LAPORAN PRAKTIKUM
ALGORITMA DAN STRUKTUR DATA
JOBSHEET 10

NAMA = RAKAGALI RESDA KRISANDI PUTRA
KELAS = TI -1B / 19
NIM = 244107020136

```
'4 11 pilih     5);
```

1.1.1. Verifikasi Hasil Percobaan

Samakan hasil compile kode program Anda dengan gambar berikut ini.

```
Masukan Kapasitas Queue :4
Masukan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
----------------------
1
Masukan data baru: 15
Masukan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
----------------------
1
Masukan data baru: 31
Masukan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
----------------------
4
Elemen Terdepan: 15
```

1.1.2. Pertanyaan

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

   Front dan rear bernilai -1 karna berfungsi untuk mengecek index aray mulai dari 0 jika di set 0 maka default akan berada pada index pertama yaitu 0 yang akan berpengaruh pada kode

   Sedangkan size dimulai dari 0 karna merupakan penjumlahan biasa tidak berfungsi untk mengecek array

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

```
if (rear=  max -  ) {
    rear = O;
```

   Kode ini memungkinkan queue bekerja secara melingkar, menghindari pemborosan memori dan memastikan operasi enqueue tetap bisa dilakukan jika ada ruang kosong.

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

```
if (front=  max -  ) {
    front= O;
```

   Kode ini mengecek apakah indeks front (posisi elemen yang akan dikeluarkan dari antrian) sudah berada di posisi akhir array (max - 1).
   Jika iya, maka front di-*reset* ke 0 agar menunjuk kembali ke awal array.

4. Pada method **print,** mengapa pada proses perulangan variabel i tidak dimulai dari O **(int i=O),** melainkan **int i=front?**

   Karna queue dimulai dari data terbaru berada dimana jadi tidak selalu pada index ke 0 makanya diperlukan variabel front untuk menyimpan data yang pertama masuk di index mana untuk memulau perulangan

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

   Kode ini digunakan untuk **mengiterasi indeks i secara melingkar (circular)** pada array berukuran max.

   Dengan i = (i + 1) % max, maka:

   - i akan bertambah satu.
   - Jika i telah mencapai akhir array (max - 1), hasil (i + 1) % max akan kembali ke 0.

6. Tunjukkan potongan kode program yang merupakan queue overflow!

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

```
public int Dequeue() {
    int dt = 0;
    if(isEmpty()) {
        System.out.println(x:"Queue masih kosong");
        System.exit(status:0);
```

Menambah System.exit(0);

2.2. Percobaan 2: Antrian Layanan Akademik

Waktu percobaan : **90 menit**

Pada percobaan ini, kita akan membuat program yang mengilustrasikan Layanan pada Admin Akademik. Perhatikan Diagram Class berikut ini:

Mahasiswa
nim:String nama: String prodi: String kelas: String
Mahasiswa(nim: String, nama: String, prodi: String, kelas: String) void tampilkanData()

2.2.1. Langkah-langkah Percobaan

Berdasarkan diagram class tersebut, akan dibuat program class Mahasiswa dalam Java.

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 Mahasiswa(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.

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

```
public Antrianlayanan(int max) {
    this.max= max;
    this.data = ne1-1llahasiswa[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 tambaMntrian(Mahasiswa  mhs) {
    if  (isfull()) {
        System.out.println("Antrian penuh, tidak dapat menambah mahasis1-1a.");
        return;
    }
    rear= (rear+ 1) % max;
    data[rear] = mhs;
    siz
    System.out.println(mhs.nama +" berhasil  masuk ke antrian.");
}
```

```
public Mahasiswa layaniMahasiswa(){
    if (isEmpty()) {
        System.out.println{"Antrian kosong.");
        return null;
    }
    MahasisNa 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 Mahasis1va dalam Antrian:");
    System.out.println("NIM - NAMA - PROD! - KELAS");
    for (inti= 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.**
7. Kemudian lakukan instansiasi objek **Antrianlayanan** dengan nama **antrian** dan nilai parameternya adalah nilai maksimal antrian yang ditentukan (misal sama dengan 5).
8. Deklarasikan variabel dengan nama **pilihan** bertipe integer untuk menampung pilih menu dari pengguna.

```
public class LayananAkademikSIAKAD {

    public static voidmain(String[] args) {
        Scanner sc = new Scanner(System.in);
        Antrianlayanan antrian = new Antrianlayanan(5);
        int pilihan;
```

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

```
do {

    System.out.println("\n=== enu Antrian Layanan Akademik ===");
    System.out.println("1. Tambah Mahasis1va ke Antrian");
    System.out.println("2. Layani Mahasis1va");
    System.out.println("3. Lihat Mahasis1va 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.layaniMahasis, a();
                    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();
    }
```

10. Compile dan jalankan class **LayananAkademikSIAKAD,** kemudian amati hasilnya.

2.2.2 Verifikasi Hasil Percobaan

Samakan hasil compile kode program Anda dengan gambar berikut ini.

```
=   Menu Antrian Layanan Akadernik ---
1. Tarnba  Ma asiswa ke Antrian
2. Layani Ma asiswa
3. Li at Ma asiswa Terdepan
4. Li at Sernua Antrian
5. Jumlah Mahasiswa dalarn Antrian
0. Kel ar
Pili men : 1
N IM      123
Nama      Aldi
Prodi     TI
Kelas     1A
Aldi her asil rnasuk ke antrian.
```

```
==  Me   Antrian Laya an Akademik --
1. Tamba  Ma asiswa ke Antrian
2. Layani Ma asiswa
3.  i at Mahasiswa Terdepan
4. Li at Semua Antrian
5. Jumla  Ma asiswa dalam Antrian
0. Keluar
Pili  menu: 1
 IM     124
 ama    Bobi
Prodi   TI
Kelas   lG
Bobi her asil masuk ke antria

==  Menu A trian Layanan Akademik --
1.  Tambah Ma asiswa ke Antrian
2.  aya ı  Ma asiswa
3.  i at Ma asiswa Terdepan
4.  i at Semua Antria
5. Jumla  Ma asiswa dalam Antrian
0. Keluar
Pili  men
Daftar Ma asiswa dalam Antrian:
 IM -  AMA - PRODI   KELAS
1. 123 - Aldi   TI   lA
2.  124   Bobi - TI   lG

==  Menu Antrian Laya an Akademik --
1. Tamba  Mahasiswa ke Antrian
2. Layani Mahasis\'7a
3. Li at Ma asiswa Terdepan
4. Li at Semua Antrian
5. Jumla  Ma asis\'7a dalam Antrian
0. Kel ar
Pili  menu: 2
Melayani ma asiswa: 123 - Aldi - TI - lA

==  Menu Antrian Laya an Akademik --
1. Tamba  Ma asiswa ke An rian
2. Layani Mahasiswa
3. Li at Ma asiswa Terdepan
4.  i at Semua Antria
5. Jumla  Ma asiswa dalam Antrian
0. Keluar
Pili  menu: 4
 aftar Ma asiswa dalam Antrian:
 IM -  AMA - PRODI   KELAS
1. 124 - Bobi - TI - lG
```

```
==  Me u Atria     ayanan Akademik ---
1. Tamba  Ma asiswa ke Antrian
2.  ayani Ma asiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumla  Ma asiswa dalam Antrian
0. Kel ar
Pili men : 5
Jumlah dalam a trian: 1

==  Menu Antrian Layanan Akademik
1. Tamba  Ma asiswa ke Antrian
2. Layani Ma asiswa
3. Lihat Ma asiswa Terdepan
4.  i at Sem a Antrian
5. Jumla  Ma asis,11a dalam Antria
0. Kel ar
Pili  menu: 0
Terima kasih.
```

```
 ==== Menu Antrian Layanan Akademik ====
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Liat mahasiswa terdepan
4. Lihat semua antrian
5. Jumlah mahasiswa dalam antrian
0. Keluar
Pilih menu : 1
Masukkan NIM : 123
Masukkan Nama : Aldi
Masukkan Prodi : TI
Masukan Kelas :
1A
AldiBerhasil masuk ke anteran

 ==== Menu Antrian Layanan Akademik ====
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Liat mahasiswa terdepan
4. Lihat semua antrian
5. Jumlah mahasiswa dalam antrian
0. Keluar
Pilih menu : 1
Masukkan NIM : 124
Masukkan Nama : BOBI
Masukkan Prodi : TI
Masukan Kelas :
1G
BOBIBerhasil masuk ke anteran
```

```
 ==== Menu Antrian Layanan Akademik ====
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Liat 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. Liat 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. Liat 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. Liat mahasiswa terdepan
4. Lihat semua antrian
5. Jumlah mahasiswa dalam antrian
0. Keluar
Pilih menu : 5
Jumlah mahasiswa dalam antrian : 1

 ==== Menu Antrian Layanan Akademik ====
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Liat mahasiswa terdepan
4. Lihat semua antrian
5. Jumlah mahasiswa dalam antrian
0. Keluar
Pilih menu : 0
```

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 !

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

2.3 Tugas

Waktu : 120 Menit

Buatlah program antrian untuk mengilustasikan antrian persetujuan Kartu Rencana Studi (KRS) Mahasiswa oleh Dasen Pembina Akademik (DPA). Ketika seorang mahasiswa akan mengantri, maka dia harus mendaftarkan datanya (data mahasiswa seperti pada praktikum 2). Gunakan class untuk antrian seperti pada Praktikum 1 dan 2, dengan method-method yang berfungsi :

- Cek antrian kosong, Cek antrian penuh, Mengosongkan antrian.
- Menambahkan antrian, Memanggil antrian untuk proses KRS - setiap lx panggilan terdiri dari 2 mahasiswa (pada antrian no 1 dan 2)
- Menampilkan semua antrian, Menampilkan 2 antrian terdepan, Menampilkan antrian paling akhir.
- Cetak jumlah antrian, Cetak jumlah yang sudah melakukan proses KRS
- Jumlah antrian maximal 10, jumlah yang ditangani masing-masing DPA 30 mahasiswa, cetak jumlah mahasiswa yang belum melakukan proses KRS.

Gambarkan **Diagram Class** untuk antriannya. Implementasikan semua method menggunakan menu pilihan pada fungsi **main**.

Mahasiswa
Nim : string Nama : string Prodi : string Kelas: String

Mahasiswa(nim, nama, prodi, kelas) tampilkanData() : void

AntrianKRS
Data : Mahasiswa[] Front : int Rear :int Size : int Max: int sudahKRS : int
AntrianKRS() isFull(): boolean isEmpty(): boolean kosongkan(): void tambahAntrian(m: Mahasiswa): void prosesKRS(): void tampilkanSemua(): void tampilkanDuaTerdepan(): void lihatAkhir(): void getJumlahAntrian(): int getSudahKRS(): int getBelumKRS(): int