LAPORAN HASIL PRAKTIKUM

ALGORITMA DAN STRUKTUR DASAR

JOBSHEET 10

NAMA : **DANDIKA MARTHA C.**

NIM : **244107020092**

KELAS : **1E**

Program Studi Teknik Informatika

Jurusan Teknologi Informasi

Praktikum

2025

Percobaan 1

Code

- Queue07.java

```
package jobsheet_10;

public class Queue07 {
    int[] data;
    int front;
    int rear;
    int size;
    int max;
    public Queue07(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.print(data[i] + " ");
                i = (i + 1)  % max;
            }
            System.out.println(data[i] + " ");
            System.out.println("Jumlah elemen = " + size);
        }
    }
```

- Lanjutan Queue07.java

```
    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;
    }
}
```

- QueueMain07.java

```
package jobsheet_10;
import java.util.Scanner;
public class QueueMain07 {
    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();

        Queue07 Q = new Queue07(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);
        sc.close();
    }
}
```

- Hasil

```
Masukkan kapasitas queue: 4
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
------------------
4
Elemen terdepan: 15
```

Pertanyaan

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

Jawab : Front di gunakan sebagai elemen pertama dan **rear** di gunakan sebagai elemen terakhir yang di tambahkan, tujuan penulisan front dan rear == -1 adalah **sebagai penanda bahwa queue tersebut kosong**, dan kenapa **size == 0** karena belum adanya data yang disimpan, dan **data pertama kali akan disimpan di data[size=0]**

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

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

Jawab : Kode tersebut di gunakan untuk mengisi kembali **slot index di depan yang sudah di Dequeue** sehingga semua **slot dapat di gunakan lagi**. Rear menandakan posisi elemen terakhir sehingga ketika **rear mencapai slot terakhir akan di pindahkan ke rear == 0** lagi agar slot terdepan bisa di isi kembali.

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

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

Jawab : Kode tersebut di gunakan agar **front bisa kembali ke awal index setelah di index hingga nilai terakhir**, hal tersebut bertujuan **agar nilai tetap bisa diisi namun tanpa membuat nilai front keluar dari index.** Front menandakan nilai terdepan yang di isi dan **ketika di Dequeue front akan terus bergeser hingga ke index akhir**, ketika **front == max -1** maka front akan kembali ke index 0 dan **bisa di Enqueue untuk mengsisi nilai kembali.**

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

Jawab : Karena setelah terjadi banyak Enqueue dan Dequeue **tidak selalu indeks 0 berisi nilai bisa saja NULL atau semacamnya**. Oleh karena itu, maka **penggunaan int i= from bertujuan agar nilai yang di print memiliki nilai atau bukan index kosong.**

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

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

Jawab : Kode **(i + 1)** di gunakan untuk menambah nilai i yang akan di print, lalu nilai **i = (i + 1) % max** adalah ketika **nilai i == max maka akan kembali ke index 0 (contoh max == 4, maka 4 % 4 == 0)** hal ini **bermanfaat jika print front dimulai dari index != 0 namun rear berada di nilai == 0 atau selanjutanya.**

6. Tunjukkan potongan kode program yang merupakan queue overflow!

Jawab :

```
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!

Jawab :

Tambahkan kode pada case 1 di class QueueMain07.java

```
if (Q.IsFull()) {
    System.out.println("Program Penuh, Overflow terdeteksi. Program
dihentikan");
    pilih = -1;
    break;
}
```

Tambahkan kode pada case 2 di class QueueMain07.java

```
if (Q.IsEmpty()) {
    System.out.println("Program Kosong, Underflow terdeteksi.
Program dihentikan");
    pilih = -1;
    break;
}
```

- Hasil

*** Overflow**

```
Masukkan kapasitas queue: 1
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
Program Penuh, Queue Overflow terdeteksi. Program dihentikan
PS G:\DATA DIKO\POLINEMA TI\SEMESTER 2\PRAK ASD\praktikumASD>
```

*** Underflow**

```
Masukkan kapasitas queue: 1
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----------------
2
Program Kosong, Queue Underflow terdeteksi. Program dihentikan
PS G:\DATA DIKO\POLINEMA TI\SEMESTER 2\PRAK ASD\praktikumASD>
```

Percobaan 2

Code

- Mahasiswa07.java

```
package jobsheet_10;

public class Mahasiswa07 {
    String nim;
    String nama;
    String prodi;
    String kelas;
    public Mahasiswa07() {
    }
    public Mahasiswa07(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);
    }
}
```

- AntrianLayanan07.java

```
package jobsheet_10;

public class AntrianLayanan07 {
    Mahasiswa07[] data;
    int front;
    int rear;
    int size;
    int max;
    public AntrianLayanan07(int max) {
        this.max = max;
        this.data = new Mahasiswa07[max];
        this.front = 0;
        this.rear = -1;
        this.size = 0;
    }
    public boolean IsEmpty() {
        if (size == 0) {
            return true;
        } else {
            return false;
        }
    }
    public boolean IsFull() {
        if (size == max) {
            return true;
        } else {
            return false;
        }
    }
```

- Lanjutan AntrianLayanan07.java

```
    public void tambahAntrian(Mahasiswa07 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 Mahasiswa07 layaniMahasiswa() {
        if (IsEmpty()) {
            System.out.println("Antrian kosong.");
            return null;
        }
        Mahasiswa07 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.print((i + 1) + ". ");
            data[index].tampilkanData();
        }
    }
    public int getJumlahAntrian() {
        return size;
    }
}
```

- LayananAkademikSIAKAD07.java

```
package jobsheet_10;
import java.util.Scanner;
public class LayananAkademikSIAKAD07 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        AntrianLayanan07 antrian = new AntrianLayanan07(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.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();
                    Mahasiswa07 mhs = new Mahasiswa07(nim, nama, prodi,
kelas);
                    antrian.tambahAntrian(mhs);
                    break;
                case 2:
                    Mahasiswa07 dilayani = antrian.layaniMahasiswa();
                    if (dilayani != null) {
                        System.out.print("Melayani mahasiswa: ");
                        dilayani.tampilkanData();
                    }
                    break;
                case 3:
                    antrian.lihatTerdepan();
                    break;
                case 4:
                    antrian.tampilkanSemua();
                    break;
```

- Lanjutan LayananAkademikSIAKAD07.java

```
                case 5:
                    System.out.println("Jumlah dalam antrian: "
+antrian.getJumlahAntrian());
                    break;
                case 0:
                    System.out.println("Terimakasih.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid.");
            }
        } while (pilihan != 0);
        sc.close();
    }
}
```

- Hasil

```
=== 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
Terimakasih.
PS G:\DATA DIKO\POLINEMA TI\SEMESTER 2\PRAK ASD\praktikumASD>
```

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!

Jawab :

- Tambahkan method pada class AntrianLayanan07.java

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

- Tambahkan menu 6 pada class LayananAkademikSIAKAD07.java

```
System.out.println("6. Cek Antrian Paling Belakang");
```

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

- Hasil

```
=== 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  : Caca
Prodi : TI
Kelas : 1H
Caca 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: 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: 6
Mahasiswa terakhir:
NIM - NAMA - PRODI - KELAS
125 - Caca - TI - 1H
```

Tugas

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 jumlah mahasiswa yang belum melakukan proses KRS.

Jawab :

- Code

- Class Mahasiswa07.java

```
package jobsheet_10;

public class Mahasiswa07 {
    String nim;
    String nama;
    String prodi;
    String kelas;
    public Mahasiswa07() {
    }
    public Mahasiswa07(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);
    }
}
```

- AntrianKRS07.java

```
package jobsheet_10;

public class AntrianKRS07 {
    Mahasiswa07[] data;
    int front;
    int rear;
    int size;
    int max;
    int krs;
    int maxDPA;
    public AntrianKRS07(int max, int maxDPA) {
        this.max = max;
        this.maxDPA = maxDPA;
        this.data = new Mahasiswa07[max];
        this.front = 0;
        this.rear = -1;
        this.size = 0;
        this.krs = 0;
    }
    public boolean IsEmpty() {
        if (size == 0) {
            return true;
        } else {
            return false;
        }
    }
    public boolean IsFull() {
        if (size == max) {
            return true;
        } else {
            return false;
        }
    }
    public void clear() {
        if (!IsEmpty()) {
            front = rear = -1;
            size = 0;
            krs = 0;
            System.out.println("Antrian berhasil di kosongkan");
        } else {
            System.out.println("Antrian belum ada");
        }
    }
```

- Lanjutan AntrianKRS07.java

```
    public void tambahAntrian(Mahasiswa07 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 Mahasiswa07[] layaniMahasiswa() {
        if (IsEmpty()) {
            System.out.println("Antrian kosong.");
            return null;
        } else if (size < 2) {
            System.out.println("Minimal harus ada 2 mahasiswa dalam
antrian untuk diproses.");
            return null;
        } else if (krs + 2 > maxDPA) {
            System.out.println("Kuota KRS DPA tidak mencukupi untuk
memproses 2 mahasiswa.");
            return null;
        } else {
            Mahasiswa07[] diproses = new Mahasiswa07[2];
            for (int i = 0; i < 2; i++) {
                diproses[i] = data[front];
                front = (front + 1) % max;
                size--;
                krs++;
            }
            return diproses;
        }
    }
    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.print((i + 1) + ". ");
            data[index].tampilkanData();
        }
    }
```

- Lanjutan AntrianKRS07.java

```
    public void lihatTerdepan() {
        if (IsEmpty()) {
            System.out.println("Antrian kosong.");
        } else if (size < 2){
            System.out.println("Minimal harus ada 2 mahasiswa dalam
antrian untuk diproses.");
        } else {
            System.out.println("2 Mahasiswa terdepan:");
            System.out.println("NIM - NAMA - PRODI - KELAS");
            for (int i = 0; i < 2; i++) {
                int index = (front + i) % max;
                data[index].tampilkanData();
            }
        }
    }
    public void lihatAkhir() {
        if (IsEmpty()) {
            System.out.println("Antrian kosong.");
        } else {
            System.out.println("Mahasiswa terakhir: ");
            System.out.println("NIM - NAMA - PRODI - KELAS");
            data[rear].tampilkanData();
        }
    }
    public int getJumlahAntrian() {
        return size;
    }
    public int getJumlahKRS() {
        return krs;
    }
    public int getSisaKRS() {
        return maxDPA - krs;
    }
}
```

- LayananKRS07.java

```
package jobsheet_10;
import java.util.Scanner;
public class LayananKRS07 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        AntrianKRS07 krs = new AntrianKRS07(10, 30);
        int pilihan;

        do {
            System.out.println("\n=== Menu Antrian Layanan Akademik
===");
            System.out.println("1. Cek Antrian Kosong");
            System.out.println("2. Cek Antrian Penuh");
            System.out.println("3. Mengosongkan Antrian");
            System.out.println("4. Menambah Antrian");
            System.out.println("5. Memproses 2 Antrian KRS");
            System.out.println("6. Menampilkan Semua Antrian");
            System.out.println("7. Menampilkan 2 Antrian Terdepan");
            System.out.println("8. Menampilkan Antrian Terakhir");
            System.out.println("9. Cetak Jumlah Antrian");
            System.out.println("10. Cetak Jumlah Yang Sudah Melakukan
KRS");
            System.out.println("11. Cetak Jumlah Yang Belum Melakukan
KRS");
            System.out.println("0. Keluar");
            System.out.print("Pilih menu: ");
            pilihan = sc.nextInt(); sc.nextLine();
            switch (pilihan) {
                case 1:
                    if (krs.IsEmpty()) {
                        System.out.println("Antrian kosong.");
                    } else {
                        System.out.println("Antrian tidak kosong.");
                    }
                    break;
                case 2:
                    if (krs.IsFull()) {
                        System.out.println("Antrian sudah penuh.");
                    } else {
                        System.out.println("Antrian belum penuh.");
                    }
                    break;
                case 3:
                    krs.clear();
                    break;
                case 4:
                    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();
                    Mahasiswa07 mhs = new Mahasiswa07(nim, nama, prodi,
kelas);
                    krs.tambahAntrian(mhs);
                    break;
```

- Lanjutan LayananKRS07.java

```
                case 5:
                    Mahasiswa07[] dilayani = krs.layaniMahasiswa();
                    if (dilayani != null) {
                        System.out.println("Memproses mahasiswa:");
                        for (Mahasiswa07 m : dilayani) {
                            m.tampilkanData();
                        }
                    }
                    break;
                case 6:
                    krs.tampilkanSemua();
                    break;
                case 7:
                    krs.lihatTerdepan();
                    break;
                case 8:
                    krs.lihatAkhir();
                    break;
                case 9:
                    System.out.println("Jumlah Mahasiswa dalam antrian:
" + krs.getJumlahAntrian());
                    break;
                case 10:
                    System.out.println("Jumlah Mahasiswa yang sudah
melakukan KRS: " + krs.getJumlahKRS());
                    break;
                case 11:
                    System.out.println("Jumlah Mahasiswa yang belum
melakukan KRS: " + krs.getSisaKRS());
                    break;
                case 0:
                    System.out.println("Terimakasih.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid.");
            }
        } while (pilihan != 0);
        sc.close();
    }
}
```

- Hasil

- 1. Cek Antrian Kosong

```
=== Menu Antrian Layanan Akademik ===
1. Cek Antrian Kosong
2. Cek Antrian Penuh
3. Mengosongkan Antrian
4. Menambah Antrian
5. Memproses 2 Antrian KRS
6. Menampilkan Semua Antrian
7. Menampilkan 2 Antrian Terdepan
8. Menampilkan Antrian Terakhir
9. Cetak Jumlah Antrian
10. Cetak Jumlah Yang Sudah Melakukan KRS
11. Cetak Jumlah Yang Belum Melakukan KRS
0. Keluar
Pilih menu: 1
Antrian kosong.
```

- 2. Cek Antrian Penuh

```
Pilih menu: 2
Antrian belum penuh.
```

- 3. Mengosongkan Antrian

```
Pilih menu: 3
Antrian berhasil di kosongkan
```

- 4. Menambah Antrian

```
Pilih menu: 4
NIM   : 123
Nama  : Adi
Prodi : TI
Kelas : 1C
Adi berhasil masuk ke antrian.
Pilih menu: 5
Memproses mahasiswa:
123 - Adi - TI - 1C
124 - Cici - TI - 1A
```

- 5. Memproses 2 Antrian KRS

```
Pilih menu: 5
Memproses mahasiswa:
123 - Adi - TI - 1C
124 - Cici - TI - 1A
```

- 6. Menampilkan Semua Antrian

```
Pilih menu: 6
Daftar Mahasiswa dalam Antrian:
NIM - NAMA - PRODI - KELAS
1. 123 - Adi - TI - 1C
2. 124 - Cici - TI - 1A
```

- 7. Menampilkan 2 Antrian Terdepan

```
Pilih menu: 7
2 Mahasiswa terdepan:
NIM - NAMA - PRODI - KELAS
123 - Adi - TI - 1C
124 - Cici - TI - 1A
```

- 8. Menampilkan Antrian Terakhir

```
Pilih menu: 8
Mahasiswa terakhir:
NIM - NAMA - PRODI - KELAS
124 - Cici - TI - 1A
```

- 9. Cetak Jumlah Antrian

```
Pilih menu: 9
Jumlah Mahasiswa dalam antrian: 2
```

- 10. Cetak Jumlah Yang Sudah Melakukan KRS

```
Pilih menu: 10
Jumlah Mahasiswa yang sudah melakukan KRS: 2
```

- 11. Cetak Jumlah Yang Belum Melakukan KRS

```
Pilih menu: 11
Jumlah Mahasiswa yang belum melakukan KRS: 28
```

- 0. Keluar

```
Pilih menu: 0
Terimakasih.
PS G:\DATA DIKO\POLINEMA TI\SEMESTER 2\PRAK ASD\praktikumASD>
```

Tambahan

- 4. Menambah Antrian bila size == max

```
Pilih menu: 4
NIM   : 121
Nama  : ANIT
Prodi : TI
Kelas : 1A
Antrian penuh, tidak dapat menambah mahasiswa.
```

- 5. Memproses 2 Antrian KRS bila antrian kosong

```
Pilih menu: 5
Antrian kosong.
```

- 5. Memproses 2 Antrian KRS apabila max kuota DPA terpenuhi

```
Pilih menu: 5
Kuota KRS DPA tidak mencukupi untuk memproses 2 mahasiswa.
```