POLITEKNIK NEGERI MALANG

TEKNOLOGI INFORMASI

TEKNIK INFORMATIKA

Nama	: Muhammad Nuril Huda
Kelas	: TI-1A
No	: 19
Mata Kuliah	: Algoritma dan Struktur Data

2.1 Percobaan 1: Operasi Dasar Queue

2.1.1 Kode Program

- Kode Program 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.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 masih kosong");
        } else {
            dt = data[front];
            size--;
            if (IsEmpty()) {
                front = rear = -1;
            } else {
                if (front == max - 1) {
                    front = 0;
                } else {
                    front++;
                }
            }
        }
        return dt;
    }
}
```

- Kode Program 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();
            System.out.print("Masukkan menu pilihan: ");
            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);
    }
}
```

2.1.2 Hasil Kode Program

```
Masukkan kapasitas queue: 4
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-------------------------
Masukkan menu pilihan: 1
Masukkan data baru: 15
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-------------------------
Masukkan menu pilihan: 1
Masukkan data baru: 31
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-------------------------
Masukkan menu pilihan: 4
Elemen terdepan: 15
```

2.1.3 Pertanyaan

1. Pada konstruktor, mengapa nilai awal atribut front dan rear bernilai -1, sementara atribut size bernilai 0?
   - Nilai awal front dan rear bernilai -1 berfungsi untuk menandakan bahwa antrian masih kosong, artinya belum ada elemen yang masuk. Karena jika front dan rear bernilai 0 maka menandakan sudah ada elemen yang masuk
   - Sedangkan nilai size bernilai 0 berfungsi untuk menandakan bahwa jumlah elemen dalam antrian adalah nol atau kosong
2. Pada method Enqueue, jelaskan maksud dan kegunaan dari potongan kode berikut!

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

   - Kode ini memiliki maksud sebagai antrian melingkar atau circular queue. Artiny ajika rear sudah berada pada indeks terakhir yaitu max-1 maka rear akan melompak ke indeks awal yaitu 0 untuk menggunakan ruang kosong pada awal array jika ada elemen array yang sudah dikeluarkan sebelumnya
3. Pada method Dequeue, jelaskan maksud dan kegunaan dari potongan kode berikut!

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

   - Kode ini memiliki maksud sebagai antrian melingkar. Artinya jika front sudah berada di indeks terakhir yaitu max-1, maka setelah

elemen dihapus atau di dequeue indeks front akan kembali ke awal array yaitu indek 0 agar antrian bisa terus berfungsi secara melingkar

4. Pada method print, mengapa pada proses perulangan variabel i tidak dimulai dari 0 (int i=0), melainkan int i=front?
   - Karena elemen pertama atau front dari queue tidak selalu dimulai dari 0. Sehingga jika ingin mencetak perulangan harus dimulai dari front bukan dari 0
5. Perhatikan kembali method print, jelaskan maksud dari potongan kode berikut!

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

   - Kode tersebut berfungsi untuk menghindari kesalahan indeks keluar dari batas array dan memungkinkan antrian berputar atau circular
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 peek() {
        if (!IsEmpty()) {
            System.out.println("Elemen terdepan: " + data[front]);
        } else {
            System.out.println("Queue masih kosong");
            System.exit(0);
        }
    }
    public void print() {
        if (IsEmpty()) {
            System.out.println("Queue masih kosong");
            System.exit(0);
        } 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");
            System.exit(0);
        }
    }
    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++;
        }
    }
```

```
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;
    }
```

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

2.2. Percobaan 2: Antrian Layanan Akademik

2.2.1 Kode Program

- Kode Program 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);
    }
}
```

- Kode Program 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 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.print("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;
    }
}
```

- Kode Program 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 \t: ");
                    String nim = sc.nextLine();
                    System.out.print("Nama \t: ");
                    String nama = sc.nextLine();
                    System.out.print("Prodi \t: ");
                    String prodi = sc.nextLine();
                    System.out.print("Kelas \t: ");
                    String kelas = sc.nextLine();
                    Mahasiswa mhs = new Mahasiswa(nim, nama, prodi, kelas);
                    antrian.tambahAntrian(mhs);
                    break;
                case 2:
                    Mahasiswa dilayani = antrian.layaniMahasiswa();
                    if (dilayani != null) {
                        System.out.print("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);
    }
}
```

2.2.3 Hasil Kode Program

```
=== 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!

- Modifikasi Kode Program AntrianLayanan

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

- Modifikasi Program LayananAkademikSikad

```
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. Lihat Mahasiswa 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 \t: ");
                    String nim = sc.nextLine();
                    System.out.print("Nama \t: ");
                    String nama = sc.nextLine();
                    System.out.print("Prodi \t: ");
                    String prodi = sc.nextLine();
                    System.out.print("Kelas \t: ");
                    String kelas = sc.nextLine();
                    Mahasiswa mhs = new Mahasiswa(nim, nama, prodi, kelas);
                    antrian.tambahAntrian(mhs);
                    break;
                case 2:
                    Mahasiswa dilayani = antrian.layaniMahasiswa();
                    if (dilayani != null) {
                        System.out.print("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);
```

2.3 Tugas

2.3.1 Kode Program

- Kode Program krs

```
public class krs {
    String nim;
    String nama;
    String prodi;
    String kelas;
    public krs(String nim, String nama, String prodi, String kelas) {
        this.nim = nim;
        this.nama = nama;
        this.prodi = prodi;
        this.kelas = kelas;
    }
    public void tampilData() {
        System.out.println(nim + " - " + nama + " - " + prodi + " - " + kelas);
    }
}
```

- Kode Program krsMethod

```
public class krsMethod {
    krs[] dta;
    int front;
    int rear;
    int size;
    int max;
    int sudahDilayani;
    public krsMethod(int max) {
        this.max = max;
        this.dta = new krs[max];
        this.front = 0;
        this.rear = -1;
        this.size = 0;
        this.sudahDilayani = 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;
            System.out.println("Antrian berhasil dikosongkan");
        } else {
            System.out.println("Antrian masih kosong");
            return;
        }
    }
    public void tambahAntrian(krs k) {
        if (isFull()) {
            System.out.println("Antrian penuh, tidak dapat menambah mahasiswa.");
            return;
        }
        rear = (rear + 1) % max;
        dta[rear] = k;
        size++;
        System.out.println(k.nama + " berhasil masuk ke antrian.");
    }
```

```
    public PasanganKRS memanggilAntrian() {
        if (isEmpty()) {
            System.out.println("Antrian kosong.");
            return null;
        } else if (size < 2){
            System.out.println("Antrian kurang dari 2.");
            return null;
        }
        krs k1 = dta[front];
        front = (front + 1) % max;
        size--;
        krs k2 = dta[front];
        front = (front + 1) % max;
        size--;
        sudahDilayani +=2;
        return new PasanganKRS(k1, k2);
    }
    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) + ". ");
            dta[index].tampilData();
        }
    }
    public void lihat2Terdepan() {
        if (isEmpty()) {
            System.out.println("Antrian kosong.");
        } else {
            System.out.println("Mahasiswa terdepan: ");
            System.out.println("NIM - NAMA - PRODI - KELAS");
            dta[front].tampilData();
            dta[front+1].tampilData();
        }
    }
```

```
public void lihatAkhir(){
        if (isEmpty()) {
            System.out.println("Antrian kosong.");
        } else {
            System.out.println("Mahasiswa paling belakang: ");
            System.out.println("NIM - NAMA - PRODI - KELAS");
            dta[rear].tampilData();
        }
    }
    public int jumlahAntrian(){
        return size;
    }
    public int sudahDiproses(){
        return sudahDilayani;
    }
}
```

- Kode Program PasanganKRS

```
public class PasanganKRS {
    krs k1;
    krs k2;

    public PasanganKRS(krs k1, krs k2) {
        this.k1 = k1;
        this.k2 = k2;
    }
    public void tampilData() {
        System.out.println("Mahasiswa 1:");
        k1.tampilData();
        System.out.println("Mahasiswa 2:");
        k2.tampilData();
    }
}
```

- Kode Program krsMain

```
import java.util.Scanner;
public class krsMain {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        krsMethod krsm = new krsMethod(10);
        int pilihan;
        do {
            System.out.println("\n=== Menu Antrian Layanan Akademik ===");
            System.out.println("1. Tambah Antrian");
            System.out.println("2. Proses KRS");
            System.out.println("3. Menampilkan Semua Antrian");
            System.out.println("4. Menampilkan 2 Antrian Terdepan");
            System.out.println("5. Menmapilkan Antrian Paling Akhir");
            System.out.println("6. Lihat Jumlah Antrian");
            System.out.println("7. Lihat Jumlah Yang Sudah Dilayani");
            System.out.println("0. Keluar");
            System.out.print("Pilih menu: ");
            pilihan = sc.nextInt(); sc.nextLine();
            switch (pilihan) {
                case 1:
                    System.out.print("NIM \t: ");
                    String nim = sc.nextLine();
                    System.out.print("Nama \t: ");
                    String nama = sc.nextLine();
                    System.out.print("Prodi \t: ");
                    String prodi = sc.nextLine();
                    System.out.print("Kelas \t: ");
                    String kelas = sc.nextLine();
                    krs k = new krs(nim, nama, prodi, kelas);
                    krsm.tambahAntrian(k);
                    break;
```

```
                case 2:
                    PasanganKRS diproses = krsm.memanggilAntrian();
                    if (diproses != null) {
                        System.out.println("Memanggil dua mahasiswa untuk proses
KRS:");
                        diproses.tampilData();
                    }
                    break;
                case 3:
                    krsm.tampilkanSemua();
                    break;
                case 4:
                    krsm.lihat2Terdepan();
                    break;
                case 5:
                    krsm.lihatAkhir();
                    break;
                case 6:
                    System.out.println("Jumlah Antrian: " +
krsm.jumlahAntrian());
                    break;
                case 7:
                    System.out.println("Jumlah Yang Sudah Dilayani: " +
krsm.jumlahAntrian());
                    break;
                case 0:
                    System.out.println("Terima kasih.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid.");
            }
        } while (pilihan != 0);
    }
}
```

2.3.2 Hasil Kode Program

```
=== Menu Antrian Layanan Akademik ===
1. Tambah Antrian
2. Proses KRS
3. Menampilkan Semua Antrian
4. Menampilkan 2 Antrian Terdepan
5. Menmapilkan Antrian Paling Akhir
6. Lihat Jumlah Antrian
7. Lihat Jumlah Yang Sudah Dilayani
0. Keluar
Pilih menu: 1
NIM     : 123
Nama    : Nuril
Prodi   : TI
Kelas   : 1A
Nuril berhasil masuk ke antrian.

=== Menu Antrian Layanan Akademik ===
1. Tambah Antrian
2. Proses KRS
3. Menampilkan Semua Antrian
4. Menampilkan 2 Antrian Terdepan
5. Menmapilkan Antrian Paling Akhir
6. Lihat Jumlah Antrian
7. Lihat Jumlah Yang Sudah Dilayani
0. Keluar
Pilih menu: 1
NIM     : 456
Nama    : Huda
Prodi   : TI
Kelas   : 1B
Huda berhasil masuk ke antrian.

=== Menu Antrian Layanan Akademik ===
1. Tambah Antrian
2. Proses KRS
3. Menampilkan Semua Antrian
4. Menampilkan 2 Antrian Terdepan
5. Menmapilkan Antrian Paling Akhir
6. Lihat Jumlah Antrian
7. Lihat Jumlah Yang Sudah Dilayani
0. Keluar
Pilih menu: 1
NIM     : 789
Nama    : Zeta
Prodi   : TI
Kelas   : 1C
Zeta berhasil masuk ke antrian.

=== Menu Antrian Layanan Akademik ===
1. Tambah Antrian
2. Proses KRS
3. Menampilkan Semua Antrian
4. Menampilkan 2 Antrian Terdepan
5. Menmapilkan Antrian Paling Akhir
6. Lihat Jumlah Antrian
7. Lihat Jumlah Yang Sudah Dilayani
0. Keluar
Pilih menu: 1
NIM     : 194
Nama    : Ahmad
Prodi   : TI
Kelas   : 1D
Ahmad berhasil masuk ke antrian.
```

```
=== Menu Antrian Layanan Akademik ===
1. Tambah Antrian
2. Proses KRS
3. Menampilkan Semua Antrian
4. Menampilkan 2 Antrian Terdepan
5. Menmapilkan Antrian Paling Akhir
6. Lihat Jumlah Antrian
7. Lihat Jumlah Yang Sudah Dilayani
0. Keluar
Pilih menu: 3
Daftar Mahasiswa dalam Antrian:
NIM - NAMA - PRODI - KELAS
1. 123 - Nuril - TI - 1A
2. 456 - Huda - TI - 1B
3. 789 - Zeta - TI - 1C
4. 194 - Ahmad - TI - 1D

=== Menu Antrian Layanan Akademik ===
1. Tambah Antrian
2. Proses KRS
3. Menampilkan Semua Antrian
4. Menampilkan 2 Antrian Terdepan
5. Menmapilkan Antrian Paling Akhir
6. Lihat Jumlah Antrian
7. Lihat Jumlah Yang Sudah Dilayani
0. Keluar
Pilih menu: 4
Mahasiswa terdepan:
NIM - NAMA - PRODI - KELAS
123 - Nuril - TI - 1A
456 - Huda - TI - 1B

=== Menu Antrian Layanan Akademik ===
1. Tambah Antrian
2. Proses KRS
3. Menampilkan Semua Antrian
4. Menampilkan 2 Antrian Terdepan
5. Menmapilkan Antrian Paling Akhir
6. Lihat Jumlah Antrian
7. Lihat Jumlah Yang Sudah Dilayani
0. Keluar
Pilih menu: 5
Mahasiswa paling belakang:
NIM - NAMA - PRODI - KELAS
194 - Ahmad - TI - 1D

=== Menu Antrian Layanan Akademik ===
1. Tambah Antrian
2. Proses KRS
3. Menampilkan Semua Antrian
4. Menampilkan 2 Antrian Terdepan
5. Menmapilkan Antrian Paling Akhir
6. Lihat Jumlah Antrian
7. Lihat Jumlah Yang Sudah Dilayani
0. Keluar
Pilih menu: 6
Jumlah Antrian: 4
```

```
=== Menu Antrian Layanan Akademik ===
1. Tambah Antrian
2. Proses KRS
3. Menampilkan Semua Antrian
4. Menampilkan 2 Antrian Terdepan
5. Menmapilkan Antrian Paling Akhir
6. Lihat Jumlah Antrian
7. Lihat Jumlah Yang Sudah Dilayani
0. Keluar
Pilih menu: 2
Memanggil dua mahasiswa untuk proses KRS:
Mahasiswa 1:
123 - Nuril - TI - 1A
Mahasiswa 2:
456 - Huda - TI - 1B

=== Menu Antrian Layanan Akademik ===
1. Tambah Antrian
2. Proses KRS
3. Menampilkan Semua Antrian
4. Menampilkan 2 Antrian Terdepan
5. Menmapilkan Antrian Paling Akhir
6. Lihat Jumlah Antrian
7. Lihat Jumlah Yang Sudah Dilayani
0. Keluar
Pilih menu: 7
Jumlah Yang Sudah Dilayani: 2

=== Menu Antrian Layanan Akademik ===
1. Tambah Antrian
2. Proses KRS
3. Menampilkan Semua Antrian
4. Menampilkan 2 Antrian Terdepan
5. Menmapilkan Antrian Paling Akhir
6. Lihat Jumlah Antrian
7. Lihat Jumlah Yang Sudah Dilayani
0. Keluar
Pilih menu: 0
Terima kasih.
```

2.3.3 Diagram Class

krs
nim: String nama: String prodi: String kelas: String
krs(nim: String, nama: String, prodi: String, kelas: String tampilData(): void

PasanganKRS
K1: krs K2: krs
pasanganKRS(k1: krs, k2: krs) tampilData(): void

krsMethod
dta: krs[ ] front: int rear: int max: int sudahDilayani: int
krsMethod(max: int) isEmpety(): boolean isFull(): boolean clear(): void tambahAntrian(k: krs): void memanggilAntrian(): PasanganKRS tampilkanSemua(): void lihat2Terdepan(): void lihatAkhir(): void jumlahAntrian(): int sudahDiproses(): int

krsMain
sc: Scanner krsm: krsMethod
main(String[ ]): void

Link Github: https://github.com/nurilhuda05/Algoritma-dan-Struktur-Data.git