POLITEKNIK NEGERI MALANG

TEKNOLOGI INFORMASI

TEKNIK INFORMATIKA

Mohammad Ariq Baihaqi

244107020161

TI – 1A

16

2.1 Percobaan 1 : Operasi Dasar Queue

2.1.2. Verifikasi Hasil Percobaan

Class Queue

```
package Minggu11;
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.print(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("Qeueue masih kosong");
        } else {
            dt = data[front];
            size--;
            if (isEmpty()) {
                front = rear = -1;
            } else {
                if (front == max - 1) {
                    front = 0;
                } else {
                    front++;
                }
            }
        }
        return dt;
    }
```

Class QeueueMain

```
package Minggu11;
import java.util.Scanner;
public class QueueMain {
    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();
```

```
case 5:
                    q.clear();
                    break;
                default:
                    System.out.println("Pilihan tidak valid.");
                    break;
            }
        } while (pilih >= 1 && pilih <= 5);
    }

    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("-----------------------------");
    }
}
```

OUTPUT

```
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
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
```

2.1.3. Pertanyaan

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

- Karena nilai -1 pada front dan rear menunjukkan bahwa queue dalam keadaan kosong dan belumada elemen yang dimasukkan

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

```
If (rear == max  -1) {
Rear = 0;
```

- Jika rear mencapai akhir array (max -1) maka nilai rear di reset ke indeks awal agar queue bisa melanjutkan pengisian dari depan selama slot tersebut kosong

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

```
If (front == max  -1) {
Front = 0;
```

- Jika indeks terakhir array (max -1)maka diarahkan ke indeks awal untuk mengambil elemen dari awal array jika tersedia

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

- Karena elemen pertama yang valid dalam queue selalu berada di indeks front

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

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

- Untuk menggeser indeks I ke elemen berikutnya secara melingkar

6. Tunjukkan potongan kode program yang merupakan queue overflow!

- ```
  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!

```
public void Enqueue(int dt) {
    if (isFull()) {
        System.out.println("Queue sudah penuh (Overflow). Program
dihentikan.");
        System.exit(1); // Menghentikan program
    } else {
        if (isEmpty()) {
            front = rear = 0;
        } else {
            rear = (rear + 1) % max;
        }
        data[rear] = dt;
        size++;
    }
}
```

2.2. Percobaan 2 : Antrian Layanan Akademik

2.2.2 Verifikasi Hasil Percobaan

Class AntriLayanan

```
package Minggu11;

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

    public AntriLayanan(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() {
```

```
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;
            data[index].tampilkanData();
        }
    }

    public int getJumlahAntrian() {
```

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

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

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

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

Class LayananAkademikSIAKAD

```
package Minggu11;
import java.util.Scanner;

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

        AntriLayanan antrian = new AntriLayanan(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 Terdepan");
            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("Masukkan NIM: ");
                    String nim = sc.nextLine();
                    System.out.print("Masukkan Nama: ");
                    String nama = sc.nextLine();
                    System.out.print("Masukkan Prodi: ");
                    String prodi = sc.nextLine();
                    System.out.print("Masukkan 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("Mahasiswa yang dilayani:");
                        dilayani.tampilkanData();
                    }
                    break;
                case 3:
                    antrian.lihatTerdepan();
                    break;
                case 4:
                    antrian.tampilkanSemua();
                    break;
                case 5:
                    System.out.println("Jumlah mahasiswa dalam antrian: " +
antrian.getJumlahAntrian());
                    break;
                case 0:
                    System.out.println("Terima Kasih.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid.");
                    break;
            }
        } while (pilihan != 0);



    }
}
```

OUTPUT

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

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

=== Menu Antrian Layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Terdepan
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
123-Aldi-TI-1A
124-Bobi-TI-IG
```

```
1. Tambah Mahasiswa ke Antrian
2. Layani Terdepan
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
0. Keluar
Pilih menu: 2
Mahasiswa yang dilayani:
123-Aldi-TI-1A

=== Menu Antrian Layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Terdepan
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
124-Bobi-TI-IG

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

=== Menu Antrian Layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Terdepan
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!

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

```
System.out.println("6. Cek Antrian paling belakang");
case 6:
                    antrian.lihatAkhir();
                    break;
```

2.3 Tugas

Mahasiswa
Nim : String Nama: String Prodi: String Kelas: String
tampilkanData()_

AntrianKRS
data: Mahasiswa[] front: int rear: int size: int max: int totalProses: int
isFull(), isEmpty(), tambah(), panggil(), lihatDepan(), Akhir(), TampilkanSemua() jumlahAntrian(), jumlahDiproses(), jumlahBelumProses(), clear()

Class Mahasiswa16

```
package Minggu11;


public class Mahasiswa16 {
    String nim, nama, prodi, kelas;

    public Mahasiswa16(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 AntrianKRS

```
package Minggu11;

public class AntrianKRS {
    Mahasiswa[] data;
    int front, rear, size, max, totalProses;

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

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

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

    public void tambah(Mahasiswa mhs) {
        if (isFull()) {
            System.out.println("Antrian penuh.");
            return;
        }
        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("Antrian kurang dari 2. Tidak bisa proses.");
            return;
        }
        System.out.println("Memproses 2 mahasiswa:");
        for (int i = 0; i < 2; i++) {
            data[front].tampilkanData();
            front = (front + 1) % max;
            size--;
            totalProses++;
        }
    }

    public void tampilkanSemua() {
        if (isEmpty()) {
            System.out.println("Antrian kosong.");
            return;
        }
        System.out.println("Daftar Mahasiswa dalam Antrian:");
        for (int i = 0; i < size; i++) {
            int idx = (front + i) % max;
            data[idx].tampilkanData();
        }
    }

    public void lihat2Terdepan() {
        if (size < 2) {
            System.out.println("Kurang dari 2 mahasiswa dalam antrian.");
            return;
        }
        System.out.println("2 Mahasiswa Terdepan:");
        data[front].tampilkanData();
        data[(front + 1) % max].tampilkanData();
    }
```

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

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

    public int getJumlahAntrian() {
        return size;
    }

    public int getJumlahDiproses() {
        return totalProses;
    }

    public int getJumlahBelumProses() {
        return 30 - totalProses;
    }
}
```

Class KRSmain

```
package Minggu11;
import java.util.Scanner;

public class KRSmain {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        AntrianKRS antrian = new AntrianKRS(10);
        int pilih;

        do {
            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 Mahasiswa Terdepan");
            System.out.println("4. Lihat Mahasiswa Paling Belakang");
            System.out.println("5. Tampilkan Semua Antrian");
            System.out.println("6. Cetak Jumlah Antrian");
            System.out.println("7. Cetak Jumlah Sudah Proses");
            System.out.println("8. Cetak Jumlah Belum Proses");
            System.out.println("9. Kosongkan Antrian");
            System.out.println("0. Keluar");
            System.out.print("Pilih menu: ");
            pilih = sc.nextInt(); sc.nextLine();

            switch (pilih) {
                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();
                    antrian.tambah(new Mahasiswa(nim, nama, prodi, kelas));
                    break;
                case 2:
                    antrian.panggilKRS();
                    break;
                case 3:
                    antrian.lihat2Terdepan();
                    break;
                case 4:
                    antrian.lihatAkhir();
                    break;
                case 5:
                    antrian.tampilkanSemua();
                    break;
                case 6:
                    System.out.println("Jumlah dalam antrian: " +
antrian.getJumlahAntrian());
                    break;
                case 7:
                    System.out.println("Jumlah yang sudah proses KRS: " +
antrian.getJumlahDiproses());
                    break;
                case 8:
                    System.out.println("Jumlah belum proses KRS: " +
antrian.getJumlahBelumProses());
                    break;
                case 9:
                    antrian.clear();
                    break;
                case 0:
                    System.out.println("Terima kasih.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid.");
            }
        } while (pilih != 0);
    }
}
```

OUTPUT

```
=== MENU ANTRIAN KRS ===
1. Tambah Mahasiswa ke Antrian
2. Panggil 2 Mahasiswa untuk Proses KRS
3. Lihat 2 Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Belakang
5. Tampilkan Semua Antrian
6. Cetak Jumlah Antrian
7. Cetak Jumlah Sudah Proses
8. Cetak Jumlah Belum Proses
9. Kosongkan Antrian
0. Keluar
Pilih menu: 1
NIM: 123
Nama: Hanif
Prodi: TM
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 Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Belakang
5. Tampilkan Semua Antrian
6. Cetak Jumlah Antrian
7. Cetak Jumlah Sudah Proses
8. Cetak Jumlah Belum Proses
9. Kosongkan Antrian
0. Keluar
Pilih menu: 1
NIM: 134
Nama: Ilham
Prodi: TI
Kelas: 1B
Mahasiswa berhasil ditambahkan ke antrian.
```

```
=== MENU ANTRIAN KRS ===
1. Tambah Mahasiswa ke Antrian
2. Panggil 2 Mahasiswa untuk Proses KRS
3. Lihat 2 Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Belakang
5. Tampilkan Semua Antrian
6. Cetak Jumlah Antrian
7. Cetak Jumlah Sudah Proses
8. Cetak Jumlah Belum Proses
9. Kosongkan Antrian
0. Keluar
Pilih menu: 5
Daftar Mahasiswa dalam Antrian:
123-Hanif-TM-1A
134-Ilham-TI-1B

=== MENU ANTRIAN KRS ===
1. Tambah Mahasiswa ke Antrian
2. Panggil 2 Mahasiswa untuk Proses KRS
3. Lihat 2 Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Belakang
5. Tampilkan Semua Antrian
6. Cetak Jumlah Antrian
7. Cetak Jumlah Sudah Proses
8. Cetak Jumlah Belum Proses
9. Kosongkan Antrian
0. Keluar
Pilih menu: 2
Memproses 2 mahasiswa:
123-Hanif-TM-1A
134-Ilham-TI-1B
```

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

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

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

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

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

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

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

Link Github: https://github.com/Ariqq16/semester2