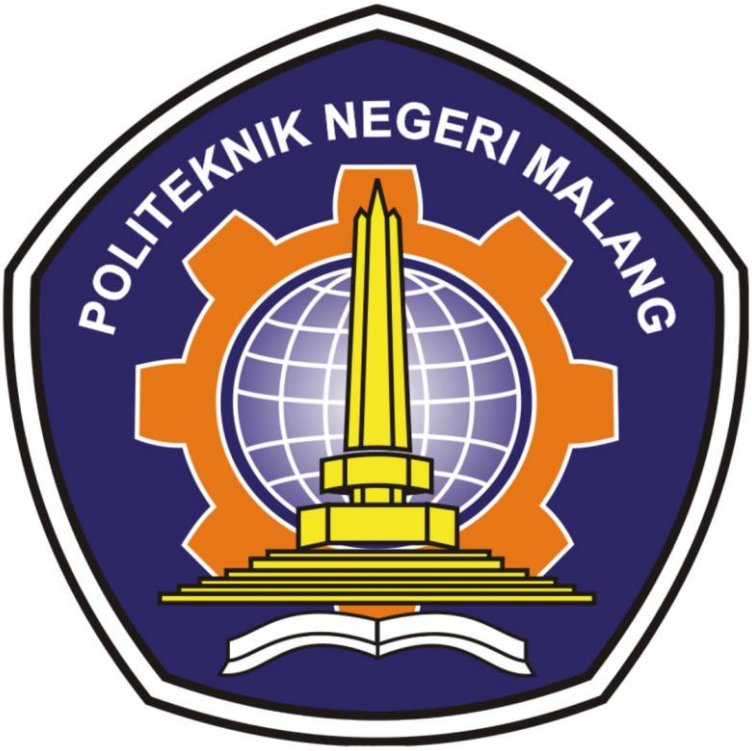
LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN STRUKTUR DATA
JOBSHEET 10



RIFO ANGGI BARBARA DANUARTA
244107020063
TI_1E
PROGRAM STUDI D_IV TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG

Percobaan 1

1. Buat folder baru bernama P1Jobsheet10 di dalam repository Praktikum ASD, kemudian buat class baru dengan nama Queue.
2. Tambahkan atribut-atribut Queue sesuai diagram class, kemudian tambahkan pula konstruktornya seperti berikut ini :

```
int[] data;
int front;
int rear;
int size;
int max;

public Queue26(int n) {
    max = n;
    data = new int[max];
    size = 0;
    front = rear = -1;
}
```

3. Buat method IsEmpty bertipe boolean yang digunakan untuk mengecek apakah queue kosong.

```
public boolean isEmpty() {
    if (size == 0) {
        return true;
    } else {
        return false;
    }
}
```

4. Buat method IsFull bertipe boolean yang digunakan untuk mengecek apakah queue sudah penuh.

```
public boolean isFull() {
    if (size == max) {
        return true;
    } else {
        return false;
    }
}
```

5. Buat method peek bertipe void untuk menampilkan elemen queue pada posisi paling depan.

```
public void peek() {
    if (!isEmpty()) {
        System.out.println("Elemen terdepan: " + data[front]);
    } else {
        System.out.println("Queue masih kosong");
    }
}
```

6. Buat method print bertipe void untuk menampilkan seluruh elemen pada queue mulai dari posisi front sampai dengan posisi rear.

```
    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);
        }
    }
```

7. Buat method clear bertipe void untuk menghapus semua elemen pada queue

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

8. Buat method Enqueue bertipe void untuk menambahkan isi queue dengan parameter dt yang bertipe integer

```
public void Engueue(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++;
        }
    }
```

9. Buat method Dequeue bertipe int untuk mengeluarkan data pada queue di posisi belakang

```
    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;
    }
}
```

10. Selanjutnya, buat class baru dengan nama QueueMain tetap pada package Praktikum1. Buat method menu bertipe void untuk memilih menu program pada saat dijalankan.

```
    public static void menu() {
        System.out.println("Masukan 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("========================");
```

11. Buat fungsi main, kemudian deklarasikan Scanner dengan nama sc.

```
        Scanner sc = new Scanner(System.in);
```

12. Buat variabel n untuk menampung masukan berupa jumlah maksimal elemen yang dapat disimpan pada queue.

```
        System.out.print("Masukan kapasitas queue: ");
        int n = sc.nextInt();
```

13. Lakukan instansiasi objek Queue dengan nama Q dengan mengirimkan parameter n sebagai kapasitas elemen queue

```
        Queue26 Q = new Queue26(n);
```

14. Deklarasikan variabel dengan nama pilih bertipe integer untuk menampung pilih menu dari pengguna.

```
        int pilih;
```

15. Lakukan perulangan menggunakan do-while untuk menjalankan program secara terus menerus sesuai masukan yang diberikan. Di dalam perulangan tersebut, terdapat pemilihan kondisi menggunakan switch-case untuk menjalankan operasi queue sesuai dengan masukan pengguna.

```
        do {
            menu();
            pilih = sc.nextInt();
            switch (pilih) {
                case 1:
                    System.out.print("Masukan data baru: ");
                    int dataMasuk = sc.nextInt();
                    Q.Engueue(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("Pilihan tidak valid");
            }
        } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih
== 4 || pilih == 5);
    }
}
```

16. Compile dan jalankan class QueueMain, kemudian amati hasilnya.

```
Masukan kapasitas queue: 4
Masukan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit
========================
1
Masukan data baru: 15
Masukan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit
========================
1
Masukan data baru: 31
Masukan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit
========================
4
Elemen terdepan: 15
```

Pertanyaan

1. Pada konstruktor, mengapa nilai awal atribut front dan rear bernilai -1, sementara atribut size bernilai 0?
2. Pada method Enqueue, jelaskan maksud dan kegunaan dari potongan kode berikut!
3. Pada method Dequeue, jelaskan maksud dan kegunaan dari potongan kode berikut!
4. Pada method print, mengapa pada proses perulangan variabel i tidak dimulai dari 0 (int i=0), melainkan int i=front?
5. Perhatikan kembali method print, jelaskan maksud dari potongan kode berikut!
6. Tunjukkan potongan kode program yang merupakan queue overflow!
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!

Jawaban

1. Nilai -1 dipilih karena indeks ini tidak mungkin ada dalam array. Ini menjadi penanda jelas bahwa queue belum berisi data apa pun.
   Saat queue benar-benar kosong, front dan rear tidak menunjuk ke lokasi valid dalam array. Ketika elemen pertama dimasukkan, keduanya akan menunjuk ke indeks 0, menandakan queue sekarang berisi satu elemen. jadi intinya jika front dan rear bernilai 0 maka array akan ada isinya padahan belum di isi data sama sekali jadi jika bernilai -1 array memang belum ada isinya dan ini lebih aman.
2. Saat pointer rear sudah berada di posisi paling akhir array (indeks max - 1), pointer tersebut akan "melompat" kembali ke awal array (indeks 0), jika posisi tersebut kosong.max - 1 adalah indeks terakhir dari array.Ketika rear == max - 1, itu artinya pointer sudah mencapai ujung.Supaya tidak membuang ruang, kita atur rear = 0 agar bisa memulai lagi dari depan.
3. Saat kita melakukan dequeue, pointer front akan maju ke elemen berikutnya. Nah, kalau front sudah ada di posisi paling akhir array (yaitu indeks max - 1), maka kita harus "melompat" balik ke awal array, yaitu indeks 0. Jadi, supaya antrian tetap bisa menggunakan ruang kosong di awal array (yang mungkin sudah tidak terpakai karena elemen sebelumnya sudah dikeluarkan), kita atur front kembali ke 0.
4. Karena dalam circular queue, elemen pertama (yang paling depan dalam antrian) bisa jadi bukan di indeks 0. Bisa aja elemen paling depan ada di indeks 2, 4, atau bahkan 9, tergantung dari berapa kali dequeue() sudah dilakukan sebelumnya.Jadi, kalau kita mulai perulangan dari i = 0, kita justru akan mencetak elemen yang sudah lama dikeluarkan atau bahkan kosong. Supaya yang dicetak adalah isi antrian yang masih aktif, perulangan harus dimulai dari front, yaitu posisi elemen pertama yang masih ada dalam antrian.
5. Setiap kali i bertambah satu, kita pakai % max supaya nilainya tetap berada dalam batas indeks array.Normalnya, kalau i = 4 dan max = 5, maka i + 1 = 5, tapi indeks array cuma sampai 4 Nah, 5 % 5 = 0, jadi i balik lagi ke indeks 0.Dengan cara ini, perulangan bisa terus jalan dari ujung array ke awal array tanpa keluar dari batas. jadi intinya Kode ini bikin antrian bisa muter balik ke depan lagi setelah sampai di akhir array.
6. Potongan ini

```
    if (isFull()) {
        System.out.println("Queue sudah penuh");
```

   adalah pengecekan queue overflow. Artinya, kalau antrian sudah penuh, maka program akan mencetak Queue sudah penuh.
7. Method Engueue

```
    if (isFull()) {
        System.err.println("Queue sudah penuh");
        System.exit(1);
```

   Method Degueue

```
        if (isEmpty()) {
            System.err.println("Queue masih kosong");
            System.exit(1);
```

Percobaan 2

1. Buat folder baru bernama P2Jobsheet10 di dalam repository Praktikum ASD, kemudian buat class baru dengan nama Mahasiswa.
2. Tambahkan atribut-atribut Nasabah seperti pada Class Diagram, kemudian tambahkan pula konstruktornya seperti gambar berikut ini.

```
    public Mahasiswa26(String nim, String nama, String prodi,
String kelas) {
        this.nim = nim;
        this.nama = nama;
        this.prodi = prodi;
        this.kelas = kelas;
    }
```

Dan tambahkan method tampilkanData berikut :

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

3. Salin kode program class Queue pada Praktikum 1 untuk digunakan kembali pada Praktikum 2 ini, ganti nama class-nya dengan AntrianLayanan. Karena pada Praktikum 1, data yang disimpan pada queue hanya berupa array bertipe integer, sedangkan pada Praktikum 2 data yang digunakan adalah object, maka perlu dilakukan modifikasi pada class AntrianLayanan tersebut.

```
    Mahasiswa26[] data;
    int front;
    int rear;
    int size;
    int max;

    public AntrianLayanan26(int max) {
        this.max = max;
        this.data = new Mahasiswa26[max];
        this.front = 0;
        this.rear = -1;
        this.size = 0;
    }
```

4. Lakukan modifikasi pada class AntrianLayanan dengan mengubah tipe int[] data menjadi Mahasiswa[] data karena pada kasus ini data yang akan disimpan berupa object Mahasiswa. Modifikasi perlu dilakukan pada atribut, method Enqueue, dan method Dequeue.

```
    public void tambahAntrian(Mahasiswa26 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 masuk ke
antrian.");
    }

    public Mahasiswa26 layaniMahasiswa26() {
        if (isEmpty()) {
            System.out.println("Antrian kosong.");
            return null;
        }
        Mahasiswa26 mhs = data[front];
        front = (front + 1) % max;
        size--;
        return mhs;
    }
```

5. Berikutnya method peek dan print yaitu untuk menampilkan data antrian layanan paling depan dan menampilkan semua data antrian layanan.

```
    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();
        }
    }
```

Ditambahkan dengan method getJumlahAntrian yaitu menampilkan nilai size

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

6. Selanjutnya, buat class baru dengan nama LayananAkademikSIAKAD tetap pada package yang sama. Buat fungsi main, deklarasikan Scanner dengan nama sc.

```
public class LayananAkademikSIAKAD26 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
```

7. Kemudian lakukan instansiasi objek AntrianLayanan dengan nama antrian dan nilai parameternya adalah nilai maksimal antrian yang ditentukan (misal sama dengan 5).

```
        AntrianLayanan26 antrian = new AntrianLayanan26(5);
```

8. Deklarasikan variabel dengan nama pilihan bertipe integer untuk menampung pilih menu dari pengguna.

```
        int pilihan;
```

9. Tambahkan kode berikut untuk melakukan perulangan menu sesuai dengan masukan yang diberikan oleh pengguna.

```
        do {
            System.out.println("\n=== Menu Antrian Layanan
Akademik SIAKAD ===");
            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();
                    Mahasiswa26 mhs = new Mahasiswa26(nim, nama,
prodi, kelas);
                    antrian.tambahAntrian(mhs);
                    break;
```

```
                case 2:
                    Mahasiswa26 dilayani =
antrian.layaniMahasiswa26();
                    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();
    }
```

10. Compile dan jalankan class LayananAkademikSIAKAD, kemudian amati hasilnya

```
=== Menu Antrian Layanan Akademik SIAKAD ===
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   : 123
Nama  : Aldi
Prodi : TI
Kelas : 1A
Aldi berhasil masuk ke antrian.

=== Menu Antrian Layanan Akademik SIAKAD ===
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   : 124
Nama  : Bobi
Prodi : TI
Kelas : 1G
Bobi berhasil masuk ke antrian.
```

```
=== Menu Antrian Layanan Akademik SIAKAD ===
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: 4
Daftar Mahasiswa dalam antrian:
NIM - NAMA - PRODI - KELAS
1. 123 - Aldi - TI - 1A
2. 124 - Bobi - TI - 1G

=== Menu Antrian Layanan Akademik SIAKAD ===
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: 2
Melayani Mahasiswa: 123 - Aldi - TI - 1A

=== Menu Antrian Layanan Akademik SIAKAD ===
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: 4
Daftar Mahasiswa dalam antrian:
NIM - NAMA - PRODI - KELAS
1. 124 - Bobi - TI - 1G
```

```
=== Menu Antrian Layanan Akademik SIAKAD ===
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: 5
Jumlah dalam antrian: 1

=== Menu Antrian Layanan Akademik SIAKAD ===
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: 0
```

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!

method LihatAkhir

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

daftar menu

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

pemanggilan method LihatAkhir

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

Tugas

```
package Jobsheet10;

public class DataAntrianKRS26 {
    String nim;
    String nama;
    String prodi;
    String kelas;

    public DataAntrianKRS26(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);
    }
}
```

```
package Jobsheet10;

public class AntrianKRS26 {
    DataAntrianKRS26[] data;
    int front;
    int rear;
    int size;
    int max;

    public AntrianKRS26(int max) {
        this.max = max;
        this.data = new DataAntrianKRS26[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(DataAntrianKRS26 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 masuk ke antrian.");
    }

    public DataAntrianKRS26 layaniMahasiswa() {
        DataAntrianKRS26 mhs = data[front];
        if (isEmpty()) {
            System.out.println("Antrian kosong.");
            return null;
        }
        for (int i = 0; i < 2; i++) {
            if (isEmpty()) {
                System.out.println("Tidak ada mahasiswa yang bisa
dilayani.");
                return null;
            }
            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 void lihatAkhir() {
        if (isEmpty()) {
            System.out.println("Antrian kosong.");
        } else {
            System.out.println("Mahasiswa terakhir: ");
            System.out.println("NIM - NAMA - PRODI - KELAS");
            data[rear].tampilkanData();
        }
    }
    public void KosongkanAntrian() {
        if (!isEmpty()) {
            front = rear = -1;
            size = 0;
            System.out.println("Antrian berhasil dikosongkan");
        } else {
            System.out.println("Antrian masih kosong");
        }
    }

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

```
package Jobsheet10;

import java.util.Scanner;

public class MainAntrianKRS26 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        AntrianKRS26 antrian = new AntrianKRS26(10);
        int mahasiswaDilayani = 0;
        int pilihan;

        do {
            System.out.println("\n=== Menu Antrian KRS ===");
            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("7. Kosongkan 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();
                    DataAntrianKRS26 mhs = new DataAntrianKRS26(nim,
nama, prodi, kelas);
                    antrian.tambahAntrian(mhs);
                    break;
                case 2:
                    DataAntrianKRS26 dilayani =
antrian.layaniMahasiswa();
                    if (dilayani != null) {
                        System.out.print("Melayani Mahasiswa: ");
                        dilayani.tampilkanData();
                    }
                    mahasiswaDilayani += 2;
                    break;
                case 3:
                    antrian.lihatTerdepan();
                    break;
                case 4:
                    antrian.tampilkanSemua();
                    break;
                case 5:
                    System.out.println("Jumlah antrian saat ini: " +
antrian.getJumlahAntrian());
                    System.out.println("Jumlah mahasiswa yang sudah
dilayani: " + mahasiswaDilayani);
                    System.out.println("Jumlah mahasiswa yang belum
melakukan KRS: " + (antrian.getJumlahAntrian() + mahasiswaDilayani));
                    break;
```

```
                case 6:
                    antrian.lihatAkhir();
                    break;
                case 7:
                    antrian.KosongkanAntrian();
                    mahasiswaDilayani = 0;
                case 0:
                    System.out.println("Terima kasih.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid.");
            }
        } while (pilihan != 0);
    }
}
```