LAPORAN HASIL PRAKTIKUM
Algoritma Struktur Data
JOBSHEET 10

Muhammad Fitra Adhim Nurrochman
2441007020089
TI 1E

PROGRAM STUDI D-IV TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG

PERCOBAAN 1

Operasi dasar queue

1. Kode program
    a. Class Queue

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

    Queue(int data, int front, int rear, int size,
int max) {
        this.data = new int[data];
        this.front = front;
        this.rear = rear;
        this.size = size;
        this.max = 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("Queue masiih kosong");
        } else {
            dt = data[front];
            size--;
            if (isEmpty()) {
                front = rear = -1;
            } else {
                if (front == max -1) {
                    front = 0;
                } else {
                    front++;
                }
            }
        }
        return dt;
    }
}
```

b. 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);
    }
}
```

2. 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: 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
----------------------
```

3. Pertanyaan
   a. Pada konstruktor, mengapa nilai awal atribut front dan rear bernilai -1, sementara atribut size bernilai 0?
   b. Pada method **Enqueue**, jelaskan maksud dan kegunaan dari potongan kode berikut

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

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

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

   d. Pada method print, mengapa pada proses perulangan variabel i tidak dimulai dari 0 (int i=0), melainkan int i=front?
   e. Perhatikan kembali method print, jelaskan maksud dari potongan kode berikut!

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

   f. Tunjukkan potongan kode program yang merupakan queue overflow!
   g. 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!
4. Jawaban
   a. Front dan rear bernilai -1 sebagai penanda kalau queue masih kosong dan belum ada elemen, sedangkan size bernilai 0 karena memang belum ada elemen yang disimpan pada queue, jikalau front dan rear bernilai 0 berarti dalam queue sudah ada 1 elemen
   b. Kode ini digunakan untuk implementasi circular queue yaitu ketika rear mencapai indeks terakshir maka rear akan kembali ke indeks 0 sebagai rear yang baru
   c. Kode ini juga digunakan untuk impelementasi circular queue ketika front mencapai indeks teakjhir, maka front kembali ke indeks 0
   d. Karena indeks ke 0 tidak selalu front dan indeks terakhir tidak selalu rear, dalam queue indeks 0 tidak selalu berisi elemen pertama atau front, diberikan front sebagai penanda bahwa itu adalah elemen pertama(tidak selalu awal/indeks 0) karena queue bersifat FIFO
   e. Operasi modulo untuk implementasi circular queue, untuk memastikan bahwa indeks i akan tetap dalam rentang 0 sampai max-1 yaitu ketika i mencapai max, maka operasi modulo akan mengembalikan nilai ke 0
   f. Berikut adalah potongan kode ayng merupakan queue overflow

```
67          if (isFull()) {
68              System.out.println(x:"Queue sudah penuh");
69          } else {
```

   g. Disini saya menambahkan System.exit(0); untuk langsung menghentikan program secara keseluruhan saat terjadi overflow atau underflow

```
66      public void Enqueue(int dt){
67          if (isFull()) {
68              System.out.println(x:"Queue sudah penuh");
69              System.exit(status:0);
```

```
84      public int Dequeue() {
85          int dt = 0;
86          if (isEmpty()) {
87              System.out.println(x:"Queue masiih kosong");
88              System.exit(status:0);
```

PERCOBAAN 2

1. Kode program
    a. 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);
    }
}
```

    b. AntrianLayanan

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

   public AntrianLayanan(int max) {
      this.max = max;
      this.data = new Mahasiswa[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;
        }
    }
```

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

c. 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.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 0:
                    System.out.println("Terima kasih.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid.");
            }
        } while (pilihan != 0);

        sc.close();
    }
}
```

2. Verifikasi hasil percobaan

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

3. Pertanyaan
    a. 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!
4. Jawaban
    a. Kode program yang ditambahkan

```
84      public void LihatAkhir() {
85          if (isEmpty()) {
86              System.out.println(x:"Antrian kosong");
87          } else {
88              System.out.println(x:"Mahasiswa di akhir antrian: ");
89              System.out.println(x:"NIM - NAMA - PRODI - KELAS");
90              data[rear].tampilkanData();
91          }
```

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

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

    b. Hasil kode program

```
Pilih menu: 1
NIM    : 123
Nama   : adh
Prodi  : ti
Kelas  : 1a
adh 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    : 234
Nama   : sdh
Prodi  : ti
Kelas  : 1e
sdh 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 di akhir antrian:
NIM - NAMA - PRODI - KELAS
234 - sdh - ti - 1e
```

TUGAS

1. Kode program
    a. Mahasiswa.java

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

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

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

    b. AntrianLayananKRS.java

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

    public AntrianLayananKRS(int max) {
        this.max = max;
        this.data = new Mahasiswa[max];
        this.front = 0;
        this.rear = -1;
        this.size = 0;
        this.jumlahProses = 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!");
            return;
        }
        rear = (rear + 1) % max;
        data[rear] = mhs;
        size++;
        System.out.println(mhs.nama + " berhasil ditambahkan ke antrian");
    }

    public void prosesKRS() {
        if (isEmpty()) {
            System.out.println("Antrian kosong!");
            return;
        }
        System.out.println("\nMemproses 2 mahasiswa:");
        for (int i = 0; i < 2 && !isEmpty(); i++) {
            Mahasiswa mhs = data[front];
            mhs.sudahProses = true;
            System.out.println("\nMahasiswa ke-" + (i+1) + ":");
            mhs.tampilkanData();
            front = (front + 1) % max;
            size--;
            jumlahProses++;
        }
    }

    public void tampilkanSemua() {
        if (isEmpty()) {
```

```
System.out.println("Antrian kosong!");
            return;
        }
        System.out.println("\nDaftar 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 void tampilkan2Depan() {
        if (isEmpty()) {
            System.out.println("Antrian kosong!");
            return;
        }
        System.out.println("\n2 Antrian Terdepan:");
        System.out.println("NIM - NAMA - PRODI - KELAS");
        for (int i = 0; i < 2 && i < size; i++) {
            int index = (front + i) % max;
            System.out.print((i+1) + ". ");
            data[index].tampilkanData();
        }
    }

    public void tampilkanAkhir() {
        if (isEmpty()) {
            System.out.println("Antrian kosong!");
            return;
        }
        System.out.println("\nAntrian Terakhir:");
        System.out.println("NIM - NAMA - PRODI - KELAS");
        data[rear].tampilkanData();
    }

    public void getJumlahAntrian() {
        System.out.println("\nJumlah dalam antrian: " + size);
        System.out.println("Jumlah sudah diproses: " + jumlahProses);
        System.out.println("Jumlah belum diproses: " + (size));
    }
}
```

c. LayananKRS.java

```
import java.util.Scanner;

public class LayananKRS {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        AntrianLayananKRS antrian = new AntrianLayananKRS(5);
        int pilihan;

        do {
            System.out.println("\n=== Menu Antrian KRS ===");
            System.out.println("1. Tambah Antrian");
            System.out.println("2. Proses KRS (2 Mahasiswa)");
            System.out.println("3. Lihat Semua Antrian");
            System.out.println("4. Lihat 2 Antrian Terdepan");
            System.out.println("5. Lihat Antrian Terakhir");
            System.out.println("6. Cek Jumlah 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();

                    Mahasiswa mhs = new Mahasiswa(nim, nama, prodi, kelas);
                    antrian.tambahAntrian(mhs);
                    break;

                case 2:
                    antrian.prosesKRS();
                    break;

                case 3:
                    antrian.tampilkanSemua();
                    break;

                case 4:
                    antrian.tampilkan2Depan();
                    break;

                case 5:
                    antrian.tampilkanAkhir();
                    break;

                case 6:
                    antrian.getJumlahAntrian();
                    break;

                case 0:
                    System.out.println("Terima kasih!");
                    break;

                default:
                    System.out.println("Pilihan tidak valid!");
            }
        } while (pilihan != 0);

        sc.close();
    }
`
```

2. Hasil verifikasi kode program

```
=== Menu Antrian KRS ===
1. Tambah Antrian
2. Proses KRS (2 Mahasiswa)
3. Lihat Semua Antrian
4. Lihat 2 Antrian Terdepan
5. Lihat Antrian Terakhir
6. Cek Jumlah Antrian
0. Keluar
Pilih menu: 1
NIM    : 123
Nama   : adh
Prodi  : ti
Kelas  : 1a
adh berhasil ditambahkan ke antrian

=== Menu Antrian KRS ===
1. Tambah Antrian
2. Proses KRS (2 Mahasiswa)
3. Lihat Semua Antrian
4. Lihat 2 Antrian Terdepan
5. Lihat Antrian Terakhir
6. Cek Jumlah Antrian
0. Keluar
Pilih menu: 1
NIM    : 234
Nama   : qwe
Prodi  : ti
Kelas  : 1g
qwe berhasil ditambahkan ke antrian

=== Menu Antrian KRS ===
1. Tambah Antrian
2. Proses KRS (2 Mahasiswa)
3. Lihat Semua Antrian
4. Lihat 2 Antrian Terdepan
5. Lihat Antrian Terakhir
6. Cek Jumlah Antrian
0. Keluar
Pilih menu: 1
NIM    : 456
Nama   : zxc
Prodi  : ti
Kelas  : 1f
zxc berhasil ditambahkan ke antrian

=== Menu Antrian KRS ===
1. Tambah Antrian
2. Proses KRS (2 Mahasiswa)
3. Lihat Semua Antrian
4. Lihat 2 Antrian Terdepan
5. Lihat Antrian Terakhir
6. Cek Jumlah Antrian
0. Keluar
Pilih menu: 3

Daftar Mahasiswa dalam Antrian:
NIM - NAMA - PRODI - KELAS
1. 123 - adh - ti - 1a
2. 234 - qwe - ti - 1g
3. 456 - zxc - ti - 1f

=== Menu Antrian KRS ===
1. Tambah Antrian
2. Proses KRS (2 Mahasiswa)
3. Lihat Semua Antrian
4. Lihat 2 Antrian Terdepan
5. Lihat Antrian Terakhir
6. Cek Jumlah Antrian
0. Keluar
Pilih menu: 4

2 Antrian Terdepan:
NIM - NAMA - PRODI - KELAS
1. 123 - adh - ti - 1a
2. 234 - qwe - ti - 1g

=== Menu Antrian KRS ===
1. Tambah Antrian
2. Proses KRS (2 Mahasiswa)
3. Lihat Semua Antrian
4. Lihat 2 Antrian Terdepan
5. Lihat Antrian Terakhir
6. Cek Jumlah Antrian
0. Keluar
Pilih menu: 5

Antrian Terakhir:
NIM - NAMA - PRODI - KELAS
456 - zxc - ti - 1f

=== Menu Antrian KRS ===
1. Tambah Antrian
2. Proses KRS (2 Mahasiswa)
3. Lihat Semua Antrian
4. Lihat 2 Antrian Terdepan
5. Lihat Antrian Terakhir
6. Cek Jumlah Antrian
0. Keluar
Pilih menu: 6

Jumlah dalam antrian: 3
Jumlah sudah diproses: 0
Jumlah belum diproses: 3
```

```
=== Menu Antrian KRS ===
1. Tambah Antrian
2. Proses KRS (2 Mahasiswa)
3. Lihat Semua Antrian
4. Lihat 2 Antrian Terdepan
5. Lihat Antrian Terakhir
6. Cek Jumlah Antrian
0. Keluar
Pilih menu: 2

Memproses 2 mahasiswa:

Mahasiswa ke-1:
123 - adh - ti - 1a

Mahasiswa ke-2:
234 - qwe - ti - 1g

=== Menu Antrian KRS ===
1. Tambah Antrian
2. Proses KRS (2 Mahasiswa)
3. Lihat Semua Antrian
4. Lihat 2 Antrian Terdepan
5. Lihat Antrian Terakhir
6. Cek Jumlah Antrian
0. Keluar
Pilih menu: 3

Daftar Mahasiswa dalam Antrian:
NIM - NAMA - PRODI - KELAS
1. 456 - zxc - ti - 1f

=== Menu Antrian KRS ===
1. Tambah Antrian
2. Proses KRS (2 Mahasiswa)
3. Lihat Semua Antrian
4. Lihat 2 Antrian Terdepan
5. Lihat Antrian Terakhir
6. Cek Jumlah Antrian
0. Keluar
Pilih menu: 4

2 Antrian Terdepan:
NIM - NAMA - PRODI - KELAS
1. 456 - zxc - ti - 1f
```

```
=== Menu Antrian KRS ===
1. Tambah Antrian
2. Proses KRS (2 Mahasiswa)
3. Lihat Semua Antrian
4. Lihat 2 Antrian Terdepan
5. Lihat Antrian Terakhir
6. Cek Jumlah Antrian
0. Keluar
Pilih menu: 5

Antrian Terakhir:
NIM - NAMA - PRODI - KELAS
456 - zxc - ti - 1f

=== Menu Antrian KRS ===
1. Tambah Antrian
2. Proses KRS (2 Mahasiswa)
3. Lihat Semua Antrian
4. Lihat 2 Antrian Terdepan
5. Lihat Antrian Terakhir
6. Cek Jumlah Antrian
0. Keluar
Pilih menu: 6

Jumlah dalam antrian: 1
Jumlah sudah diproses: 2
Jumlah belum diproses: 1
```