Laporan Hasil Pratikum

Algoritma Dan Struktur Data

Jobsheet 10

Nama : Zacky Rio Orlando

NIM : 244107020086

Kelas : 1E

Program Studi D-IV Teknik Informatika

Jurusan Teknologi Informasi

Praktikum

2025

2.1 Percobaan 1 : Operasi Dasar Queue

Class Queue

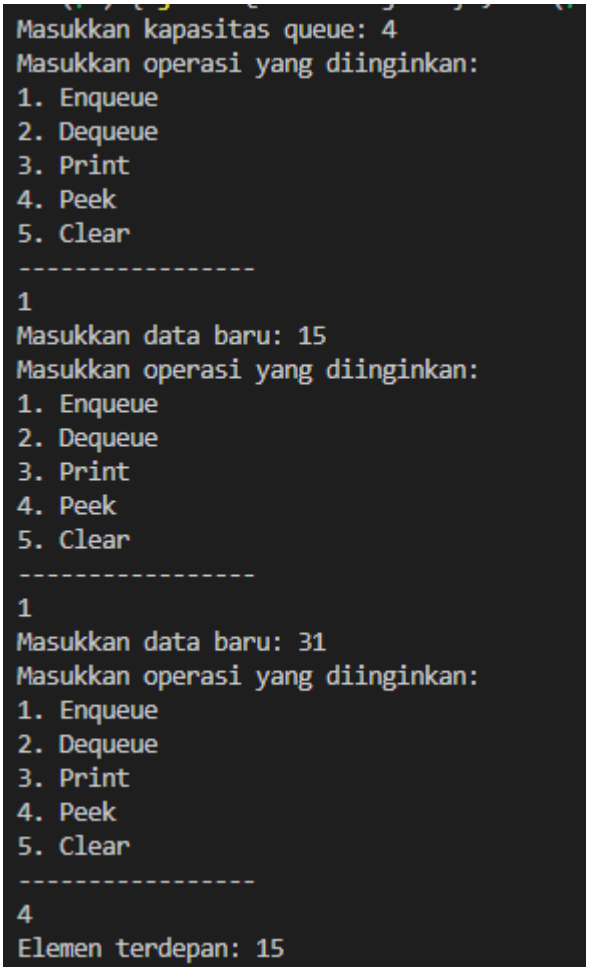
```
public class Queue {
    int[] data;
    int front;
    int rear;
    int size;
    int max;

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

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

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

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

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

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

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

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

Class QueueMain

```
import java.util.Scanner;
public class QueueMain {
    public static void menu() {
        System.out.println("Masukkan operasi yang diinginkan: ");
        System.out.println("1. Enqueue");
        System.out.println("2. Dequeue");
        System.out.println("3. Print");
        System.out.println("4. Peek");
        System.out.println("5. Clear");
        System.out.println("-----------------");
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Masukkan kapasitas queue: ");
        int n = sc.nextInt();
        Queue Q = new Queue(n);
        int pilih;


        do {
            menu();
            pilih = sc.nextInt();
            switch (pilih) {
                case 1:
                    System.out.print("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);
    }
}
```

Hasil Output dari kode program diatas



2.1.3. Pertanyaan

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

Jawaban :

Pada konstruktor, nilai front dan rear dibuat -1 karena saat queue baru dibuat, belum ada data yang dimasukkan, jadi belum ada posisi yang ditunjuk di dalam array. Nilai -1 ini semacam tanda bahwa antriannya masih kosong. Sedangkan size diberi nilai 0 karena memang belum ada elemen yang masuk ke queue. Nanti setelah data mulai dimasukkan, nilai front, rear, dan size akan berubah menyesuaikan kondisi queue tersebut.

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

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

Jawaban :

Jika posisi belakang antrian (rear) sudah sampai ujung, kode ini bikin posisinya balik lagi ke awal supaya bisa terus masuk data baru. Jadi antriannya muter terus kayak lingkaran, nggak cuma berhenti di ujung, jadi tempat yang kosong di depan bisa dipakai lagi tanpa sia-sia.

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

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

Jawaban :

jika front sudah berada di posisi terakhir, agar proses mengeluarkan data berikutnya tetap lancar, posisi front diubah ke 0 lagi. Dengan begitu, antrian berjalan melingkar dan bisa terus digunakan tanpa kehilangan tempat kosong yang ada di awal.

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

Jawaban :

Variabel i dimulai dari front karena front itu menunjukkan posisi data paling depan yang harus diproses atau dilihat duluan. Jadi saat mau menampilkan isi antrian, kita mulai dari depan supaya urutan datanya benar sesuai antrian sebenarnya. Kalau mulai dari 0, bisa saja kita lihat data yang sudah tidak di antrian lagi atau data yang urutannya salah karena antrian ini bersifat melingkar dan posisi datanya tidak selalu di awal. Maka dari itu, mulai dari front supaya tampilannya sesuai dengan urutan antrian yang sebenarnya.

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

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

Jawaban :
digunakan supaya nilai i maju ke posisi berikutnya dalam array secara melingkar. Karena queue ini berbentuk melingkar, kalau i sudah sampai di posisi terakhir, maka dengan operasi (i + 1) % max, i akan kembali ke 0, bukan melewati batas array. Jadi, kode ini memastikan i terus bergerak di antara posisi 0 sampai max-1 tanpa keluar dari batas, sehingga data antrian bisa diakses dengan benar dan urut.

6. Tunjukkan potongan kode program yang merupakan queue overflow!

Jawaban :

```
if (IsFull()) {

    System.out.println("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!

Jawaban :

Pada method Enqueue

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

Pada method Dequeue

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

Hasil Outputnya

- Queue Overflow

```
Masukkan kapasitas queue: 2
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
----------------
1
Masukkan data baru: 15
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
----------------
1
Masukkan data baru: 31
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
----------------
1
Masukkan data baru: 40
Queue sudah penuh
```

- Queue Underflow

```
Masukkan kapasitas queue: 2
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
----------------
2
Queue masih kosong
```

2.2. Percobaan 2 : Antrian Layanan Akademik

Class Mahasiswa

```
public class Mahasiswa {
    String nim;
    String nama;
    String prodi;
    String kelas;

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

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

Class AntrianLayanan

```
public class AntrianLayanan {
    Mahasiswa[] data;
    int front;
    int rear;
    int size;
    int max;


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

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

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

    public void tambahAntrian(Mahasiswa mhs) {
        if (IsFull()) {
            System.out.println("Antrian penuh, tidak dapat menambah
mahasiswa.");
            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("Antrian kosong.");
            return null;
        }
        Mahasiswa mhs = data[front];
        front = (front + 1) % max;
        size--;
        return mhs;
    }

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

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

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

Class LayananAkademikSiakad

```
import java.util.Scanner;
public class LayananAkademikSIAKAD {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        AntrianLayanan antrian = new AntrianLayanan(5);
        int pilihan;

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

            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();
                    System.out.print("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.println("Melayani mahasiswa: ");
                        dilayani.tampilkanData();
                    }
                    break;
                case 3:
                    antrian.lihatTerdepan();
                    break;
                case 4:
                    antrian.tampilkanSemua();
                    break;
                case 5:
                    System.out.println("Jumlah dalam antrian: " +
antrian.getJumlahAntrian());
                    break;
                case 0:
                    System.out.println("Terima kasih.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid.");
            }
        } while (pilihan != 0);
    }
}
```

Hasil Output kode program diatas

```
=== 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  : Aldi
Prodi  : TI
Kelas  : 1A
Aldi 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  : Bobi
Prodi  : TI
Kelas  : 1G
Bobi 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 - Aldi - TI - 1A
2.
124 - Bobi - TI - 1G

=== 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 - Aldi - TI - 1A
```

```
=== 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 - Bobi - TI - 1G

=== 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.
```

2.2.3 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!

Jawaban :

Class Mahasiswa

```
public class Mahasiswa {
    String nim;
    String nama;
    String prodi;
    String kelas;

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

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

Class AntrianLayanan

```
public class AntrianLayanan {
    Mahasiswa[] data;
    int front;
    int rear;
    int size;
    int max;


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

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

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

    public void tambahAntrian(Mahasiswa mhs) {
        if (IsFull()) {
            System.out.println("Antrian penuh, tidak dapat menambah
mahasiswa.");
            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("Antrian kosong.");
            return null;
        }
        Mahasiswa mhs = data[front];
        front = (front + 1) % max;
        size--;
        return mhs;
    }

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

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

    public int getJumlahAntrian() {
        return size;
    }

    public void lihatAkhir() {
        if (IsEmpty()) {
            System.out.println("Antrian kosong.");
        } else {
            System.out.println("Mahasiswa di antrian paling belakang: ");
            System.out.println("NIM - NAMA - PRODI - KELAS");
            data[rear].tampilkanData();
        }
    }
}
```

Class LayananAkademikSIAKAD

```
import java.util.Scanner;
public class LayananAkademikSIAKAD {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        AntrianLayanan antrian = new AntrianLayanan(5);
        int pilihan;

        do {
            System.out.println("\n=== Menu Antrian Layanan Akademik ===");
            System.out.println("1. Tambah Mahasiswa ke Antrian");
            System.out.println("2. Layani Mahasiswa");
            System.out.println("3. Lihat Mahasiswa Terdepan");
            System.out.println("4. Lihat Semua Antrian");
            System.out.println("5. Jumlah Mahasiswa dalam Antrian");
            System.out.println("6. Cek Antrian paling belakang");
            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();
                    System.out.print("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.println("Melayani mahasiswa: ");
                        dilayani.tampilkanData();
                    }
                    break;
                case 3:
                    antrian.lihatTerdepan();
                    break;
                case 4:
                    antrian.tampilkanSemua();
                    break;
                case 5:
                    System.out.println("Jumlah dalam antrian: " +
antrian.getJumlahAntrian());
                    break;
                case 6:
                    antrian.lihatAkhir();
                    break;
                case 0:
                    System.out.println("Terima kasih.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid.");
            }
        } while (pilihan != 0);
    }
}
```

Hasil Outputnya

```
=== 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: 1
NIM   : 123
Nama  : Aldi
Prodi  : TI
Kelas  : 1A
Aldi 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
6. Cek Antrian paling belakang
0. Keluar
Pilih menu: 1
NIM   : 124
Nama  : Bobi
Prodi  : TI
Kelas  : 1G
Bobi 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
6. Cek Antrian paling belakang
0. Keluar
Pilih menu: 1
NIM   : 125
Nama  : Zacky
Prodi  : TI
Kelas  : 1E
Zacky 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
6. Cek Antrian paling belakang
0. Keluar
Pilih menu: 3
Mahasiswa terdepan:
NIM - NAMA - PRODI - KELAS
123 - Aldi - TI - 1A

=== 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 di antrian paling belakang:
NIM - NAMA - PRODI - KELAS
125 - Zacky - TI - 1E
```

2.3 Tugas
Waktu : 120 Menit
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.
• Menambahkan antrian, Memanggil antrian untuk proses KRS – setiap 1x panggilan terdiri dari 2 mahasiswa (pada antrian no 1 dan 2)
• Menampilkan semua antrian, Menampilkan 2 antrian terdepan, Menampilkan antrian paling akhir.
• Cetak jumlah antrian, Cetak jumlah yang sudah melakukan proses KRS
• Jumlah antrian maximal 10, jumlah yang ditangani masing-masing DPA 30 mahasiswa, cetak jumlahmahasiswa yang belum melakukan proses KRS.
Gambarkan **Diagram Class** untuk antriannya. Implementasikan semua method menggunakan menu pilihan pada fungsi **main**.

Jawaban :

Class Mahasiswa

```
public class Mahasiswa {
    String nim;
    String nama;
    String prodi;
    String kelas;

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

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

Class AntrianLayanan

```
public class AntrianLayanan {
    Mahasiswa[] data;
    int front;
    int rear;
    int size;
    int max;
    int sudahDiproses;

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

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

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

    public void tambahAntrian(Mahasiswa mhs) {
        if (IsFull()) {
            System.out.println("Antrian penuh!");
        } else {
            rear = (rear + 1) % max;
            data[rear] = mhs;
            size++;
            System.out.println("Mahasiswa berhasil ditambahkan ke
antrian.");
        }
    }

    public void panggilKRS() {
        if (size < 2) {
            System.out.println("Jumlah antrian kurang dari 2.");
        } else {
            System.out.println("Mahasiswa dipanggil untuk proses KRS:");
            for (int i = 0; i < 2; i++) {
                System.out.print((i + 1) + ". ");
                data[front].tampilkanData();
                front = (front + 1) % max;
                size--;
                sudahDiproses++;
            }
        }
    }
```

```
    public void lihatTerdepan() {
        if (IsEmpty()) {
            System.out.println("Antrian kosong.");
        } else {
            System.out.println("2 Antrian terdepan:");
            for (int i = 0; i < Math.min(2, size); i++) {
                int index = (front + i) % max;
                System.out.print((i + 1) + ". ");
                data[index].tampilkanData();
            }
        }
    }

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

    public void tampilkanSemua() {
        if (IsEmpty()) {
            System.out.println("Antrian kosong.");
        } else {
            System.out.println("Seluruh Mahasiswa dalam Antrian:");
            for (int i = 0; i < size; i++) {
                int index = (front + i) % max;
                System.out.print((i + 1) + ". ");
                data[index].tampilkanData();
            }
        }
    }

    public void clear() {
        front = 0;
        rear = -1;
        size = 0;
        sudahDiproses = 0;
        System.out.println("Antrian dikosongkan.");
    }

    public int getJumlahAntrian() {
        return size;
    }

    public int getSudahDiproses() {
        return sudahDiproses;
    }

    public int getBelumDiproses() {
        return 30 - sudahDiproses;
    }
}
```

Class AntrianKRSMain

```
import java.util.Scanner;
public class AntrianKRSMain {
    public static void menu() {
        System.out.println("\n=== MENU ANTRIAN KRS ===");
        System.out.println("1. Tambah Mahasiswa ke Antrian");
        System.out.println("2. Panggil 2 Mahasiswa untuk Proses KRS");
        System.out.println("3. Lihat 2 Antrian Terdepan");
        System.out.println("4. Lihat Antrian Paling Belakang");
        System.out.println("5. Tampilkan Semua Antrian");
        System.out.println("6. Cek Jumlah Antrian");
        System.out.println("7. Cek Jumlah Mahasiswa yang Sudah Proses KRS");
        System.out.println("8. Cek Jumlah Mahasiswa yang Belum Proses KRS");
        System.out.println("9. Kosongkan Antrian");
        System.out.println("0. Keluar");
        System.out.print("Pilih menu: ");
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        AntrianLayanan antrian = new AntrianLayanan(10);
        int pilihan;

        do {
            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();
                    System.out.print("Kelas : ");
                    String kelas = sc.nextLine();
                    Mahasiswa mhs = new Mahasiswa(nim, nama, prodi, kelas);
                    antrian.tambahAntrian(mhs);
                    break;
                case 2:
                    antrian.panggilKRS();
                    break;
                case 3:
                    antrian.lihatTerdepan();
                    break;
                case 4:
                    antrian.lihatAkhir();
                    break;
                case 5:
                    antrian.tampilkanSemua();
                    break;
                case 6:
                    System.out.println("Jumlah antrian saat ini: " +
antrian.getJumlahAntrian());
                    break;
                case 7:
                    System.out.println("Jumlah mahasiswa yang sudah diproses KRS: " +
antrian.getSudahDiproses());
                    break;
                case 8:
                    System.out.println("Jumlah mahasiswa yang belum diproses KRS: " +
antrian.getBelumDiproses());
                    break;
                case 9:
                    antrian.clear();
                    break;
                case 0:
                    System.out.println("Terima kasih.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid.");
            }
        } while (pilihan != 0);
    }
}
```

Hasil Output dari kode program diatas

```
=== MENU ANTRIAN KRS ===
1. Tambah Mahasiswa ke Antrian
2. Panggil 2 Mahasiswa untuk Proses KRS
3. Lihat 2 Antrian Terdepan
4. Lihat Antrian Paling Belakang
5. Tampilkan Semua Antrian
6. Cek Jumlah Antrian
7. Cek Jumlah Mahasiswa yang Sudah Proses KRS
8. Cek Jumlah Mahasiswa yang Belum Proses KRS
9. Kosongkan Antrian
0. Keluar
Pilih menu: 1
NIM   : 123
Nama  : Aldi
Prodi : TI
Kelas : 1A
Mahasiswa berhasil ditambahkan ke antrian.

=== MENU ANTRIAN KRS ===
1. Tambah Mahasiswa ke Antrian
2. Panggil 2 Mahasiswa untuk Proses KRS
3. Lihat 2 Antrian Terdepan
4. Lihat Antrian Paling Belakang
5. Tampilkan Semua Antrian
6. Cek Jumlah Antrian
7. Cek Jumlah Mahasiswa yang Sudah Proses KRS
8. Cek Jumlah Mahasiswa yang Belum Proses KRS
9. Kosongkan Antrian
0. Keluar
Pilih menu: 1
NIM   : 124
Nama  : Bobi
Prodi : TI
Kelas : 1G
Mahasiswa berhasil ditambahkan ke antrian.
```

```
=== MENU ANTRIAN KRS ===
1. Tambah Mahasiswa ke Antrian
2. Panggil 2 Mahasiswa untuk Proses KRS
3. Lihat 2 Antrian Terdepan
4. Lihat Antrian Paling Belakang
5. Tampilkan Semua Antrian
6. Cek Jumlah Antrian
7. Cek Jumlah Mahasiswa yang Sudah Proses KRS
8. Cek Jumlah Mahasiswa yang Belum Proses KRS
9. Kosongkan Antrian
0. Keluar
Pilih menu: 1
NIM   : 125
Nama  : Zacky
Prodi : TI
Kelas : 1E
Mahasiswa berhasil ditambahkan ke antrian.

=== MENU ANTRIAN KRS ===
1. Tambah Mahasiswa ke Antrian
2. Panggil 2 Mahasiswa untuk Proses KRS
3. Lihat 2 Antrian Terdepan
4. Lihat Antrian Paling Belakang
5. Tampilkan Semua Antrian
6. Cek Jumlah Antrian
7. Cek Jumlah Mahasiswa yang Sudah Proses KRS
8. Cek Jumlah Mahasiswa yang Belum Proses KRS
9. Kosongkan Antrian
0. Keluar
Pilih menu: 5
Seluruh Mahasiswa dalam Antrian:
1. 123 - Aldi - TI - 1A
2. 124 - Bobi - TI - 1G
3. 125 - Zacky - TI - 1E
```

```
=== MENU ANTRIAN KRS ===
1. Tambah Mahasiswa ke Antrian
2. Panggil 2 Mahasiswa untuk Proses KRS
3. Lihat 2 Antrian Terdepan
4. Lihat Antrian Paling Belakang
5. Tampilkan Semua Antrian
6. Cek Jumlah Antrian
7. Cek Jumlah Mahasiswa yang Sudah Proses KRS
8. Cek Jumlah Mahasiswa yang Belum Proses KRS
9. Kosongkan Antrian
0. Keluar
Pilih menu: 2
Mahasiswa dipanggil untuk proses KRS:
1. 123 - Aldi - TI - 1A
2. 124 - Bobi - TI - 1G

=== MENU ANTRIAN KRS ===
1. Tambah Mahasiswa ke Antrian
2. Panggil 2 Mahasiswa untuk Proses KRS
3. Lihat 2 Antrian Terdepan
4. Lihat Antrian Paling Belakang
5. Tampilkan Semua Antrian
6. Cek Jumlah Antrian
7. Cek Jumlah Mahasiswa yang Sudah Proses KRS
8. Cek Jumlah Mahasiswa yang Belum Proses KRS
9. Kosongkan Antrian
0. Keluar
Pilih menu: 5
Seluruh Mahasiswa dalam Antrian:
1. 125 - Zacky - TI - 1E
```

```
=== MENU ANTRIAN KRS ===
1. Tambah Mahasiswa ke Antrian
2. Panggil 2 Mahasiswa untuk Proses KRS
3. Lihat 2 Antrian Terdepan
4. Lihat Antrian Paling Belakang
5. Tampilkan Semua Antrian
6. Cek Jumlah Antrian
7. Cek Jumlah Mahasiswa yang Sudah Proses KRS
8. Cek Jumlah Mahasiswa yang Belum Proses KRS
9. Kosongkan Antrian
0. Keluar
Pilih menu: 6
Jumlah antrian saat ini: 1

=== MENU ANTRIAN KRS ===
1. Tambah Mahasiswa ke Antrian
2. Panggil 2 Mahasiswa untuk Proses KRS
3. Lihat 2 Antrian Terdepan
4. Lihat Antrian Paling Belakang
5. Tampilkan Semua Antrian
6. Cek Jumlah Antrian
7. Cek Jumlah Mahasiswa yang Sudah Proses KRS
8. Cek Jumlah Mahasiswa yang Belum Proses KRS
9. Kosongkan Antrian
0. Keluar
Pilih menu: 7
Jumlah mahasiswa yang sudah diproses KRS: 2
```

```
=== MENU ANTRIAN KRS ===
1. Tambah Mahasiswa ke Antrian
2. Panggil 2 Mahasiswa untuk Proses KRS
3. Lihat 2 Antrian Terdepan
4. Lihat Antrian Paling Belakang
5. Tampilkan Semua Antrian
6. Cek Jumlah Antrian
7. Cek Jumlah Mahasiswa yang Sudah Proses KRS
8. Cek Jumlah Mahasiswa yang Belum Proses KRS
9. Kosongkan Antrian
0. Keluar
Pilih menu: 8
Jumlah mahasiswa yang belum diproses KRS: 28

=== MENU ANTRIAN KRS ===
1. Tambah Mahasiswa ke Antrian
2. Panggil 2 Mahasiswa untuk Proses KRS
3. Lihat 2 Antrian Terdepan
4. Lihat Antrian Paling Belakang
5. Tampilkan Semua Antrian
6. Cek Jumlah Antrian
7. Cek Jumlah Mahasiswa yang Sudah Proses KRS
8. Cek Jumlah Mahasiswa yang Belum Proses KRS
9. Kosongkan Antrian
0. Keluar
Pilih menu: 0
Terima kasih.
```

Diagram Classnya

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

AntrianLayanan
data : Mahasiswa[] front: int rear : int size : int max: int sudahDiproses: int
AntrianLayanan(n : int) IsEmpty(): boolean IsFull(): boolean TambahAntrian(m : Mahasiswa) panggilKRS(): void lihatTerdepan(): void lihatAkhir(): void tampilkanSemua(): void clear(): void getJumlahAntrian(): int getSudahDiproses(): int getBelumDiproses(): int

AntrianKRSMain
main(args: String[]): void menu(): void