

LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR DATA
JOBSHEET QUEUE



VANESA MARDIANA PUTRI

244107020129

KELAS TI-1B

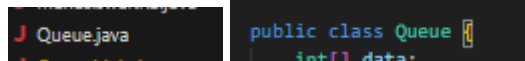
PRODI D-IV TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG

A. PERCOBAAN

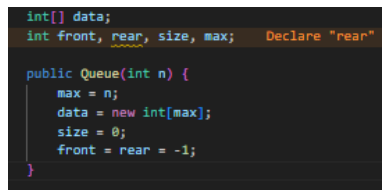
1. Percobaan 1 : Operasi Dasar Queue

Langkah - langkah :

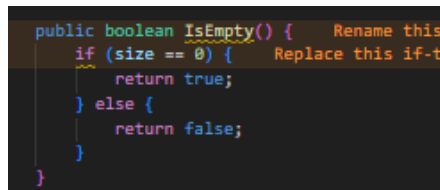
1. Buat class baru dengan nama Queue.



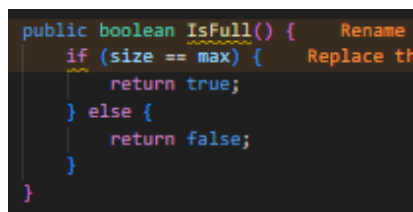
2. Tambahkan atribut-atribut Queue sesuai diagram class, kemudian tambahkan pula konstruktornya seperti berikut ini :



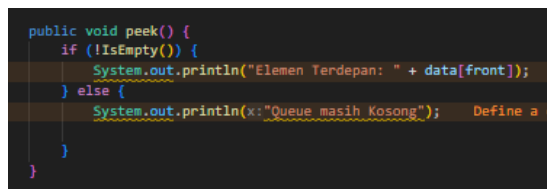
3. Buat method IsEmpty bertipe boolean yang digunakan untuk mengecek apakah queue kosong.



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



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



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(x:"Queue Masih Kosong"); Replace this use
    } else {
        int i = front;
        while (i != rear) {
            System.out.print(data[i] + " "); Replace this use of Sys
            i = (i + 1) % max;
        }
        System.out.println(data[i] + " "); Replace this use of System
        System.out.println("Jumlah elemen: " + size); Replace this u
    }
}

```

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

```

public void clear() {
    if (!IsEmpty()) {
        front = rear = size = -1;
        size = 0;
        System.out.println(x:"Queue berhasil dikosongkan"); Replac
    } else {
        System.out.println(x:"Queue masih Kosong"); Replace this u
    }
}

```

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

```

public void Enqueue(int dt) { Rename this method name t
    if (IsFull()) {
        System.out.println(x:"Queue sudah penuh"); Rep
    } 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(x:"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 class QueueMain {
    public static void menu() {
        System.out.println(x:"Masukkan operasi yang diinginkan: ");
        System.out.println(x:"1. Enqueue");
        System.out.println(x:"2. Dequeue");
        System.out.println(x:"3. Print");
        System.out.println(x:"4. Peek");
        System.out.println(x:"5. Clear");
        System.out.println(x:"-----");
    }
}

```

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

```

Run | Debug
public static void main(String[] args) {
    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(s:"Masukkan kapasitas queue: ");
int n = sc.nextInt();

```

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

```

Queue Q = new Queue(n);

```

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

```

int pilih;
do {

```

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) {    Add a default case to this switch.  
        case 1:  
            System.out.print(s:"Masukkan data baru: ");    Replace this use of  
            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);  
}
```

16. Hasil run :

```

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

```

Jawaban pertanyaan :

1. Karena menghitung rear dan front dari index ke-0 sementara size menggunakan data yang ada.
2. Jika pointer rear sudah mencapai ujung array ($\text{max} - 1$), maka akan kembali ke awal array (indeks 0).
3. Mengecek apakah pointer front (penunjuk elemen pertama) berada di indeks terakhir array ($\text{max} - 1$). Jika front sudah di indeks terakhir, reset front ke 0 (indeks pertama array).
4. Karena belum tentu front berada pada indeks ke-0.
5. Untuk mengembalikan indeks ke awal array apabila sudah mencapai akhir.

6.

```

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

```

7. Yang dimodifikasi :

```

public void Enqueue(int dt) {
    if (IsFull()) {
        System.out.println("Queue sudah penuh");
        System.exit(status:1);
    } else {
        if (IsEmpty()) {

```

```

public int Dequeue() {
    int dt = 0;
    if (IsEmpty()) {
        System.out.println("Queue masih Kosong");
        System.exit(status:1);
    }
}

```

2. Percobaan 2 : Antrian Layanan Akademik

Langkah - langkah :

1. Buat class baru dengan nama Mahasiswa.

```

public class Mahasiswa {
    String nim;
    String nama;
    String prodi;
    String kelas;
}

```

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.

```

public class AntrianLayanan {
    Mahasiswa[] data;
    int front, rear, size, max;
    public AntrianLayanan(int max) {
        this.max = max;
        this.data = new Mahasiswa[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(Mahasiswa mhs) {
    if (isFull()) {
        System.out.println(x:"Antrian penuh, tidak dapat menambahkan mah");
        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(x:"Antrian kosong.");
        return null;
    }
    Mahasiswa 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(x:"Antrian kosong.");
    } else {
        System.out.println(x:"Mahasiswa terdepan: ");
        System.out.println(x:"NIM - NAMA - PRODI - KELAS");
        data[front].tampilkanData();
    }
}

public void tampilkanSemua() {
    if (isEmpty()) {
        System.out.println(x:"Antrian kosong.");
        return;
    }
    System.out.println(x:"Daftar Mahasiswa dalam Antrian");
    System.out.println(x:"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 LayananAkademikSIKAD tetap pada package yang sama. Buat fungsi main, deklarasi Scanner dengan nama sc.

```
Run | Debug  
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). Deklarasi variabel dengan nama pilihan bertipe integer untuk menampung pilih menu dari pengguna.

```
Scanner sc = new Scanner(System.in);  
AntrianLayanan antrian = new AntrianLayanan(max:5);  
int pilihan;
```

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

```
do {  
    System.out.println(x:"\n== Menu Antrian Layanan Akademik ==");  
    System.out.println(x:"1. Tambah Mahasiswa ke Antrian");  
    System.out.println(x:"2. Layani Mahasiswa");  
    System.out.println(x:"3. Lihat Mahasiswa Terdepan");  
    System.out.println(x:"4. Lihat Semua Antrian");  
    System.out.println(x:"5. Jumlah Mahasiswa dalam Antrian");  
    System.out.println(x:"0. Keluar");  
    System.out.println(x:"Pilihan menu: ");  
    pilihan = sc.nextInt(); sc.nextLine();  
  
    switch (pilihan) {  
        case 1:  
            System.out.print(s:"NIM : ");  
            String nim = sc.nextLine();  
            System.out.print(s:"Nama : ");  
            String nama = sc.nextLine();  
            System.out.print(s:"Prodi : ");  
            String prodi = sc.nextLine();  
            System.out.print(s:"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(s:"Melayani Mahasiswa: ");  
                dilayani.tampilkanData();  
            }  
            break;  
    }  
}
```

```

        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(x:"Terima kasih.");
        break;
    default:
        System.out.println(x:"Pilihan tidak valid.");
    }
} while (pilihan != 0);
}

```

9. Compile dan jalankan class LayananAkademikSIKAD, kemudian amati hasilnya.

```

=== 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
Pilihan 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
Pilihan 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
Pilihan 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
Pilihan 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
Pilihan 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
Pilihan menu:
0
Terima kasih.

```

Jawaban pertanyaan :

```

        case 6:
            Mahasiswa terakhir = antrian.lihatAkhir();
            if (terakhir != null) {
                System.out.println("Mahasiswa di belakang antrian: " + terakhir.nama);
            }
            break;
        case 0:

```

```

    public Mahasiswa lihatAkhir() {
        if (isEmpty()) {
            System.out.println(x:"Antrian kosong!");
            return null;
        }
        return data[rear];
    }

```

B. TUGAS

Main :

```

J LayananKRS.java > ...
1 package jobsheet10;
2 import java.util.Scanner;
3
4 public class LayananKRS {
5
6     public static void main(String[] args) {
7         Scanner sc = new Scanner(System.in);
8         AntrianKRS antrian = new AntrianKRS();
9         int pilihan;
10
11         do {
12             System.out.println(x:"\n=== Menu Antrian Layanan KRS ===");
13             System.out.println(x:"1. Tambah Mahasiswa ke Antrian");
14             System.out.println(x:"2. Layani Mahasiswa");
15             System.out.println(x:"3. Tampilkan Antrian");
16             System.out.println(x:"4. Tampilkan 2 Mahasiswa Terdepan");
17             System.out.println(x:"5. Tampilkan Mahasiswa Terakhir");
18             System.out.println(x:"6. Jumlah Mahasiswa dalam Antrian");
19             System.out.println(x:"7. Jumlah Mahasiswa yang sudah melakukan KRS");
20             System.out.println(x:"8. Jumlah Mahasiswa yang belum melakukan KRS");
21             System.out.println(x:"9. Clear Antrian");
22             System.out.println(x:"0. Keluar");
23             System.out.print(s:"Pilih Menu: ");
24             pilihan = sc.nextInt();
25
26             switch (pilihan) {
27                 case 1:
28                     System.out.print(s:"NIM: ");
29                     String nim = sc.next();
30                     System.out.print(s:"Nama: ");
31                     String nama = sc.next();
32                     System.out.print(s:"Prodi: ");
33                     String prodi = sc.next();
34                     System.out.print(s:"Kelas: ");
35                     String kelas = sc.next();
36                     MahasiswaKRS mhs = new MahasiswaKRS(nim, nama, prodi, kelas);
37                     antrian.tambahAntrian(mhs);
38                     break;
39                 case 2:
40                     MahasiswaKRS[] dilayani = antrian.layaniMahasiswa();
41                     if (dilayani != null) {
42                         System.out.println(x:"Melayani Mahasiswa: ");
43                         System.out.println(dilayani[0].tampilkanData());
44                         System.out.println(dilayani[1].tampilkanData());
45                     }
46                     break;

```

```

47                     break;
48                 case 3:
49                     antrian.lihatTerdepan();
50                     break;
51                 case 4:
52                     antrian.tampilkanSemua();
53                     break;
54                 case 5:
55                     System.out.println("Jumlah dalam antrian: " + antrian.getJumlahAntrian());
56                     break;
57                 case 0:
58                     System.out.println(x:"Terima kasih.");
59                     break;
60                 default:
61                     System.out.println(x:"Pilihan tidak valid.");
62             }
63         } while (pilihan != 0);
64     }
65 }

```

AntrianKRS :

```
package jobsheet10;

public class AntrianKRS {
    MahasiswaKRS[] data;
    int front;
    int rear;
    int size;
    int max;
    int kuota = 30;

    public AntrianKRS() {
        max = 10;
        data = new MahasiswaKRS[max];
        front = size = 0;
        rear = -1;
    }

    public boolean isFull() {
        return size == max;
    }

    public boolean isEmpty() {
        return size == 0;
    }

    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(MahasiswaKRS mhs) {
        if (isFull()) {
            System.out.println("Antrian sudah penuh, tidak dapat menambahkan Mahasiswa.");
            return;
        }
        rear = (rear + 1) % max;
        data[rear] = mhs;
        size++;
        System.out.println(mhs.nama + " berhasil masuk ke antrian.");
    }

    public MahasiswaKRS[] layaniMahasiswa() {
        MahasiswaKRS[] dilayani = new MahasiswaKRS[2];
        if (isEmpty()) {
            System.out.println("Antrian kosong.");
            return null;
        }
        if (size < 2) {
            System.out.println("Masih kurang dari 2 Mahasiswa dalam Antrian.");
            return null;
        }
        dilayani[0] = data[front];
        dilayani[1] = data[(front + 1) % max];
        front = (front + 2) % max;
        size -= 2;
        kuota -= 2;
        return dilayani;
    }
}
```

```

public void tampilkanSemua() {
    if (isEmpty()) {
        System.out.println(x:"Antrian kosong.");    Replace this use of System.out by a logger.
        return;
    }
    System.out.println(x:"Daftar Mahasiswa dalam Antrian");    Replace this use of System.out by a logger.
    System.out.println(x:"NIM - NAMA - PRODI - KELAS");    Define a constant instead of duplicating this
    for (int i = 0; i < size; i++) {
        int index = (front + i) % max;
        System.out.print((i + 1) + ". ");    Replace this use of System.out by a logger.
        data[index].tampilkanData();
    }
}

public void lihatDuaTerdepan() {
    if (isEmpty()) {
        System.out.println(x:"Antrian kosong.");    Replace this use of System.out by a logger.
    } else {
        System.out.print(s:"Mahasiswa terdepan: ");    Replace this use of System.out by a logger.
        System.out.println(x:"NIM - NAMA - PRODI - KELAS");    Replace this use of System.out by a logger.
        data[front].tampilkanData();
        data[front + 1].tampilkanData();
    }
}
}

```

```

}

public void lihatTerakhir() {
    if (isEmpty()) {
        System.out.println(x:"Antrian kosong.");    Replace this use of System.out by a logger.
    } else {
        System.out.print(s:"Mahasiswa terakhir: ");    Replace this use of System.out by a logger.
        System.out.println(x:"NIM - NAMA - PRODI - KELAS");    Replace this use of System.out by a logger.
        data[rear].tampilkanData();
    }
}

public void getJumlahAntrian() {
    if (isEmpty()) {
        System.out.println(x:"Antrian kosong.");    Replace this use of System.out by a logger.
    } else {
        System.out.println("Jumlah Antrian = " + size);    Replace this use of System.out by a logger.
    }
}

public void getMhsBelumProses() {
    System.out.println("Mahasiswa yang belum melakukan proses KRS: " + (kuota - size));    Replace this
}

public void getMhsSudahProses() {
    System.out.println("Mahasiswa sudah melakukan proses KRS: " + (30 - kuota + size));    Replace this
}
}

```

MahasiswaKRS :

```

1 package jobsheet10;    Rename this package name to match the regular expression '^[a-z_]+(
2
3 public class MahasiswaKRS {
4     String nim;
5     String nama;
6     String prodi;
7     String kelas;
8
9     public MahasiswaKRS(String nim, String nama, String prodi, String kelas) {
10         this.nim = nim;
11         this.nama = nama;
12         this.prodi = prodi;
13         this.kelas = kelas;
14     }
15
16     public String tampilkanData() {
17         return nim + " - " + nama + " - " + prodi + " - " + kelas;
18     }
19 }

```

HASIL RUB :

```
=== Menu Antrian Layanan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Tampilkan Antrian
4. Tampilkan 2 Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Jumlah Mahasiswa dalam Antrian
7. Jumlah Mahasiswa yang sudah melakukan KRS
8. Jumlah Mahasiswa yang belum melakukan KRS
9. Clear Antrian
0. Keluar
Pilih Menu: 1
NIM: 123
Nama: Jeno
Prodi: TI
Kelas: 1B
Jeno berhasil masuk ke antrian.

=== Menu Antrian Layanan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Tampilkan Antrian
4. Tampilkan 2 Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Jumlah Mahasiswa dalam Antrian
7. Jumlah Mahasiswa yang sudah melakukan KRS
8. Jumlah Mahasiswa yang belum melakukan KRS
9. Clear Antrian
0. Keluar
Pilih Menu: 1
NIM: 124
Nama: Jaehyun
Prodi: TI
Kelas: 1C
Jaehyun berhasil masuk ke antrian.

=== Menu Antrian Layanan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Tampilkan Antrian
4. Tampilkan 2 Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Jumlah Mahasiswa dalam Antrian
7. Jumlah Mahasiswa yang sudah melakukan KRS
8. Jumlah Mahasiswa yang belum melakukan KRS
9. Clear Antrian
0. Keluar
Pilih Menu: 1
NIM: 125
Nama: Nanon
Prodi: TI
Kelas: 1B
Nanon berhasil masuk ke antrian.
```

```
=== Menu Antrian Layanan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Tampilkan Antrian
4. Tampilkan 2 Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Jumlah Mahasiswa dalam Antrian
7. Jumlah Mahasiswa yang sudah melakukan KRS
8. Jumlah Mahasiswa yang belum melakukan KRS
9. Clear Antrian
0. Keluar
Pilih Menu: 3
Daftar Mahasiswa dalam Antrian
NIM - NAMA - PRODI - KELAS
1. 123 - Jeno - TI - 1B
2. 124 - Jaehyun - TI - 1C
3. 125 - Nanon - TI - 1B
```

```
=== Menu Antrian Layanan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Tampilkan Antrian
4. Tampilkan 2 Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Jumlah Mahasiswa dalam Antrian
7. Jumlah Mahasiswa yang sudah melakukan KRS
8. Jumlah Mahasiswa yang belum melakukan KRS
9. Clear Antrian
0. Keluar
Pilih Menu: 4
Mahasiswa terdepan: NIM - NAMA - PRODI - KELAS
123 - Jeno - TI - 1B
124 - Jaehyun - TI - 1C
```

```
=== Menu Antrian Layanan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Tampilkan Antrian
4. Tampilkan 2 Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Jumlah Mahasiswa dalam Antrian
7. Jumlah Mahasiswa yang sudah melakukan KRS
8. Jumlah Mahasiswa yang belum melakukan KRS
9. Clear Antrian
0. Keluar
Pilih Menu: 2
Melayani Mahasiswa:
123 - Jeno - TI - 1B
124 - Jaehyun - TI - 1C
```

```
=== Menu Antrian Layanan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Tampilkan Antrian
4. Tampilkan 2 Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Jumlah Mahasiswa dalam Antrian
7. Jumlah Mahasiswa yang sudah melakukan KRS
8. Jumlah Mahasiswa yang belum melakukan KRS
9. Clear Antrian
0. Keluar
Pilih Menu: 5
Mahasiswa terakhir: NIM - NAMA - PRODI - KELAS
125 - Nanon - TI - 1B
```

```
=== Menu Antrian Layanan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Tampilkan Antrian
4. Tampilkan 2 Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Jumlah Mahasiswa dalam Antrian
7. Jumlah Mahasiswa yang sudah melakukan KRS
8. Jumlah Mahasiswa yang belum melakukan KRS
9. Clear Antrian
0. Keluar
Pilih Menu: 6
Jumlah Antrian = 1
```

```
=== Menu Antrian Layanan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Tampilkan Antrian
4. Tampilkan 2 Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Jumlah Mahasiswa dalam Antrian
7. Jumlah Mahasiswa yang sudah melakukan KRS
8. Jumlah Mahasiswa yang belum melakukan KRS
9. Clear Antrian
0. Keluar
Pilih Menu: 7
Mahasiswa sudah melakukan proses KRS: 3
```



```
=== Menu Antrian Layanan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Tampilkan Antrian
4. Tampilkan 2 Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Jumlah Mahasiswa dalam Antrian
7. Jumlah Mahasiswa yang sudah melakukan KRS
8. Jumlah Mahasiswa yang belum melakukan KRS
9. Clear Antrian
0. Keluar
Pilih Menu: 8
Mahasiswa yang belum melakukan proses KRS: 27
```

```
=== Menu Antrian Layanan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Tampilkan Antrian
4. Tampilkan 2 Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Jumlah Mahasiswa dalam Antrian
7. Jumlah Mahasiswa yang sudah melakukan KRS
8. Jumlah Mahasiswa yang belum melakukan KRS
9. Clear Antrian
0. Keluar
Pilih Menu: 9
Queue berhasil dikosongkan.
```

```
=== Menu Antrian Layanan KRS ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Tampilkan Antrian
4. Tampilkan 2 Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Jumlah Mahasiswa dalam Antrian
7. Jumlah Mahasiswa yang sudah melakukan KRS
8. Jumlah Mahasiswa yang belum melakukan KRS
9. Clear Antrian
0. Keluar
Pilih Menu: 0
Terimakasih.
PS D:\kuliah\smt 2\alsd\praktikum-alsd-1\jobsheet10> []
```