LAPORAN HASIL PRAKTIKUM

ALGORITMA DAN STRUKTUR DATA

JOBSHEET 10

ATHAULLA HAFIZH

244107020030

TI 1 E

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2024

2. Praktikum

Percobaan 1

Kode program class Queue

```
package P1Jobsheet10;
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 class QueueMain

```
package P1Jobsheet10;
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("Pilih 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;
                default:
                    System.out.println("Keluar dari program");
                    break;
            }
        } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih ==
4 || pilih == 5);
    }
}
```

Output

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

Pertanyaan

1. Nilai awal atribut front dan rear bernilai -1 karena bentuk dari queue ini kan seperti array jadi wajib diinisialisasi dari -1 yang bermakna “kosong” karena indeks array mulai dari 0, jika atribut size bernilai 0 karena sesuai dengan namanya yaitu ukuran dari queue tersebut jika belum terisi maka diawal pasti bernilai 0 dan akan bertambah sesuai pengisian nantinya.
2. Ketika rear sudah mencapai indeks terakhir dari array (max - 1), tetapi masih ada ruang kosong di awal array (karena beberapa elemen di depan sudah di-dequeue), maka rear akan direset ke posisi 0.
3. Ketika operasi dequeue dilakukan dan pointer front sudah mencapai indeks terakhir array (max - 1), maka front akan direset ke posisi 0.
4. Elemen-elemen queue tidak selalu berada di posisi awal array. Posisi awal elemen queue ditunjukkan oleh front, yang bisa berada di posisi mana saja dalam array karena konsep queue FIFO (First In First Out) juga.
5. Operasi modulo (%) memastikan bahwa ketika i akan melebihi batas array (max), nilai i akan kembali ke 0, sehingga iterasi dapat berlanjut dari awal array.
6. Potongan kode yang menangani kondisi queue overflow adalah :

```
public void Enqueue(int dt) {
    if (IsFull()) {
        System.out.println(x:"Queue sudah penuh");
    } else {
```

7. Perubahan kode program (Menggunakan metode helper) di class queue

```
public void safeEnqueue(int dt) {
    if (IsFull()) {
        System.out.println(x:"Queue sudah penuh");
        System.out.println(x:"Program dihentikan karena queue overflow!");
        System.exit(status:1);
    } else {
        Enqueue(dt);
    }
}

Windsurf: Refactor | Explain | Generate Javadoc | ×
public int safeDequeue() {
    if (IsEmpty()) {
        System.out.println(x:"Queue masih kosong");
        System.out.println(x:"Program dihentikan karena queue underflow!");
        System.exit(status:1);
        return -1;
    }
    return Dequeue();
}
```

Dan perubahan di QueueMain

```
case 1:
    System.out.print(s:"Masukkan data baru: ");
    int dataMasuk = sc.nextInt();
    Q.safeEnqueue(dataMasuk);
    break;
case 2:
    int dataKeluar = Q.safeDequeue();
    System.out.println("Data yang dikeluarkan: " + dataKeluar);
    break;
```

Percobaan 2

Kode program class Mahasiswa

```
package P2Jobsheet10;

public class Mahasiswa {

    String nim, nama, prodi, 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 class AntrianLayanan

```
package P2Jobsheet10;
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.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;
    }

    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;
    }
}
```

Kode program class LayananAkademikSIAKAD

```
package P2Jobsheet10;
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.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);

        sc.close();
    }
}
```

Output

```
=== 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.
```

Pertanyaan

Penambahan method baru bernama LihatAkhir pada class AntrianLayanan

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

Penambahan daftar menu 6. Cek Antrian paling belakang pada class LayananAkademikSIAKAD

```
System.out.println(x:"6. Cek Antrian paling belakang");
```

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

Tugas

Kode program class Mahasiswa

```
package TugasJobsheet10;
public class Mahasiswa {
    String nim, nama, prodi, 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);
    }

    public void setProses(boolean status) {
        this.sudahProses = status;
    }

    public boolean getSudahProses() {
        return sudahProses;
    }
}
```

Kode program class AntrianKRS

```
package TugasJobsheet10;
public class AntrianKRS {
    Mahasiswa[] data;
    int front;
    int rear;
    int size;
    int max;
    int jumlahProses;

    public AntrianKRS(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() {
        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 = 0;
            rear = -1;
            size = 0;
            System.out.println("Antrian berhasil dikosongkan");
        } else {
            System.out.println("Antrian masih kosong");
        }
    }

    public void Enqueue(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 void Dequeue() {
        if (IsEmpty()) {
            System.out.println("Antrian kosong.");
            return;
        }

        for (int i = 0; i < 2; i++) {
            if (size > 0) {
                Mahasiswa mhs = data[front];
                mhs.setProses(true);
                jumlahProses++;

                System.out.println("Memproses KRS mahasiswa: ");
                mhs.tampilkanData();

                front = (front + 1) % max;
                size--;
            } else {
                System.out.println("Tidak ada lagi mahasiswa dalam
antrian");
                break;
            }
        }
    }

    public void print() {
        if (IsEmpty()) {
            System.out.println("Antrian kosong.");
            return;
        }
        System.out.println("Daftar Mahasiswa dalam antrian:");
        System.out.println("NO | 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 peek() {
        if (IsEmpty()) {
            System.out.println("Antrian kosong.");
            return;
        }

        System.out.println("Dua Mahasiswa terdepan dalam antrian:");
        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 peekRear() {
        if (IsEmpty()) {
            System.out.println("Antrian kosong.");
            return;
        }

        System.out.println("Mahasiswa paling belakang dalam
antrian:");
        System.out.println("NIM - NAMA - PRODI - KELAS");
        data[rear].tampilkanData();
    }

    public int getSize() {
        return size;
```

```
    }
}
```

Kode program class AntrianKRSMain

```
package TugasJobsheet10;
import java.util.Scanner;
public class AntrianKRSMain {
    public static void menu() {
        System.out.println("\n=== Sistem Antrian Persetujuan KRS
===");
        System.out.println("1. Tambah Mahasiswa ke Antrian");
        System.out.println("2. Proses KRS (2 Mahasiswa)");
        System.out.println("3. Cek Antrian Kosong");
        System.out.println("4. Cek Antrian Penuh");
        System.out.println("5. Tampilkan Semua Antrian");
        System.out.println("6. Tampilkan 2 Antrian Terdepan");
        System.out.println("7. Tampilkan Antrian Paling Akhir");
        System.out.println("8. Tampilkan Jumlah Antrian");
        System.out.println("9. Tampilkan Jumlah Mahasiswa yang Sudah
Proses KRS");
        System.out.println("10. Tampilkan Jumlah Mahasiswa yang
Belum Proses KRS");
        System.out.println("11. Kosongkan Antrian");
        System.out.println("0. Keluar");
        System.out.println("-----------------------------------");
    }

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

        AntrianKRS[] antrian = new AntrianKRS[3];
        for (int i = 0; i < antrian.length; i++) {
            antrian[i] = new AntrianKRS(30);
        }

        int pilih;
```

```
        int dpaPilihan;

        do {
            menu();
            System.out.print("Pilih menu: ");
            pilih = sc.nextInt(); sc.nextLine();

            if (pilih >= 1 && pilih <= 11) {
                System.out.println("\nPilih DPA (1-3): ");
                dpaPilihan = sc.nextInt() - 1; sc.nextLine();

                if (dpaPilihan < 0 || dpaPilihan >= antrian.length)
{
                    System.out.println("DPA tidak valid");
                    continue;
                }
            } else {
                dpaPilihan = 0;
            }

            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();
                    Mahasiswa mhs = new Mahasiswa(nim, nama, prodi,
kelas);
                    antrian[dpaPilihan].Enqueue(mhs);
                    break;
                case 2:
                    System.out.println("Proses KRS untuk DPA " +
(dpaPilihan + 1));
```

```
                    antrian[dpaPilihan].Dequeue();
                    break;
                case 3:
                    if (antrian[dpaPilihan].IsEmpty()) {
                        System.out.println("Antrian DPA " +
(dpaPilihan + 1) + " kosong");
                    } else {
                        System.out.println("Antrian DPA " +
(dpaPilihan + 1) + " tidak kosong");
                    }
                    break;
                case 4:
                    if (antrian[dpaPilihan].IsFull()) {
                        System.out.println("Antrian DPA " +
(dpaPilihan + 1) + " penuh");
                    } else {
                        System.out.println("Antrian DPA " +
(dpaPilihan + 1) + " tidak penuh");
                    }
                    break;
                case 5:
                    System.out.println("Antrian DPA " + (dpaPilihan
+ 1) + ":");
                    antrian[dpaPilihan].print();
                    break;
                case 6:
                    System.out.println("Dua Antrian Terdepan DPA " +
(dpaPilihan + 1) + ":");
                    antrian[dpaPilihan].peek();
                    break;
                case 7:
                    System.out.println("Antrian Terakhir DPA " +
(dpaPilihan + 1) + ":");
                    antrian[dpaPilihan].peekRear();
                    break;
                case 8:
                    System.out.println("Jumlah antrian DPA " +
(dpaPilihan + 1) + ": " +
```

```
                                        antrian[dpaPilihan].getSize()
+ " mahasiswa");
                    break;
                case 9:
                    System.out.println("Jumlah mahasiswa yang sudah
proses KRS di DPA " +
                                        (dpaPilihan + 1) + ": " +
antrian[dpaPilihan].getJumlahProses());
                    break;
                case 10:
                    System.out.println("Jumlah mahasiswa yang belum
proses KRS di DPA " +
                                        (dpaPilihan + 1) + ": " +
antrian[dpaPilihan].getJumlahBelumProses());
                    break;
                case 11:
                    antrian[dpaPilihan].clear();
                    break;
                case 0:
                    System.out.println("Terima kasih telah
menggunakan program antrian KRS.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid.");
            }
        } while (pilih != 0);


        sc.close();
    }
}
```

Output

```
=== Sistem Antrian Persetujuan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Proses KRS (2 Mahasiswa)
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Semua Antrian
6. Tampilkan 2 Antrian Terdepan
7. Tampilkan Antrian Paling Akhir
8. Tampilkan Jumlah Antrian
9. Tampilkan Jumlah Mahasiswa yang Sudah Proses KRS
10. Tampilkan Jumlah Mahasiswa yang Belum Proses KRS
11. Kosongkan Antrian
0. Keluar
---------------------------------
Pilih menu: 1

Pilih DPA (1-3):
1
NIM   : 12313
Nama  : athaulla
Prodi : ti
Kelas : 1e
athaulla berhasil masuk ke antrian
```

```
Pilih menu: 1

Pilih DPA (1-3):
1
NIM   : 123131
Nama  : hafizh
Prodi : ti
Kelas : 1e
hafizh berhasil masuk ke antrian
```

```
Pilih menu: 1

Pilih DPA (1-3):
1
NIM   : 12311312315r
Nama  : athaulla hafizh
Prodi : ti
Kelas : 1e
athaulla hafizh berhasil masuk ke antrian
```

```
Pilih menu: 1

Pilih DPA (1-3):
1
NIM   : 1313113
Nama  : apis
Prodi : ti
Kelas : 1e
apis berhasil masuk ke antrian
```

```
Pilih menu: 2

Pilih DPA (1-3):
1
Proses KRS untuk DPA 1
Memproses KRS mahasiswa:
12313 - athaulla - ti - 1e
Memproses KRS mahasiswa:
123131 - hafizh - ti - 1e
```

```
Pilih menu: 3

Pilih DPA (1-3):
1
Antrian DPA 1 tidak kosong

=== Sistem Antrian Persetujuan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Proses KRS (2 Mahasiswa)
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Semua Antrian
6. Tampilkan 2 Antrian Terdepan
7. Tampilkan Antrian Paling Akhir
8. Tampilkan Jumlah Antrian
9. Tampilkan Jumlah Mahasiswa yang Sudah Proses KRS
10. Tampilkan Jumlah Mahasiswa yang Belum Proses KRS
11. Kosongkan Antrian
0. Keluar
---------------------------------
Pilih menu: 4
```

```
Pilih DPA (1-3):
1
Antrian DPA 1 tidak penuh

=== Sistem Antrian Persetujuan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Proses KRS (2 Mahasiswa)
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Semua Antrian
6. Tampilkan 2 Antrian Terdepan
7. Tampilkan Antrian Paling Akhir
8. Tampilkan Jumlah Antrian
9. Tampilkan Jumlah Mahasiswa yang Sudah Proses KRS
10. Tampilkan Jumlah Mahasiswa yang Belum Proses KRS
11. Kosongkan Antrian
0. Keluar
---------------------------------
Pilih menu: 5

Pilih DPA (1-3):
1
Antrian DPA 1:
Daftar Mahasiswa dalam antrian:
NO | NIM - NAMA - PRODI - KELAS
1. 12311312315r - athaulla hafizh - ti - 1e
2. 1313113 - apis - ti - 1e
```

```
Pilih menu: 6

Pilih DPA (1-3):
1
Dua Antrian Terdepan DPA 1:
Dua Mahasiswa terdepan dalam antrian:
NIM - NAMA - PRODI - KELAS
1. 12311312315r - athaulla hafizh - ti - 1e
2. 1313113 - apis - ti - 1e
```

```
Pilih menu: 7

Pilih DPA (1-3):
1
Antrian Terakhir DPA 1:
Mahasiswa paling belakang dalam antrian:
NIM - NAMA - PRODI - KELAS
1313113 - apis - ti - 1e

=== Sistem Antrian Persetujuan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Proses KRS (2 Mahasiswa)
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Semua Antrian
6. Tampilkan 2 Antrian Terdepan
7. Tampilkan Antrian Paling Akhir
8. Tampilkan Jumlah Antrian
9. Tampilkan Jumlah Mahasiswa yang Sudah Proses KRS
10. Tampilkan Jumlah Mahasiswa yang Belum Proses KRS
11. Kosongkan Antrian
0. Keluar
---------------------------------
Pilih menu: 8

Pilih DPA (1-3):
1
Jumlah antrian DPA 1: 2 mahasiswa
```

```
Pilih menu: 9

Pilih DPA (1-3):
1
Jumlah mahasiswa yang sudah proses KRS di DPA 1: 2

=== Sistem Antrian Persetujuan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Proses KRS (2 Mahasiswa)
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Semua Antrian
6. Tampilkan 2 Antrian Terdepan
7. Tampilkan Antrian Paling Akhir
8. Tampilkan Jumlah Antrian
9. Tampilkan Jumlah Mahasiswa yang Sudah Proses KRS
10. Tampilkan Jumlah Mahasiswa yang Belum Proses KRS
11. Kosongkan Antrian
0. Keluar
---------------------------------
Pilih menu: 10

Pilih DPA (1-3):
1
Jumlah mahasiswa yang belum proses KRS di DPA 1: 2
```

```
Pilih menu: 11

Pilih DPA (1-3):
1
Antrian berhasil dikosongkan

=== Sistem Antrian Persetujuan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Proses KRS (2 Mahasiswa)
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Semua Antrian
6. Tampilkan 2 Antrian Terdepan
7. Tampilkan Antrian Paling Akhir
8. Tampilkan Jumlah Antrian
9. Tampilkan Jumlah Mahasiswa yang Sudah Proses KRS
10. Tampilkan Jumlah Mahasiswa yang Belum Proses KRS
11. Kosongkan Antrian
0. Keluar
----------------------------------
Pilih menu: 5

Pilih DPA (1-3):
1
Antrian DPA 1:
Antrian kosong.
```

```
=== Sistem Antrian Persetujuan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Proses KRS (2 Mahasiswa)
3. Cek Antrian Kosong
4. Cek Antrian Penuh
5. Tampilkan Semua Antrian
6. Tampilkan 2 Antrian Terdepan
7. Tampilkan Antrian Paling Akhir
8. Tampilkan Jumlah Antrian
9. Tampilkan Jumlah Mahasiswa yang Sudah Proses KRS
10. Tampilkan Jumlah Mahasiswa yang Belum Proses KRS
11. Kosongkan Antrian
0. Keluar
----------------------------------
Pilih menu: 0
Terima kasih telah menggunakan program antrian KRS.
```