# JOBSHEET 9

# LAPORAN HASIL PRAKTIKUM

# ALGORITMA DAN STRUKTUR DATA



MUHAMMAD DAFFI FIROS ZAIDAN 244107020182

TI 1E

# PROGRAM STUDI D-IV TEKNIK INFORMATIKA JURUSAN TEKNOLOGI INFORMASI

# POLITEKNIK NEGERI MALANG

# 2025

**Percobaan 1: Mahasiswa Mengumpulkan Tugas**

Kode Program

Class Mahasiswa

public class Queue17 {

    int [] data;

    int front, rear, size, max;

    public Queue17 (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;

    }

}

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;

    }

}

Class QueueMain17.java

import java.util.Scanner;

public class QueueMain17 {

    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();

        Queue17 Q = new Queue17(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;

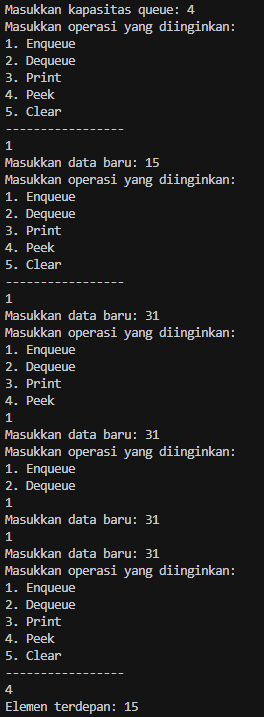
            }

        } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);

    }

}

Output



**Pertanyaan**

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

Jawaban: Nilai awal: size = 0 karena antrean kosong. front = -1 dan rear = -1 adalah penanda bahwa antrean belum memiliki elemen, belum ada elemen pertama (front), dan belum ada posisi untuk menambahkan elemen berikutnya (rear).

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



Jawaban: Potongan kode ini dipakai untuk mengatur agar indeks belakang (rear) bisa kembali ke awal array kalau sudah sampai di akhir (max - 1). Jadi kalau rear sudah di ujung array, dia bakal di-set ke 0 supaya ruang kosong di depan array bisa dipakai lagi. Ini penting supaya antrean bisa terus berputar (circular queue) dan gak buang-buang ruang. Tapi pengaturan ini cuma dilakukan kalau antrean belum penuh.

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



Jawaban: Potongan kode ini digunakan untuk mengatur agar nilai front kembali ke posisi awal array (indeks 0) jika front sudah sampai di akhir array (yaitu max - 1). Hal ini dilakukan supaya antrean tetap bisa berjalan dengan baik, karena bisa saja masih ada data yang perlu diakses di bagian awal array. Pengaturan ini dilakukan selama antrean tidak kosong.

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

Jawaban:Perulangan dimulai dari front karena front adalah indeks elemen pertama yang valid dalam antrean. Ini memastikan perulangan hanya mencetak elemen-elemen yang *saat ini* ada di dalam antrean, bukan seluruh isi array termasuk slot kosong di awal (jika ada).

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



Jawaban: Potongan kode ini digunakan untuk memindahkan indeks i ke posisi selanjutnya dalam array. Operator modulo (% max) digunakan agar jika i + 1 sudah mencapai jumlah maksimal (max), maka indeks akan kembali ke 0. Dengan begitu, pergerakan indeks bisa kembali ke awal array dan bisa digunakan terus-menerus secara berulang.

1. Tunjukkan potongan kode program yang merupakan queue overflow!

public void Enqueue (int dt) {

if (isFull()) {

System.out.println("Queue sudah penuh");

}

}

1. 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

Jawaban:

public void **Enqueue** (int *dt*) {

        if (isFull()) {

            System.out.println("Queue penuh");

            System.out.println("Program dihentikan karena queue overflow!");

            System.exit(1);

        } else {

            if (isEmpty()) {

                front = rear = 0;

            } else {

                if (rear == max -1) {

                    rear = 0;

                } else {

                    rear++;

                }

            }

            data[rear] = *dt*;

            size++;

            System.out.println(*dt* + "Berhasil masuk ke antrian");

        }

    }

**Percobaan 2: Antrian Layanan Akademil**

Kode Program

Mahasiswa

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);

    }

}

AntrianLayanan

public class AntrianLayanan17 {

    Mahasiswa[] data;

    int front;

    int rear;

    int size;

    int max;

    public AntrianLayanan17(int *max*) {

        this.max = *max*;

        this.data = new Mahasiswa[*max*];

        this.front = 0;

        this.rear = -1;

        this.size = 0;

    }

    public boolean **IsEmpty**() {

        return size == 0;

    }

    public boolean **IsFull**() {

        return size == max;

    }

    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 **LihatAkhir**() {

        if (IsEmpty()) {

            System.out.println("Antrian kosong.");

        } else {

            System.out.print("Mahasiswa paling belakang: ");

            System.out.println("NIM - NAMA - PRODI - KELAS");

            data[rear].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;

    }

}

LayananAkademikSIAKAD

import java.util.Scanner;

public class LayananAkademikSIAKAD17 {

    public static void **main**(String[] *args*) {

        Scanner sc = new Scanner(System.in);

        AntrianLayanan17 antrian = new AntrianLayanan17(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("6. Cek Antrian 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   : ");

                    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 6:

                    antrian.LihatAkhir();

                    break;

                case 0:

                    System.out.println("Terima kasih.");

                    break;

                default:

                    System.out.println("Pilihan tidak valid.");

            }

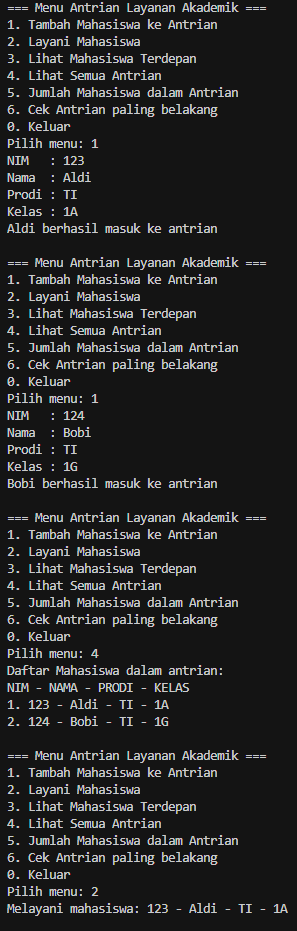
        } while (pilihan != 0);

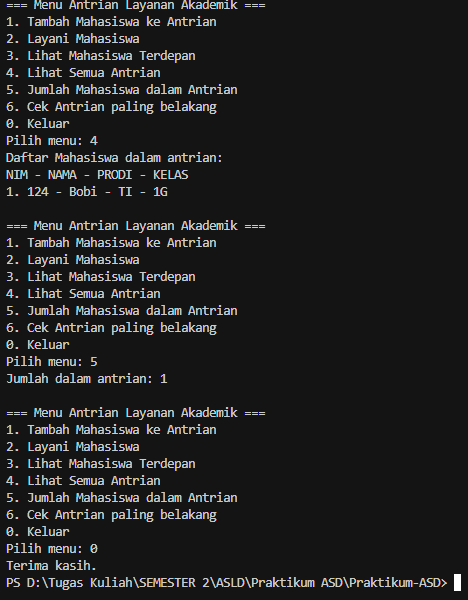
        sc.close();

    }

}

Output





**Pertanyaan**

1. 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!

**Latihan Praktikum**

Kode Program

Class Mahasiswa

package Tugas;

public class Mahasiswa17 {

    String nim, nama, prodi, kelas;

    public Mahasiswa17(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: " + nim);

        System.out.println("Nama: " + nama);

        System.out.println("Prodi: " + prodi);

        System.out.println("Kelas: " + kelas);

        System.out.println("--------------------");

    }

}

Class LayananKRS17

package Tugas;

import java.util.Scanner;

public class LayananKRS17 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        AntrianKRS17 antrian = new AntrianKRS17(10);

        int pilihan;

        do {

            System.out.println("\n=== Menu Antrian KRS Mahasiswa ===");

            System.out.println("1. Tambah Antrian");

            System.out.println("2. Proses 2 Mahasiswa");

            System.out.println("3. Lihat 2 Terdepan");

            System.out.println("4. Lihat Semua Antrian");

            System.out.println("5. Lihat Antrian Terakhir");

            System.out.println("6. Cetak Jumlah Antrian");

            System.out.println("7. Cetak Jumlah yang Sudah Proses KRS");

            System.out.println("8. Lihat Sisa Mahasiswa yang Belum Diproses");

            System.out.println("9. Kosongkan Antrian");

            System.out.println("0. Keluar");

            System.out.print("Pilih menu: ");

            pilihan = sc.nextInt();

            sc.nextLine();

            switch (pilihan) {

                case 1:

                    if (!antrian.isFull()) {

                        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();

                        Mahasiswa17 mhs = new Mahasiswa17(nim, nama, prodi, kelas);

                        antrian.tambahAntrian(mhs);

                    } else {

                        System.out.println("Antrian penuh!");

                    }

                    break;

                case 2:

                    antrian.prosesKRS();

                    break;

                case 3:

                    antrian.lihatDuaTerdepan();

                    break;

                case 4:

                    antrian.tampilkanSemua();

                    break;

                case 5:

                    antrian.lihatAkhir();

                    break;

                case 6:

                    antrian.jumlahAntrian();

                    break;

                case 7:

                    antrian.jumlahDiproses();

                    break;

                case 8:

                    antrian.jumlahBelumDiproses();

                    break;

                case 9:

                    antrian.clear();

                    break;

                case 0:

                    System.out.println("Terima kasih!");

                    break;

                default:

                    System.out.println("Pilihan tidak valid.");

            }

        } while (pilihan != 0);

        sc.close();

    }

}

Class AntrianKRS17

package Tugas;

public class AntrianKRS17 {

    Mahasiswa17[] data;

    int front, rear, size, max;

    int totalDiproses = 0;

    public AntrianKRS17(int max) {

        this.max = max;

        data = new Mahasiswa17[max];

        front = 0;

        rear = -1;

        size = 0;

    }

    public boolean isEmpty() {

        return size == 0;

    }

    public boolean isFull() {

        return size == max;

    }

    public void clear() {

        front = 0;

        rear = -1;

        size = 0;

        totalDiproses = 0;

        System.out.println("Antrian dikosongkan.");

    }

    public void tambahAntrian(Mahasiswa17 mhs) {

        if (isFull()) {

            System.out.println("Antrian penuh, tidak bisa menambah mahasiswa.");

            return;

        }

        rear = (rear + 1) % max;

        data[rear] = mhs;

        size++;

        System.out.println(mhs.nama + " berhasil ditambahkan ke antrian.");

    }

    public void prosesKRS() {

        if (size < 2) {

            System.out.println("Antrian kurang dari 2 mahasiswa. Tidak bisa proses.");

            return;

        }

        System.out.println("Memproses 2 mahasiswa:");

        for (int i = 0; i < 2; i++) {

            Mahasiswa17 mhs = data[front];

            mhs.tampilkanData();

            front = (front + 1) % max;

            size--;

            totalDiproses++;

        }

    }

    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 index = (front + i) % max;

            System.out.print((i + 1) + ". ");

            data[index].tampilkanData();

        }

    }

    public void lihatDuaTerdepan() {

        if (size < 2) {

            System.out.println("Kurang dari dua mahasiswa dalam antrian.");

            return;

        }

        System.out.println("2 Mahasiswa Terdepan:");

        for (int i = 0; i < 2; i++) {

            int index = (front + i) % max;

            data[index].tampilkanData();

        }

    }

    public void lihatAkhir() {

        if (isEmpty()) {

            System.out.println("Antrian kosong.");

        } else {

            System.out.println("Mahasiswa terakhir dalam antrian:");

            data[rear].tampilkanData();

        }

    }

    public void jumlahAntrian() {

        System.out.println("Jumlah mahasiswa dalam antrian: " + size);

    }

    public void jumlahDiproses() {

        System.out.println("Jumlah mahasiswa yang sudah melakukan proses KRS: " + totalDiproses);

    }

    public void jumlahBelumDiproses() {

        int sisa = 30 - totalDiproses;

        System.out.println("Sisa mahasiswa yang belum melakukan proses KRS oleh DPA: " + (sisa > 0 ? sisa : 0));

    }

}

public void lihatAkhir() {

        if (isEmpty()) {

            System.out.println("Antrian kosong.");

        } else {

            System.out.println("Mahasiswa terakhir dalam antrian:");

            data[rear].tampilkanData();

        }

    }

    public void jumlahAntrian() {

        System.out.println("Jumlah mahasiswa dalam antrian: " + size);

    }

    public void jumlahDiproses() {

        System.out.println("Jumlah mahasiswa yang sudah melakukan proses KRS: " + totalDiproses);

    }

    public void jumlahBelumDiproses() {

        int sisa = 30 - totalDiproses;

        System.out.println("Sisa mahasiswa yang belum melakukan proses KRS oleh DPA: " + (sisa > 0 ? sisa : 0));

    }

}

Diagram

