt();
```

13. Lakukan instansiasi objek Queue dengan nama Q dengan mengirimkan parameter n sebagai kapasitas elemen queue

```
Queue Q = new Queue(n); // instansiasi objek
```

14. Deklarasikan variabel dengan nama pilih bertipe integer untuk menampung pilih menu dari pengguna.

```
int pilih = sc.nextInt();
```

15. Lakukan perulangan menggunakan do-while untuk menjalankan program secara terus menerus sesuai masukan yang diberikan. Di dalam perulangan tersebut, terdapat

pemilihan kondisi menggunakan switch-case untuk menjalankan operasi queue sesuai dengan masukan pengguna.

```
int pilih = sc.nextInt();
do {
    menu();
    switch (pilih) {
        case 1:
            System.out.println(x:"Masukkan data baru: ");
            int dataMasuk = sc.nextInt();
            Q.enqueue(dataMasuk);
            break;
        case 2:
            int dataKeluar = Q.dequeue();
            if (dataKeluar != 0) {
                System.out.println("Data yang dikeluarkan: " + dataKeluar);
                break;
            }
        case 3:
            Q.print();
            break;
        case 4:
            Q.peek();
            break;
        case 5:
            Q.clear();
            break;
    }
} while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
```

16. Compile dan jalankan class QueueMain, kemudian amati hasilnya.

Verifikasi Hasil Percobaan

```
Masukkan kapasitas Queue: 4
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
------------------
1
Masukkan data baru: 34
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
------------------
1
Masukkan data baru: 25
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
------------------
4
Elemen terdepan: 34
```

Pertanyaan

1. Pada konstruktor, mengapa nilai awal atribut front dan rear bernilai -1, sementara atribut size bernilai 0?

   Front dan rear = -1 dikarenakan agar tidak menunjuk atau memilih data dalam array, dikarenakan pada saat menginputkan data pertama, front dan rear akan ditambahkan menjadi sehingga menjadi 0, menunjuk pada data pertama

   Size = 0 dikarenakan array masih kosong, atribut ini digunakan untuk mengetahui berapa banyak elemen yang ada dalam array

2. Pada method Enqueue, jelaskan maksud dan kegunaan dari potongan kode berikut!

```
if (rear == max - 1) {
    rear = 0;
```

   Maksud dari kode program tersebut adalah melakukan pengecekan, apakah index dari atribut rear sudah sama dengan max -1 atau ukuran dari queue, jika sama maka index rear akan diberi nilai 0 atau kembali ke awal untuk menginputkan elemen terbaru.

3. Pada method Dequeue, jelaskan maksud dan kegunaan dari potongan kode berikut!

```
if (front == max - 1) {
    front = 0;
```

   Melakukan pengecekan terlebih dahulu, apakah atribut index atribut front sudah mencapai ukuran max queue, jika sesuai maka atribut index atribut front akan dipindahkan ke depan atau index 0 setelah menghapus elemen queue sebelumnya

4. Pada method print, mengapa pada proses perulangan variabel i tidak dimulai dari 0 (int i=0), melainkan int i=front?

   Karena data awal dari queue tidak selalu berada di index 0, oleh karena itu digunakan atribut front untuk mengetahui elemen pertama dari array. Jika menggunakan i=0, maka elemen yang dicetak tidak sesuai dengan konsep FIFO

5. Perhatikan kembali method print, jelaskan maksud dari potongan kode berikut!

```
i = (i + 1) % max;
```

   Kode ini bermaksud untuk mengubah index i, ketika i+1 habis dibagi dengan atribut max, maka atriburt i akan kembali bernilai 0. Hal ini bertujuan agar atribut i membaca semua elemen sampai akhir karena data pertama dalam queue tidak selalu ada pada index 0.

6. Tunjukkan potongan kode program yang merupakan queue overflow!

```
if (IsFull()) {
    System.out.println(x:"Queue sudah penuh");
```

7. Pada saat terjadi queue overflow dan queue underflow, program tersebut tetap dapat berjalan dan hanya menampilkan teks informasi. Lakukan modifikasi program sehingga pada saat terjadi queue overflow dan queue underflow, program dihentikan!

   Queue overflow:

```
if (Q.IsFull()){
    System.out.println(x:"Queue sudah penuh. Program dihentikan");
    return;
```

```
Masukkan data baru:
67
Queue sudah penuh. Program dihentikan
```

   Queue underflow:

```
if (Q.IsEmpty()) {
    System.out.println(x:"Queue masih kosong. Program dihentikan");
    return;
```

```
2
Queue masih kosong. Program dihentikan
```

Percobaan 2: Antrian Layanan Akademik

Langkah-langkah Percobaan

1. Buat folder baru bernama P2Jobsheet10 di dalam repository Praktikum ASD, kemudian buat class baru dengan nama Mahasiswa.

```
public class Mahasiswa {
```

2. Tambahkan atribut-atribut Nasabah seperti pada Class Diagram, kemudian tambahkan pula konstruktornya seperti gambar berikut ini.

```
    public Mahasiswa(String nim, String nama, String prodi, String kelas)
        this.nim = nim;
        this.nama = nama;
        this.prodi = prodi;
        this.kelas = kelas;
    }
```

Dan tambahkan method tampilkanData berikut :

```
    public void tampilkanData() {
        System.out.println(nim + " - " + nama + "  - " + prodi + " - " + k
    }
}
```

3. Salin kode program class Queue pada Praktikum 1 untuk digunakan kembali pada Praktikum 2 ini, ganti nama class-nya dengan AntrianLayanan. Karena pada Praktikum 1, data yang disimpan pada queue hanya berupa array bertipe integer, sedangkan pada Praktikum 2 data yang digunakan adalah object, maka perlu dilakukan modifikasi pada class AntrianLayanan tersebut

```
Mahasiswa[] data;
int front, rear, size, max;
```

```
public AntrianLayanan(int max) {
    this.max = max;
    this.data = new Mahasiswa[max];
    this.front = 0;
    this.rear = -1;
    this.size = 0;
}
```

4. Lakukan modifikasi pada class AntrianLayanan dengan mengubah tipe int[] data menjadi Mahasiswa[] data karena pada kasus ini data yang akan disimpan berupa object Mahasiswa. Modifikasi perlu dilakukan pada atribut, method Enqueue, dan method Dequeue.

```
public void tambahAntrian(Mahasiswa mhs) {
    if (IsFull()) {
        System.out.println(x:"Antrian penuhm tidak dapat menambah mahasis
        return;
    }
    rear = (rear + 1) % max;
    data[rear] = mhs;
    size++;
    System.out.println(mhs.nama + " berhasil masuk ke antrian");
}
```

```
public Mahasiswa layaniMahasiswa() {
    if (IsEmpty()) {
        System.out.println(x:"Antrian kosong.");
        return null;
    }
    Mahasiswa mhs = data[front];
    front = (front + 1) % max;
    size--;
    return mhs;
}
```

5. Berikutnya method peek dan print yaitu untuk menampilkan data antrian layanan paling depan dan menampilkan semua data antrian layanan.

```
public void lihatTerdepan() {
    if (IsEmpty()) {
        System.out.println(x:"Antrian kosong.");
    } else {
        System.out.println(x:"Mahasiswa tterdepan: ");
        System.out.println(x:"NIM - NAMA - PRODI - KELAS");
        data[front].tampilkanData();
    }
}
```

```
public void tampilkanSemua() {
    if (IsEmpty()) {
        System.out.println(x:"Antrian kosong.");
        return;
    }
    System.out.println(x:"Daftar Mahasiswa dalam Antrian:");
    System.out.println(x:"NIM - NAMA - PRODI - KELAS");
    for (int i = 0; i < size; i++) {
        int index = (front + i) % max;
        System.out.print((i + 1) + ". ");
        data[index].tampilkanData();
    }
}
```

Ditambahkan dengan method getJumlahAntrian yaitu menampilkan nilai size

```
public int getJumlahAntrian() {
    return size;
}
```

6. Selanjutnya, buat class baru dengan nama LayananAkademikSIAKAD tetap pada package yang sama. Buat fungsi main, deklarasikan Scanner dengan nama sc.

```
import java.util.Scanner;

public class LayananAkademikSIAKAD {

    Run main | Debug main | Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner (System.in);
    }
}
```

7. Kemudian lakukan instansiasi objek AntrianLayanan dengan nama antrian dan nilai parameternya adalah nilai maksimal antrian yang ditentukan (misal sama dengan 5).

```
import java.util.Scanner;

public class LayananAkademikSIAKAD {

    Run main | Debug main | Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner (System.in);
        AntrianLayanan antrian = new AntrianLayanan(max:5);
    }
}
```

8. Deklarasikan variabel dengan nama pilihan bertipe integer untuk menampung pilih menu dari pengguna.

```
import java.util.Scanner;

public class LayananAkademikSIAKAD {

    Run main | Debug main | Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner (System.in);
        AntrianLayanan antrian = new AntrianLayanan(max:5);
        int pilihan;
    }
}
```

9. Tambahkan kode berikut untuk melakukan perulangan menu sesuai dengan masukan yang diberikan oleh pengguna.

```java
do {
    System.out.println(x:"\n===Menu Antrian Layanan Akademik===");
    System.out.println(x:"1. Tambah Mahasiswa ke Antrian");
    System.out.println(x:"2. Layani Mahasiswa");
    System.out.println(x:"3. Lihat Mahasiswa Terdepan");
    System.out.println(x:"4. Lihat Semua Antrian");
    System.out.println(x:"5. Jumlah Mahasiswa dalam Antrian");
    System.out.println(x:"0. Keluar");
    System.out.print(s:"Pilih menu: ");
    pilihan = sc.nextInt(); sc.nextLine();

    switch (pilihan) {
        case 1:
            System.out.print(s:"NIM: ");
            String nim = sc.nextLine();
            System.out.print(s:"Nama: ");
            String nama =sc.nextLine();
            System.out.print(s:"Prodi: ");
            String prodi = sc.nextLine();
            System.out.print(s:"Kelas: ");
            String kelas = sc.nextLine();
            Mahasiswa mhs = new Mahasiswa(nim, nama, prodi, kelas);
            antrian.tambahAntrian(mhs);
            break;
        case 2:
            Mahasiswa dilayani = antrian.layaniMahasiswa();
            if (dilayani != null) {
                System.out.print(s:"Melayani mahasiswa: ");
                dilayani.tampilkanData();
            }
            break;
        case 3:
            antrian.lihatTerdepan();
            break;
        case 4:
            antrian.tampilkanSemua();;
            break;
        case 5:
            System.out.println("Jumlah dalam Antrian " + antrian.getJum
            break;
        case 0:
            System.out.println(x:"Terima kasih");
            break;
        default:
            System.out.println(x:"Pilihan tidak valid");
            break;
    }
} while (pilihan != 0);
```

10. Compile dan jalankan class LayananAkademikSIAKAD, kemudian amati hasilnya.

Verifikasi Hasil Percobaan

```
===Menu Antrian Layanan Akademik===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
0. Keluar
Pilih menu: 1
NIM   : 123
Nama  : Dicky
Prodi : TI
Kelas : 1B
Dicky berhasil masuk ke antrian

===Menu Antrian Layanan Akademik===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
0. Keluar
Pilih menu: 1
NIM   : 124
Nama  : Ahmad
Prodi : TI
Kelas : 1F
Ahmad berhasil masuk ke antrian

===Menu Antrian Layanan Akademik===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
0. Keluar
Pilih menu: 4
Daftar Mahasiswa dalam Antrian:
NIM - NAMA - PRODI - KELAS
1. 123 - Dicky  - TI  - 1B -
2. 124 - Ahmad  - TI  - 1F -
```

```
===Menu Antrian Layanan Akademik===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
0. Keluar
Pilih menu: 2
Melayani mahasiswa: 123 - Dicky  - TI  - 1B

===Menu Antrian Layanan Akademik===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
0. Keluar
Pilih menu: 4
Daftar Mahasiswa dalam Antrian:
NIM - NAMA - PRODI - KELAS
1. 124 - Ahmad  - TI  - 1F -

===Menu Antrian Layanan Akademik===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
0. Keluar
Pilih menu: 5
Jumlah dalam Antrian 1

===Menu Antrian Layanan Akademik===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
0. Keluar
Pilih menu: 0
Terima kasih
```

Pertanyaan

Lakukan modifikasi program dengan menambahkan method baru bernama LihatAkhir pada class AntrianLayanan yang digunakan untuk mengecek antrian yang berada di posisi belakang. Tambahkan pula daftar menu 6. Cek Antrian paling belakang pada class LayananAkademikSIAKAD sehingga method LihatAkhir dapat dipanggil!

```
case 6:
    antrian.lihatAkhir();
    break;
```

```
=== Menu Antrian Layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
6. Cek Antrian paling belakang
0. Keluar
Pilih menu: 6
Mahasiswa Antrian belakang:
789 - Adel  - TI - 1D
```

```
public void lihatAkhir() {
    if (IsEmpty()) {
        System.out.println(x:"Antrian kosong");
    }
    System.out.println(x:"Mahasiswa Antrian belakang: ");
    data[rear].tampilkanData();
}
```

TUGAS

Buatlah program antrian untuk mengilustasikan antrian persetujuan Kartu Rencana Studi (KRS) Mahasiswa oleh Dosen Pembina Akademik (DPA). Ketika seorang mahasiswa akan mengantri, maka dia harus mendaftarkan datanya (data mahasiswa seperti pada praktikum 2). Gunakan class untuk antrian seperti pada Praktikum 1 dan 2, dengan method-method yang berfungsi :

- Cek antrian kosong, Cek antrian penuh, Mengosongkan antrian.

  Cek Antrian kosong

```
public boolean IsEmpty() {
    if (size == 0) {
        return true;
    } else {
        return false;
    }
}
```

  Cek Antrian Penuh

```
public boolean IsFull() {
    if (size == max) {
        return true;
    } else {
        return false;
    }
}
```

  Mengosongkan Queue:

```
public void clear() {
    if (!IsEmpty()) {
        front = rear = -1;
        size = 0;
        System.out.println(x:"Queue berhasil dikosongkan");
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}
```

- Menambahkan antrian, Memanggil antrian untuk proses KRS – setiap 1x panggilan terdiri dari 2 mahasiswa (pada antrian no 1 dan 2)

  Menambah Antrian

```
ublic void tambahAntrian(Mahasiswa mhs) {
    if (IsFull()) {
        System.out.println(x:"Antrian penuh tidak dapat menambah mahasiswa
        return;
    }
    rear = (rear + 1) % max;
    data[rear] = mhs;
    size++;
    System.out.println(mhs.nama + " berhasil masuk ke antrian");
}
```

  Memanggil antrian:

```
public void layaniMahasiswa() {
    if (IsEmpty()) {
        System.out.println(x:"Antrian kosong.");
        return;
    }
    if (sudahProses >= maxMhs) {
        System.out.println(x:"Semua mahasiswa telah dilayani");
        return;
    }
    int jumlahDiproses = 0;
    while (!IsEmpty() && jumlahDiproses < 2 && sudahProses <= maxMhs)
        Mahasiswa mhs = data[front];
        System.out.println(x:"Memproses KRS ");
        data[front].tampilkanData();
        front = (front + 1) % max;
        size--;
        maxMhs--;
        sudahProses++;
        jumlahDiproses++;
    }
}
```

- Menampilkan semua antrian, Menampilkan 2 antrian terdepan, Menampilkan antrian paling akhir.

  Menampilkan semua:

```
public void tampilkanSemua() {
    if (IsEmpty()) {
        System.out.println(x:"Antrian kosong.");
        return;
    }
    System.out.println(x:"Daftar Mahasiswa dalam Antrian:");
    System.out.println(x:"NIM - NAMA - PRODI - KELAS");
    for (int i = 0; i < size; i++) {
        int index = (front + i) % max;
        System.out.print((i + 1) + ". ");
        data[index].tampilkanData();
    }
}
```

  Menampilkan 2 antrian terdepan:

```
public void tampil2Terdepan() {
    if (IsEmpty()) {
        System.out.println(x:"Antrian kosong");
    }
    System.out.println(x:"2 Antrian Terdepan: ");
    for (int i = front; i < front + 2; i++) {
        data[i].tampilkanData();
    }
}
```

  Menampilkan antrian paling akhir:

```
public void lihatAkhir() {
    if (IsEmpty()) {
        System.out.println(x:"Antrian kosong");
    }
    System.out.println(x:"Mahasiswa Antrian belakang: ");
    data[rear].tampilkanData();
}
```

- Cetak jumlah antrian, Cetak jumlah yang sudah melakukan proses KRS

  Cetak jumlah antrian

```
public int getJumlahAntrian() {
    return size;
}
```

  Cetak yang sudah menyelesaikan proses KRS

```
public void sudahProses() {
    System.out.println("Mahasiswa yang telah dilayani sebanyak: " +sudahProses + " maha
}
```

- Jumlah antrian maximal 10, jumlah yang ditangani masing-masing DPA 30 mahasiswa, cetak jumlah mahasiswa yang belum melakukan proses KRS.

```
public void belumProses() {
    System.out.println("Mahasiswa yang belum dilayani sebanyak: " + maxMhs + " mahasis
}
```

Gambarkan **Diagram Class** untuk antriannya. Implementasikan semua method menggunakan menu pilihan pada fungsi main

Mahasiswa
nim: String nama: String prodi: String kelas: String
Mahasiswa(nim: String, nama: String, prodi: String, kelas: String) tampilkanData(): void

AntrianKRS
data: Mahasiswa[] front: int rear: int size: int max: int sudahProses: int maxMhs: int
AntrianKRS(max: int) IsFull(): boolean IsEmpty(): boolean tambahAntrian(mhs: Mahasiswa): void layaniMahasiswa(): void tampilkanSemua(): void tampil2Terdepan(): void getJumlahAntrian(): int sudahProses(): void

belumProses(): void lihatAkhir(): void clear(): void

Menu pilihan:

```
do {
    System.out.println(x:"\n=== Menu Antrian Layanan Akademik ===");
    System.out.println(x:"1. Tambah Mahasiswa ke Antrian");
    System.out.println(x:"2. Layani Mahasiswa");
    System.out.println(x:"3. Lihat 2 Antrian Terdepan");
    System.out.println(x:"4. Lihat Semua Antrian");
    System.out.println(x:"5. Lihat Antrian paling belakang");
    System.out.println(x:"6. Jumlah Mahasiswa dalam Antrian");
    System.out.println(x:"7. Jumlah yang belum ditangani");
    System.out.println(x:"8. Jumlah yang telah ditangani");
    System.out.println(x:"9. Kosongkan Antrian");
    System.out.println(x:"0. Keluar");
    System.out.print(s:"Pilih menu: ");
    pilihan = sc.nextInt(); sc.nextLine();
```

Kondisi sesuai pilihan

```
    switch (pilihan) {
        case 1:
            System.out.print(s:"NIM   : ");
            String nim = sc.nextLine();
            System.out.print(s:"Nama  : ");
            String nama =sc.nextLine();
            System.out.print(s:"Prodi : ");
            String prodi = sc.nextLine();
            System.out.print(s:"Kelas : ");
            String kelas = sc.nextLine();
            Mahasiswa mhs = new Mahasiswa(nim, nama, prodi, kelas);
            antrian.tambahAntrian(mhs);
            break;
        case 2:
            antrian.layaniMahasiswa();
            break;
        case 3:
            antrian.tampil2Terdepan();
            break;
        case 4:
            antrian.tampilkanSemua();;
            break;
        case 5:
            antrian.lihatAkhir();
            break;
        case 6:
            System.out.println("Jumlah dalam Antrian " + antrian.getJumlahAntrian());
            break;
        case 7:
            antrian.belumProses();
            break;
        case 8:
            antrian.sudahProses();
            break;
        case 9:
            antrian.clear();
            break;
        case 0:
            System.out.println(x:"Terima kasih");
            break;
        default:
            System.out.println(x:"Pilihan tidak valid");
            break;
    }
} while (pilihan != 0);
```