

# **Laporan Praktikum Algoritma dan Struktur Data**

## **Jobsheet 10 - Queue**

**Dosen Pengampu : Triana Fatmawati, S.T., M.T.**



**Nama : Annisa**  
**Nim : 2341760032**  
**Kelas : SIB 1E**  
**Prodi : D-IV Sistem Informasi Bisnis**

**JURUSAN TEKNOLOGI INFORMASI**

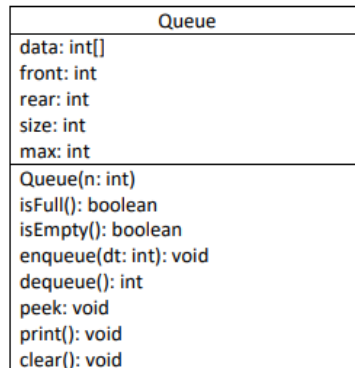
**POLITEKNIK NEGERI MALANG**

**2023/2024**

## 8.2 Praktikum 1

### 8.2.1 Langkah-langkah Percobaan

1. Perhatikan Diagram Class Queue berikut ini:



2. Berdasarkan diagram class tersebut, akan dibuat program class Queue dalam Java. Tambahkan atribut-atribut Queue sesuai diagram class, kemudian tambahkan pula konstruktornya seperti gambar berikut ini

```
1 package Praktikum1;
2 public class Queue04 {
3
4     int[] data;
5     int front;
6     int rear;
7     int size;
8     int max;
```

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

```
1 public boolean IsEmpty() {
2     if (size == 0) {
3         return true;
4     } else {
5         return false;
6     }
7 }
8 public boolean IsFull() {
9     if (size == max) {
10        return true;
11    } else {
12        return false;
13    }
14 }
```

4. Buat method peek bertipe void untuk menampilkan elemen queue pada posisi paling depan. Buat method print bertipe void untuk menampilkan seluruh elemen pada queue mulai dari posisi front sampai dengan posisi rear.

```
1 public void peek() {
2     if (!IsEmpty()) {
3         System.out.println("Elemen terdepan: " + data[front]);
4     } else {
5         System.out.println("Queue masih kosong");
6     }
7 }
8 public void print() {
9     if (IsEmpty()) {
10        System.out.println("Queue masih kosong");
11    } else {
12        int i = front;
13        while (i != rear) {
14            System.out.print(data[i] + " ");
15            i = (i + 1) % max;
16        }
17        System.out.println(data[i] + " ");
18        System.out.println("Jumlah elemen = " + size);
19    }
20 }
```

5. Buat method clear bertipe void untuk menghapus semua elemen pada queue. Buat method Enqueue bertipe void untuk menambahkan isi queue dengan parameter dt yang bertipe integer

```
1 public void clear() {
2     if (!IsEmpty()) {
3         front = rear = -1;
4         size = 0;
5         System.out.println("Queue berhasil dikosongkan");
6     } else {
7         System.out.println("Queue masih kosong");
8     }
9 }
10 public void Enqueue(int dt) {
11     if (IsFull()) {
12         System.out.println("Queue sudah penuh");
13     } else {
14         if (IsEmpty()) {
15             front = rear = 0;
16         } else {
17             if (rear == max - 1) {
18                 rear = 0;
19             } else {
20                 rear++;
21             }
22         }
23         data[rear] = dt;
24         size++;
25     }
26 }
```

6. Buat method Dequeue bertipe int untuk mengeluarkan data pada queue di posisi paling depan

```
1 public int Dequeue() {
2     int dt = 0;
3     if (IsEmpty()) {
4         System.out.println("Queue masih kosong");
5     }else {
6         dt = data[front];
7         size--;
8         if (IsEmpty()) {
9             front = rear = -1;
10        }else {
11            if (front == max -1) {
12                front = 0;
13            }else {
14                front++;
15            }
16        }
17    }
18    return dt;
19 }
20 }
```

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

```
1 package Praktikum1;
2
3 import java.util.Scanner;
4
5 /**
6  * Queue04Main
7  */
8 public class Queue04Main {
9
10     public static void menu() {
11         System.out.println("Masukkan operasi yang diinginkan:");
12         System.out.println("1. Enqueue");
13         System.out.println("2. Dequeue");
14         System.out.println("3. Print");
15         System.out.println("4. Peek");
16         System.out.println("5. Clear");
17         System.out.println("-----");
18     }
19 }
```

8. Buat fungsi main, kemudian deklarasikan Scanner dengan nama sc. . Buat variabel n untuk menampung masukan berupa jumlah maksimal elemen yang dapat disimpan pada queue.

```
1 public static void main(String[] args) {
2     Scanner sc04 = new Scanner(System.in);
3     System.out.print("Masukkan kapasitas queue:");
4     int n = sc04.nextInt();
```

9. Lakukan instansiasi objek Queue dengan nama Q dengan mengirimkan parameter n sebagai kapasitas elemen queue. Deklarasikan variabel dengan nama pilih bertipe integer untuk menampung pilih menu dari pengguna. 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.

```
1 Queue04 Q = new Queue04(n);
2 int pilih;
3
4 do{
5     menu();
6     pilih = sc04.nextInt();
7     switch (pilih) {
8         case 1:
9             System.out.print("Masukkan data baru: ");
10            int dataMasuk = sc04.nextInt();
11            Q.Enqueue(dataMasuk);
12            break;
13        case 2:
14            int dataKeluar = Q.Dequeue();
15            if (dataKeluar != 0) {
16                System.out.println("Data yang dikeluarkan: " + dataKeluar);
17                break;
18            }
19        case 3:
20            Q.print();
21            break;
22        case 4:
23            Q.peek();
24            break;
25        case 5:
26            Q.clear();
27            break;
28    }
29 } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
30 }
31 }
```

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

### 8.2.2 Verifikasi Hasil Percobaan

```
eet 10 ASD_6c2c3fd3\bin" Praktikum1.Queue04Main "  
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
```

### 8.2.3 Pertanyaan

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

Karena, diawal front dan rear tidak menunjuk index data tertentu, sehingga jika bernilai 0 maka akan menunjuk index awal sebuah array. Oleh karena itu, diawal front dan rear akan bernilai -1.

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

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

Potongan kode diatas digunakan untuk mereset nilai rear yaitu untuk mengecek apakah queue dalam kondisi kosong. jika queue masih dalam kosong data yang akan masuk menjadi data yang paling depan dan sekaligus menjadi data yang paling akhir dalam queue dan jika queue dalam kondisi tidak kosong , cek apakah posisi rear berada pada index terakhir array. jika benar, maka posisi rear selanjutnya adalah di index 0. jika posisi rear tidak berada pada index terakhir array, maka posisi rear selanjutnya adalah rear + 1.

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

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

Maksud potongan kode diatas adalah untuk mereset nilai front apabila sudah berada di index terakhir array. Sehingga nilai front akan dibalikkan ke index awal array.

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

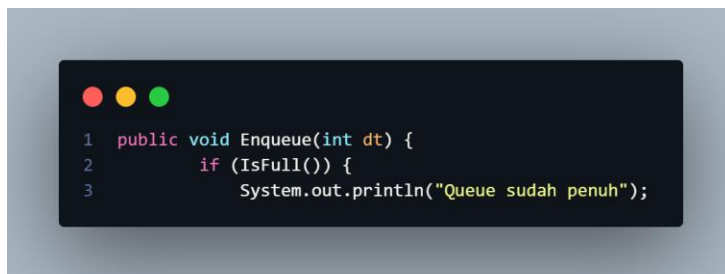
Karena, tidak semua nilai front akan dimulai dari index ke 0.

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

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

Potongan kode tersebut digunakan untuk memperbarui nilai I sebagai index yang dimana nantinya digunakan sebagai penentu pada index ke berapa informasi akan ditampilkan.

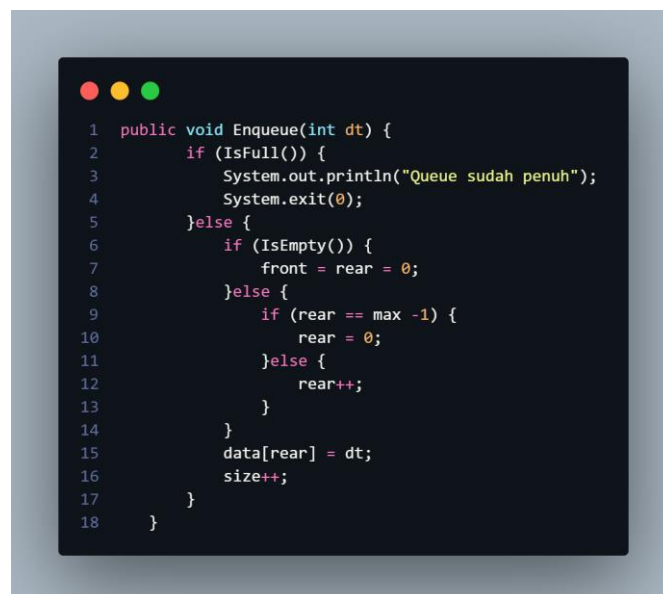
**6. Tunjukkan potongan kode program yang merupakan queue overflow!**



```
1 public void Enqueue(int dt) {
2     if (IsFull()) {
3         System.out.println("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!**

Berikut merupakan hasil modifikasi program



```
1 public void Enqueue(int dt) {
2     if (IsFull()) {
3         System.out.println("Queue sudah penuh");
4         System.exit(0);
5     }else {
6         if (IsEmpty()) {
7             front = rear = 0;
8         }else {
9             if (rear == max -1) {
10                rear = 0;
11            }else {
12                rear++;
13            }
14        }
15        data[rear] = dt;
16        size++;
17    }
18 }
```

```

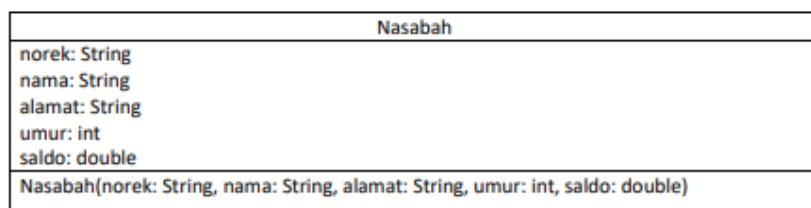
1  public int Dequeue() {
2      int dt = 0;
3      if (IsEmpty()) {
4          System.out.println("Queue masih kosong");
5      }else {
6          dt = data[front];
7          size--;
8          if (IsEmpty()) {
9              front = rear = -1;
10         }else {
11             if (front == max -1) {
12                 front = 0;
13             }else {
14                 front++;
15             }
16         }
17     }
18     return dt;
19 }
20 }

```

## 8.3 Praktikum 2

### 8.3.1 Langkah-langkah Percobaan

1. Perhatikan Diagram Class berikut ini:



2. Buat package dengan nama Praktikum2, kemudian buat class baru dengan nama Nasabah. Tambahkan atribut-atribut Nasabah seperti pada Class Diagram, kemudian tambahkan pula konstruktornya seperti gambar berikut ini

```

1  package Praktikum2;
2  public class Nasabah04 {
3      String norek;
4      String nama;
5      String alamat;
6      int umur;
7      double saldo;
8
9      Nasabah04(String norek, String nama, String alamat, int umur, double saldo) {
10         this.norek = norek;
11         this.nama = nama;
12         this.alamat = alamat;
13         this.umur = umur;
14         this.saldo = saldo;
15     }
16
17     Nasabah04() {
18
19     }
20 }

```



3. Salin kode program class Queue pada Praktikum 1 untuk digunakan kembali pada Praktikum 2 ini. 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 Queue tersebut. Lakukan modifikasi pada class Queue dengan mengubah tipe `int[]` data menjadi `Nasabah[]` data karena pada kasus ini data yang akan disimpan pada queue berupa object `Nasabah`. Modifikasi perlu dilakukan pada atribut, method `Enqueue`, dan method `Dequeue`

```
1 package Praktikum2;
2 public class QueueNasabah04 {
3     Nasabah01[] data;
4     int front;
5     int rear;
6     int size;
7     int max;
8
9     public QueueNasabah04(int n) {
10         max = n;
11         data = new Nasabah04[max];
12         size = 0;
13         front = rear = -1;
14     }
15
16     public void Enqueue(Nasabah04 dt) {
17         if (IsFull()) {
18             System.out.println("Queue sudah penuh");
19         } else {
20             if (IsEmpty()) {
21                 front = rear = 0;
22             } else {
23                 if (rear == max - 1) {
24                     rear = 0;
25                 } else {
26                     rear++;
27                 }
28             }
29             data[rear] = dt;
30             size++;
31         }
32     }
33
34     public Nasabah04 Dequeue() {
35         Nasabah04 dt = new Nasabah04();
36         if (IsEmpty()) {
37             System.out.println("Queue masih kosong");
38         } else {
39             dt = data[front];
40             size--;
41             if (IsEmpty()) {
42                 front = rear = -1;
43             } else {
44                 if (front == max - 1) {
45                     front = 0;
46                 } else {
47                     front++;
48                 }
49             }
50         }
51         return dt;
52     }
53 }
```

Baris program `Nasabah dt = new Nasabah();` akan ditandai sebagai error, untuk mengatasinya, tambahkan konstruktor default di dalam class `Nasabah`.



4. Karena satu elemen queue terdiri dari beberapa informasi (norek, nama, alamat, umur, dan saldo), maka ketika mencetak data juga perlu ditampilkan semua informasi tersebut, sehingga modifikasi perlu dilakukan pada method `peek` dan method `print`



5. Selanjutnya, buat class baru dengan nama `QueueMain` tetap pada package `Praktikum2`. Buat method `menu` untuk mengakomodasi pilihan menu dari masukan pengguna



6. Buat fungsi main, deklarasikan Scanner dengan nama sc. Buat variabel max untuk menampung kapasitas elemen pada queue. Kemudian lakukan instansiasi objek queue dengan nama antri dan nilai parameternya adalah variabel jumlah.

```
1 public static void main(String[] args) {
2     Scanner sc04 = new Scanner(System.in);
3
4     System.out.print("Masukkan kapasitas queue: ");
5     int jumlah = sc04.nextInt();
6     QueueNasabah04 antri = new QueueNasabah04(jumlah);
```

7. Deklarasikan variabel dengan nama pilih bertipe integer untuk menampung pilih menu dari pengguna. Tambahkan kode berikut untuk melakukan perulangan menu sesuai dengan masukan yang diberikan oleh pengguna.

```
1 int pilih;
2 do {
3     menu();
4     pilih = sc04.nextInt();
5
6     switch (pilih) {
7         case 1:
8             System.out.print("No. Rekening : ");
9             String norek = sc04.nextLine();
10            sc04.nextLine();
11            System.out.print("Nama : ");
12            String nama = sc04.nextLine();
13            System.out.print("Alamat : ");
14            String alamat = sc04.nextLine();
15            System.out.print("Umur : ");
16            int umur = sc04.nextInt();
17            System.out.print("Saldo : ");
18            double saldo = sc04.nextDouble();
19
20            Nasabah04 nb = new Nasabah04(norek, nama, alamat, umur, saldo);
21            sc04.nextLine();
22            antri.Enqueue(nb);
23            break;
24        case 2:
25            Nasabah04 data = antri.Dequeue();
26            if (!"".equals(data.norek) && !"".equals(data.nama) && !"".equals(data.alamat)
27                && data.umur != 0 && data.saldo != 0) {
28                System.out.println("Antrian yang keluar: " + data.norek + " " + data.nama + " "
29                    + data.alamat + " " + data.umur + " " + data.saldo);
30                break;
31            }
32        case 3:
33            antri.peek();
34            break;
35        case 4:
36            antri.print();
37            break;
38        case 5:
39            antri.peekRear();
40            break;
41    }
42    } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4);
43 }
44 }
```

8. Compile dan jalankan class QueueMain, kemudian amati hasilnya

### 8.3.2 Verifikasi Hasil Percobaan

```
Masukkan kapasitas queue: 4
Pilih Menu:
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
-----
1
No. Rekening : 1200046675
Nama : Arif
Alamat : Sukun, Malang
Umur : 25
Saldo : 12000000
Pilih Menu:
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
-----
1
No. Rekening : 1200198733
Nama : Dewi
Alamat : Rungkut, Surabaya
Umur : 30
Saldo : 86000000
Pilih Menu:
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
-----
4
Arif Sukun, Malang 25 1.2E7 Dewi Rungkut, Surabaya 30 8
6000000.0
Jumlah elemen = 2
```

```
Pilih Menu:
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
-----
3
Elemen terdepan: Arif Sukun, Malang 25 1.2E7
Pilih Menu:
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
-----
2
Elemen terdepan: Dewi Rungkut, Surabaya 30 86000000.0
Pilih Menu:
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
-----
4
Dewi Rungkut, Surabaya 30 86000000.0
Jumlah elemen = 1
```

### 8.3.3 Pertanyaan

#### 1. Pada class QueueMain, jelaskan fungsi IF pada potongan kode program berikut!

```
if (!"".equals(data.norek) && !"".equals(data.nama) && !"".equals(data.alamat)
    && data.umur != 0 && data.saldo != 0) {
    System.out.println("Antrian yang keluar: " + data.norek + " " + data.nama + " "
        + data.alamat + " " + data.umur + " " + data.saldo);
    break;
}
```

Fungsi if digunakan untuk mencocokkan jika seluruh data tidak kosong, sehingga data yang dikeluarkan terdapat nilai

2. 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 5. Cek Antrian paling belakang pada class QueueMain sehingga method peekRear dapat dipanggil!

```
public void peekRear() {
    if (!isEmpty()) {
        System.out.println("Elemen terbelakang: " + data[rear].norek + " " + data[rear].nama + " " + data[rear].alamat + " " + data[rear].umur + " " + data[rear].saldo);
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}
```

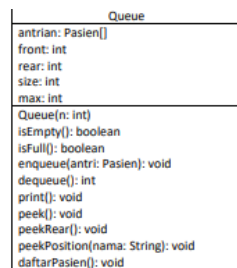
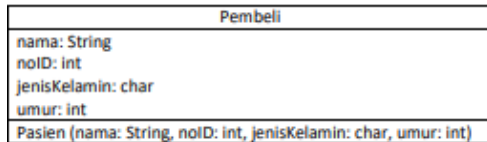
```
case 5:
    antri.peekRear();
    break;
```

Berikut hasil running setelah di modifikasi

```
Masukkan kapasitas queue: 2
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian keluar
4. Cek semua antrian
5. Cek antrian paling belakang
-----
1
No. Rekening: 12345
Nama: tera
Alamat: kalam
Umur: 23
Saldo: 23000000
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian keluar
4. Cek semua antrian
5. Cek antrian paling belakang
5. Cek antrian paling belakang
-----
1
No. Rekening: 43113
Nama: herra
Alamat: madura
Umur: 43
Saldo: 132000000
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian keluar
4. Cek semua antrian
5. Cek antrian paling belakang
-----
5
Elemen terbelakang: herra madura 43 1.32E8
```

## 8.4 Tugas

1. Buatlah program antrian untuk mengilustrasikan antrian pasien di sebuah klinik. Ketika seorang pasien akan mengantri, maka dia harus mendaftarkan nama, nomor identitas, jenis kelamin dan umur seperti yang digambarkan pada Class diagram berikut:



Keterangan method:

- Method create(), isEmpty(), isFull(), enqueue(), dequeue() dan print(), kegunaannya sama seperti yang telah dibuat pada Praktikum
- Method peek(): digunakan untuk menampilkan data Pasien yang berada di posisi antrian paling depan Method peekRear(): digunakan untuk menampilkan data Pasien yang berada di posisi antrian paling belakang
- Method peekPosition(): digunakan untuk menampilkan seorang pasien (berdasarkan nama) posisi antrian ke berapa
- Method daftarPasien(): digunakan untuk menampilkan data seluruh pasien

## Jawab

Berikut merupakan kode program dari studi kasus di atas

```
1 package Praktikum3;
2 public class Pasien04 {
3     String nama;
4     int noID, umur;
5     char jenisKelamin;
6
7
8     public Pasien04(String nama, int noID, char jenisKelamin, int umur) {
9         this.nama = nama;
10        this.noID = noID;
11        this.jenisKelamin = jenisKelamin;
12        this.umur = umur;
13    }
14 }
```

```

1 package Praktikum3;
2 public class QueuePasien04 {
3     Pasien04[] antrian;
4     int front, rear, size, max;
5
6     public QueuePasien04(int n) {
7         max = n;
8         antrian = new Pasien04[max];
9         size = 0;
10        front = rear = -1;
11    }
12
13    public boolean isEmpty() {
14        return size == 0;
15    }
16
17    public boolean isEmpty() {
18        return size == 0;
19    }
20
21    public boolean isFull() {
22        return size == max;
23    }
24
25    public void enqueue(Pasien04 p) {
26        if (isFull()) {
27            System.out.println("Antrian sudah penuh");
28        } else {
29            if (isEmpty()) {
30                front = rear = 0;
31            } else {
32                rear = (rear + 1) % max;
33            }
34            antrian[rear] = p;
35            size++;
36            System.out.println("Pasien " + p.nama + " berhasil ditambahkan ke dalam antrian");
37        }
38    }
39
40    public Pasien04 dequeue() {
41        Pasien04 p = null;
42        if (isEmpty()) {
43            System.out.println("Antrian masih kosong");
44        } else {
45            p = antrian[front];
46            if (front == rear) {
47                front = rear = -1;
48            } else {
49                front = (front + 1) % max;
50            }
51            size--;
52        }
53        return p;
54    }
55
56    public void print() {
57        if (isEmpty()) {
58            System.out.println("Antrian masih kosong");
59        } else {
60            int i = front;
61            while (i != rear) {
62                System.out.println(antrian[i].nama + " " + antrian[i].noID + " " + antrian[i].jenisKelamin + " " + antrian[i].umur);
63                i = (i + 1) % max;
64            }
65            System.out.println(antrian[i].nama + " " + antrian[i].noID + " " + antrian[i].jenisKelamin + " " + antrian[i].umur);
66        }
67    }
68
69    public void peek() {
70        if (!isEmpty()) {
71            System.out.println("Pasien terdepan: " + antrian[front].nama);
72        } else {
73            System.out.println("Antrian masih kosong");
74        }
75    }
76
77    public void peekRear() {
78        if (!isEmpty()) {
79            System.out.println("Pasien terbelakang: " + antrian[rear].nama);
80        } else {
81            System.out.println("Antrian masih kosong");
82        }
83    }
84
85    public void peekPosition(String nama) {
86        if (!isEmpty()) {
87            for (int i = front; i != rear; i = (i + 1) % max) {
88                if (antrian[i].nama.equals(nama)) {
89                    System.out.println("Posisi antrian pasien " + nama + " : " + ((i - front + max) % max + 1));
90                    return;
91                }
92            }
93            if (antrian[rear].nama.equals(nama)) {
94                System.out.println("Posisi antrian pasien " + nama + " : " + ((rear - front + max) % max + 1));
95            } else {
96                System.out.println("Pasien " + nama + " tidak ditemukan dalam antrian");
97            }
98        } else {
99            System.out.println("Antrian masih kosong");
100        }
101    }
102
103    public void daftarPasien() {
104        if (!isEmpty()) {
105            System.out.println("Daftar Pasien:");
106            int i = front;
107            while (i != rear) {
108                System.out.println(antrian[i].nama + " " + antrian[i].noID + " " + antrian[i].jenisKelamin + " " + antrian[i].umur);
109                i = (i + 1) % max;
110            }
111            System.out.println(antrian[i].nama + " " + antrian[i].noID + " " + antrian[i].jenisKelamin + " " + antrian[i].umur);
112        } else {
113            System.out.println("Antrian masih kosong");
114        }
115    }
116 }

```

```

1 package Praktikum3;
2 import java.util.Scanner;
3
4 public class QueuePasien04Main {
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Masukkan kapasitas antrian: ");
8         int capacity = sc.nextInt();
9         QueuePasien04 antrian = new QueuePasien04(capacity);
10
11         int pilihan;
12         do {
13             System.out.println("=====Pilihan Menu:=====");
14             System.out.println("1. Tambah Pasien");
15             System.out.println("2. Hapus Pasien Terdepan");
16             System.out.println("3. Lihat Pasien Terdepan");
17             System.out.println("4. Lihat Pasien Terbelakang");
18             System.out.println("5. Cari Posisi Pasien");
19             System.out.println("6. Daftar Pasien");
20             System.out.println("7. Keluar");
21             System.out.println("=====");
22             System.out.print("Pilihan: ");
23             pilihan = sc.nextInt();
24
25             switch (pilihan) {
26                 case 1:
27                     sc.nextLine();
28                     System.out.print("Nama: ");
29                     String nama = sc.nextLine();
30                     System.out.print("Nomor Identitas: ");
31                     int noID = sc.nextInt();
32                     System.out.print("Jenis Kelamin (L/P): ");
33                     char jenisKelamin = sc.next().charAt(0);
34                     System.out.print("Umur: ");
35                     int umur = sc.nextInt();
36
37                     Pasien04 pb = new Pasien04(nama, noID, jenisKelamin, umur);
38                     sc.nextLine();
39                     antrian.enqueue(pb);
40                     break;
41                 case 2:
42                     Pasien04 pasienHapus = antrian.dequeue();
43                     if (pasienHapus != null) {
44                         System.out.println("Pasien " + pasienHapus.nama + " berhasil dihapus dari antrian");
45                     }
46                     break;
47                 case 3:
48                     antrian.peek();
49                     break;
50                 case 4:
51                     antrian.peekRear();
52                     break;
53                 case 5:
54                     sc.nextLine();
55                     System.out.print("Nama Pasien: ");
56                     String namaCari = sc.nextLine();
57                     antrian.peekPosition(namaCari);
58                     break;
59                 case 6:
60                     antrian.daftarPasien();
61                     break;
62                 case 7:
63                     System.out.println("Terima kasih, program selesai.");
64                     break;
65                 default:
66                     System.out.println("Pilihan tidak valid");
67             }
68         } while (pilihan != 7);
69
70         sc.close();
71     }
72 }

```



## Hasil running

- Pilihan menu dan penambahan pasien

```
Masukkan kapasitas antrian: 3
=====Pilihan Menu:=====
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=====
Pilihan: 1
Nama: hilwana
Nomor Identitas: 001
Jenis Kelamin (L/P): P
Umur: 12
Pasien hilwana berhasil ditambahkan ke dalam antrian
=====Pilihan Menu:=====
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=====
Pilihan: 1
Nama: Celine
Nomor Identitas: 002
Jenis Kelamin (L/P): P
Umur: 19
Pasien Celine berhasil ditambahkan ke dalam antrian
```

```
=====
Pilihan: 1
Nama: Rizky
Nomor Identitas: 003
Jenis Kelamin (L/P): L
Umur: 20
Pasien Rizky berhasil ditambahkan ke dalam antrian
```

- Menghapus pasien terdepan

```
=====Pilihan Menu:=====
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=====
Pilihan: 2
Pasien hilwana berhasil dihapus dari antrian
```

- Lihat pasien terdepan

```
=====Pilihan Menu:=====
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=====
Pilihan: 3
Pasien terdepan: Celine
```

- Lihat pasien terbelakang

```
Pasien terdepan: Celine
=====Pilihan Menu:=====
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=====
Pilihan: 4
Pasien terbelakang: Rizky
```

- Cari posisi pasien

```
=====Pilihan Menu:=====
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=====
Pilihan: 5
Nama Pasien: Celine
Posisi antrian pasien Celine : 1
=====Pilihan Menu:=====
```

- Daftar pasien

```
=====Pilihan Menu:=====
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=====
Pilihan: 6
Daftar Pasien:
Celine 2 P 19
Rizky 3 L 20
```

- Keluar

```
=====Pilihan Menu:=====
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=====
Pilihan: 7
Terima kasih, program selesai.
C:\Users\user\Documents\Jobsheet 10 ASD>
```

Bu mohon maaf saya kemarin sudah mengumpulkan tugas tetapi ada sebagian kesalahan sehingga belum selesai, mau saya benarkan tetapi laptop saya tiba-tiba mati tidak bisa nyala sehingga baru bisa menambahkan sekarang, mohon maaf keterlambatan pengumpulan ulang dan pengupload an di github 🙏