

LAPORAN PRAKTIKUM MINGGU KE-9

“[Queue]”



Disusun oleh:

[Daffa Aqila Rahmatullah]

[2041720098]

**D4 TEKNIK INFORMATIKA
TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG**

2021

LAPORAN

“PRAKTIKUM”

A. KODE PROGRAM

8.2.1 Langkah-langkah Percobaan

```
package Praktikum1;

/**
 *
 * @author ACHE
 */
public class Queue {
    int max,size,front,rear;
    int Q[];

    public Queue(int n){
        max = n;
        Create();
    }

    public void Create(){
        Q = 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 terdapat: " +
Q[front]);
        }
        else
        {
            System.out.println("Queue Masih Kosong");
        }
    }

    void print(){
        if(IsEmpty()){
            System.out.println("Queue masih kosong");
        }else{
            int i = front;
            while(i != rear){
                System.out.print(Q[i]+" ");
                i = (i+1) % max;
            }
        }
    }

```

```

        }
        System.out.println(Q[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 data){
    if(IsFull())
    {
        System.out.println("Queue sudah penuh");
    }
    else
    {
        if(IsEmpty())
        {
            front = rear = 0;
        }
        else
        {
            if(rear == max -1)

```

```

        {
            rear = 0;
        }
        else
        {
            rear++;
        }
    }
    Q[rear] = data;
    size++;
}
}

public int Dequeue(){
    int data = 0;
    if(IsEmpty())
    {
        System.out.println("Queue Masih Kosong");
    }
    else
    {
        data = Q[front];
        size--;
        if(IsEmpty())
        {
            front = rear = -1;
        }
        else
        {
            if(front == max -1)
            {
                front = 0;
            }
            else
            {

```

```

        front++;
    }
}

return data;
}

package Praktikum1;

import java.util.Scanner;

public class QueueMain {

    public static void menu(){
        System.out.println("Masukkan Operasi yang di
inginkan:");
        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("6. peek position");
//        System.out.println("7. peek at");
        System.out.println("-----");
    }

    public static void Main(){
        Scanner sc = new Scanner(System.in);
        System.out.println("Masukkan Kapasitas Queue: ");
        int n = sc.nextInt();

        Queue Q = new Queue(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;
    // case 6:
    //     System.out.println("Masukkan data :
");
    //     int data = sc.nextInt();
    //     Q.peekPosition(data);
    //     break;
    // case 7:
    //     System.out.println("cari posisi index
: ");

```

```

//                int cari = sc.nextInt();
//                Q.peekAt(cari);
//                break;
            }
        }while (pilih == 1 || pilih == 2 || pilih == 3 ||
pilih == 4 || pilih == 5);
    }
    public static void main(String[] args) {
        Main();
    }
}

```

8.3.1 Langkah-langkah Percobaan

```

package Praktikum2;

/**
 *
 * @author ACHE
 */
public class Penumpang {
    String nama, kotaAsal, kotaTujuan;
    int jumlahTiket, harga;

    Penumpang(String nama, String kotaAsal, String
kotaTujuan, int jml, int harga){
        this.nama = nama;
        this.kotaAsal = kotaAsal;
        this.kotaTujuan = kotaTujuan;
        this.jumlahTiket = jml;
        this.harga = harga;
    }
}
package Praktikum2;

/**

```



```

*
* @author ACHE
*/
public class Queue {
    int front, rear, max, size;
    Penumpang[] Q;

    public Queue(int n){
        max =n;
        Create();
    }

    public void Create(){
        Q= new Penumpang[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: "+
Q[front].nama+" "+Q[front].kotaAsal
            +" "+Q[front].kotaTujuan+"
"+Q[front].jumlahTiket+" "+Q[front].harga);
        }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.println(Q[i].nama+"
"+Q[i].kotaAsal+" "+Q[i].kotaTujuan
                +" "+Q[i].jumlahTiket+" "+Q[i].harga);
                i = (i+1)%max;
            }
            System.out.println(Q[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(Penumpang data){
    if(IsFull()){
        System.out.println("Queue sudah penuh");
    }else{
        if(IsEmpty()){
            front=rear=0;
        }else{
            if(rear==max-1){
                rear=0;
            }else{
                rear++;
            }
        }
        Q[rear]=data;
        size++;
    }
}

```

```

public Penumpang Dequeue(){
    Penumpang data = new Penumpang(" ", " ", " ", 0,0);
    if(IsEmpty()){
        System.out.println("Queue masih kosong");
    }else{
        data=Q[front];
        size--;
        if(IsEmpty()){
            front=rear=-1;
        }else{
            if(front==max-1){

```

```

        front=0;
    }else{
        front++;
    }
}
}
return data;
}
}
package Praktikum2;

import java.util.*;

public class QueueMain {

    public static void menu() {
        System.out.println("pilih menu : ");
        System.out.println("1. Antrian baru");
        System.out.println("2. Antrian keluar");
        System.out.println("3. Cek Antrian terdepan");
        System.out.println("4. Cek Semua Antrian");
        System.out.println("-----");
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Masukan kapasitas queue :");
        int jumlah = sc.nextInt();
        Queue antri = new Queue(jumlah);
        int pilih;
        do{
            menu();
            pilih = sc.nextInt();
            sc.nextLine();

```

```

switch(pilih) {
    case 1:
        System.out.println("Nama: ");
        String nama = sc.nextLine();
        System.out.println("Kota Asal: ");
        String asal = sc.nextLine();
        System.out.println("Kota Tujuan: ");
        String tujuan = sc.nextLine();
        System.out.println("Jumlah Tiket: ");
        int jml = sc.nextInt();
        System.out.println("Harga: ");
        int hrg = sc.nextInt();
        Penumpang p = new Penumpang(nama, asal,
tujuan, jumlah, hrg);
        sc.nextLine();
        antri.Enqueue(p);
        break;
    case 2:
        Penumpang data = antri.Dequeue();

        if(!"".equals(data.nama)&&!"".equals(data.kotaAsal)&&!"".equals
        (data.kotaTujuan)
                &&data.jumlahTiket
        !=0&&data.harga !=0) {
            System.out.println("Antrian yang
        keluar: "+data.nama+" "+data.kotaAsal+" "+
                data.kotaTujuan+"
        "+data.jumlahTiket+" "+data.harga);
            break;
        }
    case 3:
        antri.peek();
        break;
    case 4:

```

```

        antri.print();
        break;
    }
}while(pilih == 1||pilih == 2||pilih == 3||pilih ==
4);

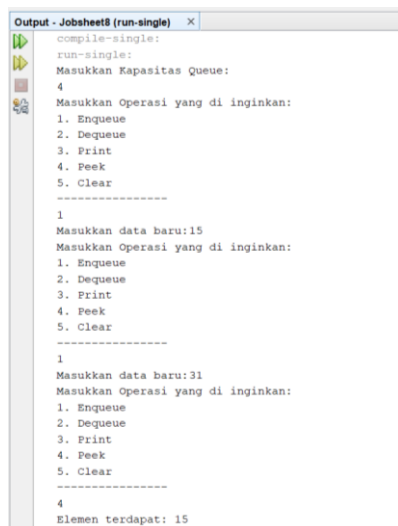
}

}

```

B. OUTPUT PROGRAM

Percobaan 8.2.1

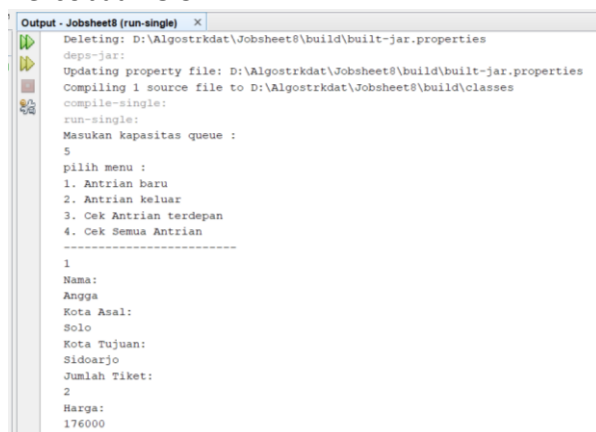


```

Output - Jobsheet8 (run-single) x
compile-single:
run-single:
Masukkan Kapasitas Queue:
4
Masukkan Operasi yang di inginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----
1
Masukkan data baru:15
Masukkan Operasi yang di inginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----
1
Masukkan data baru:31
Masukkan Operasi yang di inginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----
4
Elemen terdapat: 15

```

Percobaan 8.3.1



```

Output - Jobsheet8 (run-single) x
Deleting: D:\Algostrkdat\Jobsheet8\build\build-jar.properties
deps-jar:
Updating property file: D:\Algostrkdat\Jobsheet8\build\build-jar.properties
Compiling 1 source file to D:\Algostrkdat\Jobsheet8\build\classes
compile-single:
run-single:
Masukan kapasitas queue :
5
pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
-----
1
Nama:
Angga
Kota Asal:
Solo
Kota Tujuan:
Sidoarjo
Jumlah Tiket:
2
Harga:
176000
....

```

```
Output - Jobsheets (run-single) x
176000
pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
-----
1
Nama:
Fadin
Kota Asal:
Banyuwangi
Kota Tujuan:
Bandung
Jumlah Tiket:
1
Marga:
65000
pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
-----
3
Elemen terdepan: Angga Solo Sidoarjo 5 176000
....
```

C. PENJELASAN

7.2.3 Pertanyaan

1. pada method Create, mengapa atribut front dan rear di inisialisasi dengan nilai -1, tidak 0?

Jawab :

Karena tidak menunjuk ke data manapun kalau 0 akan menunjuk ke salah satu data

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

```
if(rear == max -1)
{
    rear = 0;
```

Jawab :

Jika rear di cek(nilai akhir hasil inputan) nilai yang telah di isikan tidak melebihi dari nilai max dikurang 1 karena array dimulai dari angka 0, maka nilai pertama dari indeks akan diisi dengan indek 0.

3. Perhatikan Kembali method Enqueue, baris kode program manakah yang menunjukkan bahwa data baru disimpan pada posisi terakhir dalam queue?

Jawab:

`Q[rear] = data;` karena sesuai prinsip yaitu (FIFO) First In First Out dimana nilai array awal yang dimasukkan akan diletakkan di bagian akhir array.

4. Perhatikan Kembali method Dequeue baris kode program manakah yang menunjukkan bahwa data yang dikeluarkan adalah data pada posisi paling depan di dalam queue?

Jawab:

`data = Q[front];` karena sesuai prinsip yaitu (FIFO) First In First Out jadi dimana nilai array akhir yang dimasukkan akan diletakkan di bagian awal array.

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

```
if(front == max -1)
{
    front = 0;
}
```

Jawab:

Jika front di cek(nilai akhir hasil inputan) nilai yang telah di isikan tidak melebihi dari nilai max dikurang 1 karena array dimulai dari angka 0, maka nilai pertama dari indeks akan diisi dengan indeks 0.

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

Jawab:

Karena posisi yang menjadi index pada front tersebut bisa berubah-ubah sesuai dengan index berupa yang keluar. Jadi tidak selamanya index 0 merupakan front.

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

```
i = (i+1) % max;
```

jawab:

ini digunakan untuk mencetak i selanjutnya. Jika max nya dalah 5 maka data yang akan di cetak pertama adalah data pada index ke 0, kemudian untuk selanjutnya data yang di cetak adalah $(0+1)\%10$ yaitu index ke 2, kemudian dengan menggunakan cara yang sama sampai pada index ke 4. Perulangan ini akan terus berlangsung sampai pada $i=\text{rear}-1$, jika i tersebut sama dengan rear maka pencarian akan berhenti.

7.3.3 Pertanyaan

1. Perhatikan class Queue, apa fungsi kode program berikut pada method Dequeue?

```
Penumpang data = new Penumpang(" ", " ", " ", 0, 0);
```

Jawab :

Karena di dalam class Penumpang dan di kontrukstor 5 data dan data tersebut untuk menyimpan atribut yang ada di class queue.

```
Q[front];
```

2. pada soal nomer 1, apabila kode program tersebut diganti dengan kode berikut

```
Penumpang data = new Penumpang();
```

Jawab:

Akan error karena tidak ada di dalam instansiasi Mahasiswa harus ada nilainya walaupun kosong.

3. Tunjukkan Kode program yang digunakan untuk menampilkan data yang dikeluarkan dari queue!

Jawab :

Script:

```
public void print() {
    if(IsEmpty()) {
        System.out.println("Queue masih kosong");
    }else{
        int i=front;
        while(i != rear){
            System.out.println(Q[i].nama+"
"+Q[i].kotaAsal+" "+Q[i].kotaTujuan
            +" "+Q[i].jumlahTiket+" "+Q[i].harga);
            i = (i+1)%max;
        }
        System.out.println(Q[i]+" ");
        System.out.println("Jumlah elemen = "+ size);
    }
}
```

4. Lakukan modifikasi program dengan menambahkan method baru bernama peekRear pada class Queue yang digunakan untuk mengecek antrian yang berada di posisi belakang! Tambahkan pula daftar menu

Jawab :

Script :

```
package Praktikum2;
```

```
/**
```

```
*
```

```
* @author ACHE
```

```
*/
```

```
public class Penumpang {
```

```
String nama, kotaAsal, kotaTujuan;
```

```
int jumlahTiket, harga;
```

```
Penumpang(String nama, String kotaAsal, String kotaTujuan, int jml, int harga){
```

```
    this.nama = nama;
```

```
    this.kotaAsal = kotaAsal;
```

```
    this.kotaTujuan = kotaTujuan;
```

```
    this.jumlahTiket = jml;
```

```
    this.harga = harga;
```

```
}
```

```
}
```

```
package Praktikum2;
```

```
/**
```

```
 *
```

```
 * @author ACHE
```

```
 */
```

```
public class Queue {
```

```
    int front, rear, max, size;
```

```
    Penumpang[] Q;
```

```
    public Queue(int n){
```

```
        max =n;
```

```
        Create();
```

```
    }
```

```
    public void Create(){
```

```
        Q= new Penumpang[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: "+ Q[front].nama+" "+Q[front].kotaAsal
        +" "+Q[front].kotaTujuan+" "+Q[front].jumlahTiket+" "+Q[front].harga);
    }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.println(Q[i].nama+" "+Q[i].kotaAsal+" "+Q[i].kotaTujuan
        +" "+Q[i].jumlahTiket+" "+Q[i].harga);
        i = (i+1)%max;
    }
    System.out.println(Q[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(Penumpang data){
    if(IsFull()){
        System.out.println("Queue sudah penuh");
    }else{
        if(IsEmpty()){
            front=rear=0;
        }else{
            if(rear==max-1){
                rear=0;
            }else{
                rear++;
            }
        }
    }
}

```

```

        Q[rear]=data;
        size++;
    }
}

```

```

public Penumpang Dequeue(){
    Penumpang data = new Penumpang(" ", " ", " ", 0,0);
    if(IsEmpty()){
        System.out.println("Queue masih kosong");
    }else{
        data=Q[front];
        size--;
        if(IsEmpty()){
            front=rear=-1;
        }else{
            if(front==max-1){
                front=0;
            }else{
                front++;
            }
        }
    }
    return data;
}

```

```

public void peekRear(){
    if(!IsEmpty()){
        System.out.println("Elemen terdepan: "+ Q[rear].nama+" "+Q[rear].kotaAsal
        +" "+Q[rear].kotaTujuan+" "+Q[rear].jumlahTiket+" "+Q[rear].harga);
    }else{
        System.out.println("Queue masih kosong ");
    }
}

```

```

    }
}
package Praktikum2;

import java.util.*;

public class QueueMain {

    public static void menu() {
        System.out.println("pilih menu : ");
        System.out.println("1. Antrian baru");
        System.out.println("2. Antrian keluar");
        System.out.println("3. Cek Antrian terdepan");
        System.out.println("4. Cek Semua Antrian");
        System.out.println("5. cek Antrian Belakang");
        System.out.println("-----");
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Masukan kapasitas queue :");
        int jumlah = sc.nextInt();
        Queue antri = new Queue(jumlah);
        int pilih;
        do{
            menu();
            pilih = sc.nextInt();
            sc.nextLine();
            switch(pilih){
                case 1:
                    System.out.println("Nama: ");
                    String nama = sc.nextLine();

```

```

        System.out.println("Kota Asal: ");
        String asal = sc.nextLine();
        System.out.println("Kota Tujuan: ");
        String tujuan = sc.nextLine();
        System.out.println("Jumlah Tiket: ");
        int jml = sc.nextInt();
        System.out.println("Harga: ");
        int hrg = sc.nextInt();
        Penumpang p = new Penumpang(nama, asal, tujuan, jumlah, hrg);
        sc.nextLine();
        antri.Enqueue(p);
        break;
    case 2:
        Penumpang data = antri.Dequeue();

        if(!"".equals(data.nama)&&!"".equals(data.kotaAsal)&&!"".equals(data.kotaTujuan)
            &&data.jumlahTiket !=0&&data.harga !=0){
            System.out.println("Antrian yang keluar: "+data.nama+" "+data.kotaAsal+"
"+
                data.kotaTujuan+" "+data.jumlahTiket+" "+data.harga);
            break;
        }
    case 3:
        antri.peek();
        break;
    case 4:
        antri.print();
        break;
    case 5:
        antri.peekRear();
        break;
}

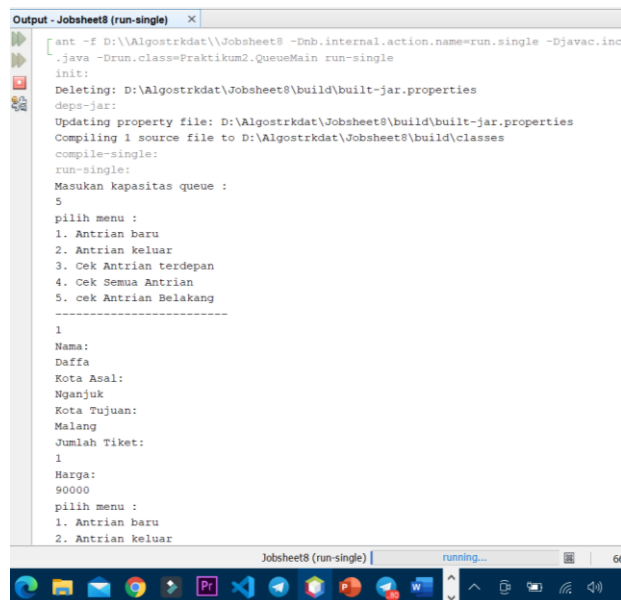
```

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

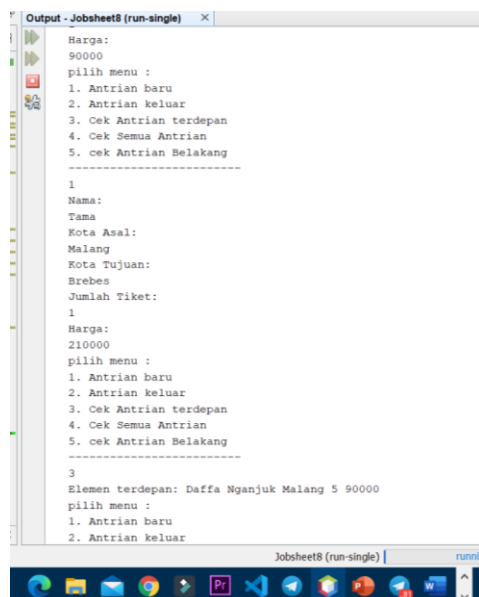
```
}
```

```
}
```

Running :



```
Output - Jobsheet8 (run-single) x
[ ant -f D:\Algostrkdat\Jobsheet8 -Dmb.internal.action.name=run.single -Djavac.inc
  .java -Drun.class=Praktikum2.QueueMain run-single
init:
Deleting: D:\Algostrkdat\Jobsheet8\build\build-jar.properties
deps-jar:
Updating property file: D:\Algostrkdat\Jobsheet8\build\build-jar.properties
Compiling 1 source file to D:\Algostrkdat\Jobsheet8\build\classes
compile-single:
run-single:
Masukan Kapasitas queue :
5
pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. cek Antrian Belakang
-----
1
Nama:
Daffa
Kota Asal:
Nganjuk
Kota Tujuan:
Malang
Jumlah Tiket:
1
Harga:
90000
pilih menu :
1. Antrian baru
2. Antrian keluar
```



```
Output - Jobsheet8 (run-single) x
Harga:
90000
pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. cek Antrian Belakang
-----
3
Elemen terdepan: Daffa Nganjuk Malang 5 90000
pilih menu :
1. Antrian baru
2. Antrian keluar
```



```
Output - Jobsheet8 (run-single) X
Kota Tujuan:
Brebes
Jumlah Tiket:
1
Harga:
210000
pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. cek Antrian Belakang
-----
3
Elemen terdepan: Daffa Nganjuk Malang 5 90000
pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. cek Antrian Belakang
-----
5
Elemen terdepan: Tama Malang Brebes 5 210000
pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. cek Antrian Belakang
-----
```

D. KESIMPULAN

Queue memiliki prinsip yang sama dengan stack. Perbedaan diantara keduanya hanya pada pengambilan datanya. Jika di stack mengambil data dari urutan paling akhir, tetapi jika pada queue mengambil data pada urutan paling depan. Pada praktikum terdapat method enqueue dan method dequeue. Method enqueue memiliki fungsi untuk menambahkan data pada queue sedangkan method dequeue memiliki fungsi untuk mengambil data pada queue. Queue juga tidak hanya dapat menyimpan data primitive saja tetapi juga dapat menyimpan data bertipe String dan objek.

“TUGAS”

1. Tambahkan dua method berikut ke dalam class Queue pada Praktikum1: a. Method `peekPosition(data: int) : void` Untuk menampilkan posisi dari sebuah data di dalam queue, misalnya dengan mengirimkan data tertentu, akan diketahui posisi (indeks) data tersebut berada di urutan keberapa b. Method `peekAt(position: int) : void` Untuk menampilkan data yang berada pada posisi (indeks) tertentu Sesuaikan daftar menu yang terdapat pada class QueueMain sehingga kedua method tersebut dapat dipanggil!
2. Buatlah program antrian untuk mengilustasikan mahasiswa yang sedang meminta tanda tangan KRS pada dosen DPA di kampus. Ketika seorang mahasiswa akan mengantri, maka dia harus menuliskan terlebih dulu NIM, nama, absen, dan IPK seperti yang digambarkan pada Class diagram berikut:

Mahasiswa
Nim:String Nama:String Absen: int Ipk:double
Mahasiswa(nim: String, nama: String, absen: int, ipk: double)

Class diagram Queue digambarkan sebagai berikut:

Queue
Max:int front:int rear:int size: int antrian: Mahasiswa[]
Queue(max:int) create(): void isEmpty(): boolean isFull(): boolean enqueue(antrian: Mahasiswa): void dequeue(): int print(): void peek(): void peekRear(): void peekPosition(): void peekPosistion(nim: String): void printMahasiswa(posisi: int): void

~~JAWAB~~

1. Tugas1

A. Kode Program

```
package Praktikum1;
```

```

/**
 *
 * @author ACHE
 */
public class Queue {
    int max,size,front,rear;
    int Q[];

    public Queue(int n){
        max = n;
        Create();
    }

    public void Create(){
        Q = 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 terdapat: " +
Q[front]);
        }
        else
        {
            System.out.println("Queue Masih Kosong");
        }
    }
}

```

```

    }

    void print(){
        if(IsEmpty()){
            System.out.println("Queue masih kosong");
        }else{
            int i = front;
            while(i != rear){
                System.out.print(Q[i]+" ");
                i = (i+1) % max;
            }
            System.out.println(Q[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 data){
        if(IsFull())
        {
            System.out.println("Queue sudah penuh");
        }
        else
        {
            if(IsEmpty())
            {
                front = rear = 0;
            }
            else
            {
                if(rear == max -1)
                {
                    rear = 0;
                }
                else
                {
                    rear++;
                }
            }
        }
    }

```

```

        }
    }
    Q[rear] = data;
    size++;
}
}
public int Dequeue(){
    int data = 0;
    if(IsEmpty())
    {
        System.out.println("Queue Masih Kosong");
    }
    else
    {
        data = Q[front];
        size--;
        if(IsEmpty())
        {
            front = rear = -1;
        }
        else
        {
            if(front == max -1)
            {
                front = 0;
            }
            else
            {
                front++;
            }
        }
    }
    return data;
}

void peekPosition(int data){
    if (!IsEmpty()) {
        for (int i = front; i < max; i++) {
            if (Q[i]==data) {
                System.out.println("Data "+data+"
berada pada index ke-"+i);
            }
        }
    }
    else{
        System.out.println("Queue masih kosong");
    }
}

void peekAt(int position){
    if(!IsEmpty()){
        for (int i = front; i < max; i++) {

```

```

        if(i == position){
            System.out.println("Data pada index
"+i+" yaitu : "+Q[i]);
        }
    }
    }else{
        System.out.println("Queue masih kosong");
    }
}
}
package Praktikum1;

import java.util.Scanner;

public class QueueMain {

    public static void menu(){
        System.out.println("Masukkan Operasi yang di
inginkan:");
        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("6. peek position");
        System.out.println("7. peek at");
        System.out.println("-----");
    }
    public static void Main(){
        Scanner sc = new Scanner(System.in);
        System.out.println("Masukkan Kapasitas Queue:
");
        int n = sc.nextInt();

        Queue Q = new Queue(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)
                    {

```

```

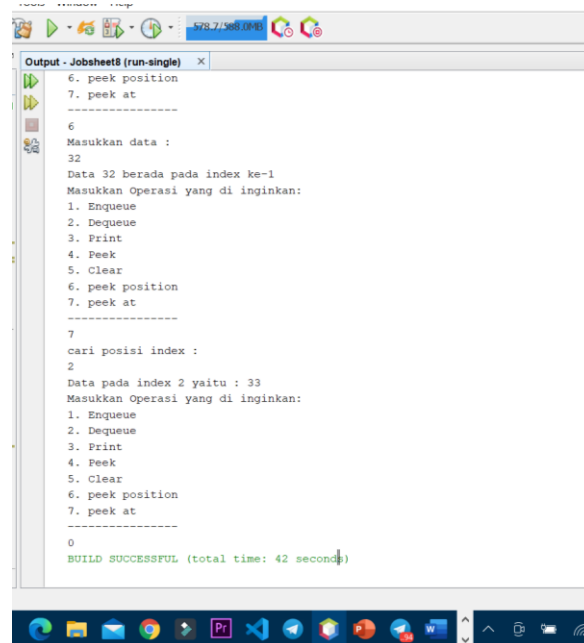
        System.out.println("Data Yang
dikeluarkan: " + dataKeluar);
        break;
    }
    case 3:
        Q.print();
        break;
    case 4:
        Q.peek();
        break;
    case 5:
        Q.clear();
        break;
    case 6:
        System.out.println("Masukkan data :
");
        int data = sc.nextInt();
        Q.peekPosition(data);
        break;
    case 7:
        System.out.println("cari posisi
index : ");
        int cari = sc.nextInt();
        Q.peekAt(cari);
        break;
    }
    }while (pilih == 1 || pilih == 2 || pilih == 3
|| pilih == 4 || pilih == 5 || pilih == 6 || pilih
==7);
    }
    public static void main(String[] args) {
        Main();
    }
}

```

**B. Output Program
Running :**

```
Output - Jobsheet8 (run-single) X
[ant -f D:\Algostrkdat\Jobsheet8 -Dmb.internal.action.name=run.single -Djava
.java -Drun.class=Praktikum1.QueueMain run-single
init:
Deleting: D:\Algostrkdat\Jobsheet8\build\build-jar.properties
deps-jar:
Updating property file: D:\Algostrkdat\Jobsheet8\build\build-jar.properties
Compiling 1 source file to D:\Algostrkdat\Jobsheet8\build\classes
compile-single:
run-single:
Masukkan Kapasitas Queue:
5
Masukkan Operasi yang di inginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. peek position
7. peek at
-----
1
Masukkan data baru:31
Masukkan Operasi yang di inginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. peek position
7. peek at
-----
1
```

```
Output - Jobsheet8 (run-single) X
1
Masukkan data baru:32
Masukkan Operasi yang di inginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. peek position
7. peek at
-----
1
Masukkan data baru:33
Masukkan Operasi yang di inginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. peek position
7. peek at
-----
6
Masukkan data :
32
Data 32 berada pada index ke-1
Masukkan Operasi yang di inginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
```

```
Output - Jobsheet8 (run-single) X
6. peek position
7. peek at
-----
6
Masukkan data :
32
Data 32 berada pada index ke-1
Masukkan Operasi yang di inginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. peek position
7. peek at
-----
7
cari posisi index :
2
Data pada index 2 yaitu : 33
Masukkan Operasi yang di inginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. peek position
7. peek at
-----
0
BUILD SUCCESSFUL (total time: 42 seconds)
```

C. Penjelasan

Pada script diatas terdapat 2 class yaitu class Queue dan QueueMain. Didalam Queue terdapat sebuah atribut (int max, size, front, rear;) dan atribut dengan array (int[] Q;) dan sebuah konstruktor seperti public Queue(int n).

Didalam lalu ada beberapa method juga seperti :

- Create(); yang digunakan untuk membuat queue yang dilakukan dalam bentuk array yang akan di instansiasi.
- IsEmpty() dengan Boolean. Yang digunakan untuk mengecek apakah ada data yang kosong apa tidak. Jika if(size == 0) maka akan di return true dan jika else maka akan direturn false.
- IsFull() dengan Boolean. Yang berfungsi untuk mengecek apakah data penuh atau tidak. Jika if(size == max) maka akan direturn true dan jika else maka akan di return false.
- peek(). Digunakan untuk menampilkan data queue paling depan.
- Print(). Digunakan untuk menampilkan seluruh data queue.
- Clear(). Digunakan untuk menghapus seluruh data queue.
- Enqueue() dengan parameter int data.

Digunakan untuk menginput atau menambahkan data pada array queue.

- Dequeue(). Digunakan untuk mengeluarkan data dari array queue.

- peekPosition() dengan parameter int position

if (Q[i]==data) ini berguna untuk menyeleksi apakah Q[i] sama dengan data, data ini merupakan inputan di main

Digunakan untuk mencari posisi pada data index array queue yang telah tersimpan sebelumnya dan di dalam main terdapat inputan.

- peekAt() dengan parameter int index

if(i == position) ini berguna untuk menyeleksi apakah i sama dengan position

Digunakan untuk mencari index pada data posisi array queue yang telah tersimpan sebelumnya dan didalam main terdapat inputan.

D. Kesimpulan

Dari program di atas saya lebih tau cara penggunaan peekPosition dan peekAt

2. Tugas no 2

A. Kode Program

```
package Tugas;

/**
 *
 * @author ACHE
 */
public class Mahasiswa {
    String nim, nama;
    int absen;
    double ipk;
    Mahasiswa(String nim,String nama, int absen, double
ipk){
        this.nim = nim;
        this.nama = nama;
        this.absen = absen;
        this.ipk = ipk;
    }
}
package Tugas;

/**
 *
 * @author ACHE
 */
public class Queue {
    int max, front, rear, size;
    Mahasiswa []antrian;
    public Queue(int n){
        max = n;
        Create();
    }
    public void Create(){
        antrian = new Mahasiswa[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 Enqueue(Mahasiswa data){
    if(IsFull()){
        System.out.println("Queue sudah penuh");
    }else {
        if(IsEmpty()){
            front = rear= 0;
        }else {
            if( rear == max -1){
                rear =0;
            }else {
                rear++;
            }
        }
    }
    antrian[rear] = data;
    size++;
}
public Mahasiswa Dequeue(){
    Mahasiswa data = new Mahasiswa("", "", 0, 0);
    if(IsEmpty()){
        System.out.println("Queue masih kosong");
    }else {
        data = antrian[front];
        size--;
        if(IsEmpty()){
            front = rear = -1;
        }else {
            if(front == max -1){
                front = 0;
            }else {
                front++;
            }
        }
    }
    return data;
}
public void peek(){
    if(!IsEmpty()){
        System.out.println("Mahasiswa terdepan: " +
antrian[front].nim + " " + antrian[front].nama + " " +
antrian[front].absen + " " + antrian[front].ipk);
    }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("Mahasiswa terdepan: "
+ antrian[front].nim + " " + antrian[front].nama + " "
+ antrian[front].absen + " " + antrian[front].ipk);
                i = (i + 1) % max;
                System.out.println("");
            }
            System.out.println(antrian[i] + " ");
            System.out.println("Jumlah elemen = " +
size);
        }
    }
    public void peekRear(){
        if(!IsEmpty()){
            System.out.println("Elemen terakhir: " +
antrian[rear].nim + " " + antrian[rear].nama + " " +
antrian[rear].absen + " " + antrian[rear].ipk);
        }else {
            System.out.println("Queue masih kosong");
        }
    }
    public void peekPosition(int nim){
        if (!IsEmpty()) {
            for (int i = front; i < max; i++) {
                int nim1 =
Integer.parseInt(antrian[i].nim);
                if (nim == nim1) {
                    System.out.println("Mahasiswa
dengan nim "+ nim + " berada pada antrian ke-" + i);
                }
            }
        } else {
            System.out.println("Queue masih kosong");
        }
    }
    public void printMahasiswa(int position1){
        if (!IsEmpty()) {
            for (int i = front; i < max; i++) {
                if (i==position1) {
                    System.out.println("Mahasiswa yang
berada pada antrian ke-"+ i +" adalah :");
                    System.out.println(antrian[i].nim +
" " + antrian[i].nama + " " + antrian[i].absen + " " +
antrian[i].ipk);
                }
            }
        } else {
            System.out.println("Queue masih kosong");
        }
    }
}

```

```

    }
}
package Tugas;

import java.util.Scanner;

public class QueueMain {
    public static void menu() {
        System.out.println("pilih menu : ");
        System.out.println("1. Masukkan Mahasiswa");
        System.out.println("2. Data Mahasiswa keluar");
        System.out.println("3. Cek Antrian terdepan");
        System.out.println("4. Cek Semua Antrian");
        System.out.println("5. Cek Antrian belakang");
        System.out.println("6. cari indeks berdasarkan
NIM");
        System.out.println("7. cari data Mahasiswa
berdasarkan Indeks");
        System.out.println("-----
");
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Masukan kapasitas queue
:");

        int jumlah = sc.nextInt();
        Queue ttd = new Queue(jumlah);
        int pilih;

        do{
            menu();
            pilih = sc.nextInt();
            sc.nextLine();
            switch(pilih){
                case 1:
                    System.out.print("Nim: ");
                    String nim = sc.nextLine();
                    System.out.print("Nama: ");
                    String nama = sc.nextLine();
                    System.out.print("Absen: ");
                    int absen = sc.nextInt();
                    System.out.print("IPK: ");
                    double ipk = sc.nextDouble();

                    Mahasiswa m = new
Mahasiswa(nim,nama,absen,ipk);
                    sc.nextLine();
                    ttd.Enqueue(m);
                    break;
                case 2:
                    Mahasiswa data = ttd.Dequeue();

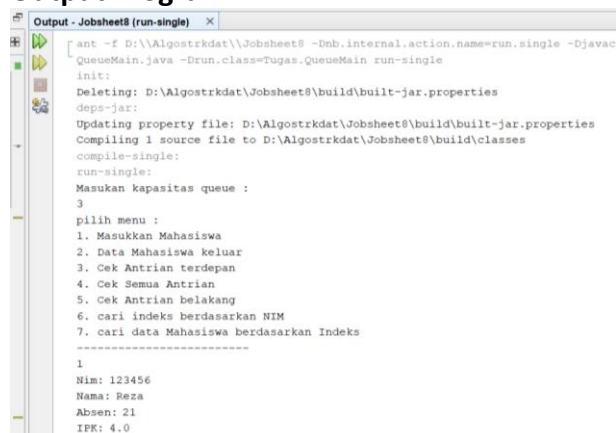
```

```

        if(!"".equals(data.nim) &&
!"".equals(data.nama) && data.absen != 0 && data.ipk !=
0){
            System.out.println("Antrian
yang keluar: " + data.nim + " " + data.nama + " " +
data.absen + " " + data.ipk);
            break;
        }
        case 3:
            ttd.peek();
            break;
        case 4:
            ttd.print();
            break;
        case 5:
            ttd.peekRear();
            break;
        case 6:
            System.out.print("Masukkan nim yang
ingin dicari: ");
            int posisi = sc.nextInt();
            ttd.peekPosition(posisi);
            break;
        case 7:
            System.out.print("Masukkan posisi
yang ingin dicari: ");
            int posisi1 = sc.nextInt();
            ttd.printMahasiswa(posisi1);
            break;
    }
    } while(pilih == 1||pilih == 2||pilih ==
3||pilih == 4||pilih == 5||pilih == 6 || pilih == 7);
}
}

```

B. Output Program



```

Output - Jobsheet8 (run-single) x
[ ant -f D:\Algostrkdat\Jobsheet8 -Dnb.internal.action.name=run.single -Djavac
QueueMain.java -Drun.class=Tugas.QueueMain run-single
init:
Deleting: D:\Algostrkdat\Jobsheet8\build\build-jar.properties
deps-jar:
Updating property file: D:\Algostrkdat\Jobsheet8\build\build-jar.properties
Compiling 1 source file to D:\Algostrkdat\Jobsheet8\build\classes
compile-single:
run-single:
Masukan kapasitas queue :
3
pilih menu :
1. Masukkan Mahasiswa
2. Data Mahasiswa keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. Cek Antrian belakang
6. cari indeks berdasarkan NIM
7. cari data Mahasiswa berdasarkan Indeks
-----
1
Nim: 123456
Nama: Reza
Absen: 21
IPK: 4.0

```

```
Output - Jobsheet8 (run-single) x
IPK: 4.0
pilih menu :
1. Masukkan Mahasiswa
2. Data Mahasiswa keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. Cek Antrian belakang
6. cari indeks berdasarkan NIM
7. cari data Mahasiswa berdasarkan Indeks
-----
1
Nim: 234567
Nama: Abdullah
Absen: 01
IPK: 3.5
pilih menu :
1. Masukkan Mahasiswa
2. Data Mahasiswa keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. Cek Antrian belakang
6. cari indeks berdasarkan NIM
7. cari data Mahasiswa berdasarkan Indeks
-----
1
Nim: 345678
Nama: Nadya
Absen: 17
IPK: 3.8
```

```
-----
pilih menu :
1. Masukkan Mahasiswa
2. Data Mahasiswa keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. Cek Antrian belakang
6. cari indeks berdasarkan NIM
7. cari data Mahasiswa berdasarkan Indeks
-----
2
Antrian yang keluar: 123456 Reza 21 4.0
pilih menu :
1. Masukkan Mahasiswa
2. Data Mahasiswa keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. Cek Antrian belakang
6. cari indeks berdasarkan NIM
7. cari data Mahasiswa berdasarkan Indeks
-----
3
Mahasiswa terdepan: 234567 Abdullah 1 3.5
pilih menu :
1. Masukkan Mahasiswa
2. Data Mahasiswa keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. Cek Antrian belakang
6. cari indeks berdasarkan NIM
7. cari data Mahasiswa berdasarkan Indeks
-----
4
Mahasiswa terdepan: 234567 Abdullah 1 3.5
```



```
pilih menu :
1. Masukkan Mahasiswa
2. Data Mahasiswa keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. Cek Antrian belakang
6. cari indeks berdasarkan NIM
7. cari data Mahasiswa berdasarkan Indeks
-----
4
Mahasiswa terdepan: 234567 Abdullah 1 3.5
-----
Output - Jobsheet8 (run-single) x
Jumlah elemen = 2
pilih menu :
1. Masukkan Mahasiswa
2. Data Mahasiswa keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. Cek Antrian belakang
6. cari indeks berdasarkan NIM
7. cari data Mahasiswa berdasarkan Indeks
-----
5
Elemen terakhir: 345678 Nadya 17 3.8
pilih menu :
1. Masukkan Mahasiswa
2. Data Mahasiswa keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. Cek Antrian belakang
6. cari indeks berdasarkan NIM
7. cari data Mahasiswa berdasarkan Indeks
-----
6
Masukkan nim yang ingin dicari: 234567
Mahasiswa dengan nim 234567 berada pada antrian ke-1
```

```

pilih menu :
1. Masukkan Mahasiswa
2. Data Mahasiswa keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. Cek Antrian belakang
6. cari indeks berdasarkan NIM
7. cari data Mahasiswa berdasarkan Indeks
-----
7
Masukkan posisi yang ingin dicariid
Mahasiswa yang berada pada antrian ke-1 adalah :
234567 Abdullah 1 3.5
pilih menu :
1. Masukkan Mahasiswa
2. Data Mahasiswa keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. Cek Antrian belakang
6. cari indeks berdasarkan NIM
7. cari data Mahasiswa berdasarkan Indeks
-----
0
BUILD SUCCESSFUL (total time: 3 minutes 57 seconds)
|

```

C. Penjelasan

Pada script diatas terdapat 3 class yaitu class Mahasiswa, Queue dan QueueMain. Didalam Mahasiswa terdapat atribut(String nim, String nama ,int absen, double ipk) Didalam Queue terdapat sebuah atribut (int max, size, front, rear;) dan atribut dengan array (Mahasiswa[] antrian;) dan sebuah konstruktor seperti public Queue(int n).

Didalam lalu ada beberapa method juga seperti :

- Create(); yang digunakan untuk membuat queue yang dilakukan dalam bentuk array yang akan di instansiasi.
- IsEmpty() dengan Boolean. Yang digunakan untuk mengecek apakah ada data yang kosong apa tidak. Jika if(size == 0) maka akan di return true dan jika else maka akan direturn false.
- IsFull() dengan Boolean. Yang berfungsi untuk mengecek apakah data penuh atau tidak. Jika if(size == max) maka akan direturn true dan jika else maka akan di return false.
- peek(). Digunakan untuk menampilkan data Mahasiswa paling depan.
- Print(). Digunakan untuk menampilkan seluruh data Mahasiswa.
- Enqueue() dengan parameter int data.

Digunakan untuk menginput atau menambahkan data pada array Mahasiswa.

- Dequeue(). Digunakan untuk mengeluarkan data dari array Mahasiswa.

- peekPosition() dengan parameter int position

int nim1 = Integer.parseInt(antrian[i].nim); untuk mengubah String menjadi integer dan if (nim == nim1) ini berguna untuk menyeleksi apakah nim yang kita nanti inputkan sama dengan nim1 yaitu perubahan String menjadi integer.

Digunakan untuk mencari posisi pada data index array Mahasiswa yang telah tersimpan sebelumnya dan di dalam main terdapat inputan.

- peekAt() dengan parameter int index

if(i == position) ini berguna untuk menyeleksi apakah i sama dengan position

Digunakan untuk mencari index pada data posisi array Mahasiswa yang telah tersimpan sebelumnya dan didalam main terdapat inputan.

- peekRear(). Digunakan untuk manampilkan data Mahasiswa paling belakang

D. Kesimpulan

Di program ini saya dapat menyimpulkan bahwa dapat mengambil posisi antrian paling belakang, dapat menampilkan posisi berapa mahasiswa tersebut, dan dapat menampilkan data posisi dengan memanggil indeksinya saja