LAPORAN PRAKTIKUM

MATA KULIAH ALGORITMA DAN STRUKTUR DATA

PERTEMUAN 10 : QUEUE

KAYLA RACHMAUDINA SATITI PUTRI

2341760103

D-IV SISTEM INFORMASI BISNIS

JURUSAN TEKNOLOGI INFORMASI POLITEKNIK

NEGERI MALANG

2024

JOBSHEET VIII
QUEUE

8.1 Tujuan Praktikum

Setelah melakukan materi praktikum ini, mahasiswa mampu:

1. Mengenal struktur data Queue
2. Membuat dan mendeklarasikan struktur data Queue
3. Menerapkan algoritma Queue dengan menggunakan array

8.2 Praktikum 1

Waktu percobaan : 45 menit

Pada percobaan ini, kita akan mengimplementasikan penggunaan class Queue.

8.2.1 Langkah-langkah Percobaan

1. Perhatikan Diagram Class Queue berikut ini:

Queue
data: int[] front: int rear: int size: int max: int
Queue(n: int) isFull(): boolean isEmpty(): boolean enqueue(dt: int): void dequeue(): int peek: void print(): void clear(): void

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

2. Buat package dengan nama **Praktikum1**, kemudian buat class baru dengan nama **Queue**.
3. Tambahkan atribut-atribut Queue sesuai diagram class,

```java
public class Queue15 {
    int[] data;
    int front;
    int rear;
    int size;
    int max;

    public Queue15(int n) {
        max = n;
        data = new int[max];
        size = 0;
        front = rear = -1;
    }
```

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

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

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

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

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

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

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

8.

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

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

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

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

```
    public void Enqueue(int dt) {
        if (IsFull()) {
            System.out.println("Queue sudah penuh");
        } else {
            if (IsEmpty()) {
                front = rear = 0;
            } else {
                if (rear == max - 1) {
                    rear = 0;
                } else {
                    rear++;
                }
            }
            data[rear] = dt;
            size++;
        }
    }
```

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

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

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

```
1  import java.util.Scanner;
2
3  public class QueueMain15 {
4      public static void menu() {
5          System.out.println("-----------------------------------");
6          System.out.println("Masukkan operasi yang diinginkan : ");
7          System.out.println("1. Enqueue");
8          System.out.println("2. Dequeue");
9          System.out.println("3. Print");
10         System.out.println("4. Peek");
11         System.out.println("5. Clear");
12         System.out.println("-----------------------------------");
13     }
```

13. Buat fungsi **main**, kemudian deklarasikan Scanner dengan nama **sc**.

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

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

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

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

```
20         Queue15 Q = new Queue15(n);
```

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

```
21         int pilih;
```

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

```
        do {
            menu();
            pilih = sc.nextInt();
            switch (pilih) {
                case 1:
                    System.out.print("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;
                default:
                    break;
            }
        } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
    }
}
```

18. Compile dan jalankan class **QueueMain**, kemudian amati hasilnya.

8.2.2 Verifikasi Hasil Percobaan

```
Masukkan kapasitas queue : 6
-----------------------------------
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 : 23
-----------------------------------
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----------------------------------
3
15
23
Jumlah elemen = 2
-----------------------------------
```

```
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
-----------------------------------
2
Data yang dikeluarkan : 15
-----------------------------------
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----------------------------------
3
23
Jumlah elemen = 1
```

8.2.3 Pertanyaan

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

   Menandakan keadaan awal yang kosong.

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

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

   Untuk mengatur penunjuk rear dalam antrian selama operasi berjalan dan memastikan bahwa penunjuk rear menunjuk dg benar ke slot kosong berikutnya untuk memasukkan elemen baru.

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

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

Untuk mengatur penunjuk front dalam antrian selama operasi berlangsung dan memastikan bahwa penunjuk front menunjuk dengan benar ke elemen berikutnya.

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

Karena front belum tentu berada di indeks ke-0, sedangkan perulangan sendiri dimulai dengan posisi front.

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

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

Apabila i tidak berfungsi sebagai rear, maka increment akan dilakukan dimodulus dengan nilai max atau kapasitas queue nya.

6. Tunjukkan potongan kode program yang merupakan queue overflow!

```
33      public void Enqueue(int dt) {
34          if (IsFull()) {
35              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!

```
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      }
19
20      public Nasabah15 Dequeue() {
21          int dt = 0;
22          if (IsEmpty()) {
23              System.out.println("Queue masih kosong");
24          } else {
25              dt = data[front];
26              size--;
27              if (IsEmpty()) {
28                  front = rear = -1;
29              } else {
30                  if (front == max - 1) {
31                      front = 0;
32                  } else {
33                      front++;
34                  }
35              }
36          }
37          return dt;
38      }
39  }
```

8.3 Praktikum 2

Waktu percobaan : 45 menit

Pada percobaan ini, kita akan membuat program yang mengilustrasikan teller di bank dalam melayani nasabah.

8.3.1 Langkah-langkah Percobaan

1. Perhatikan Diagram Class berikut ini:

Nasabah
norek: String nama: String alamat: String umur: int saldo: double
Nasabah(norek: String, nama: String, alamat: String, umur: int, saldo: double)

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

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

```java
public class Nasabah15 {
    String norek, nama, alamat;
    int umur;
    double saldo;

    Nasabah15(String norek, String nama, String alamat, int umur, double saldo) {
        this.norek = norek;
        this.nama = nama;
        this.alamat = alamat;
        this.umur = umur;
        this.saldo = saldo;
    }
```

4. 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.
5. 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**.

```
14      Nasabah15[] data;
15      int front;
16      int rear;
17      int size;
18      int max;
```

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

```
20      Nasabah15() {}
```

6. 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 meodifikasi perlu dilakukan pada method **peek** dan method **print**.

```
public void peek() {
    if (!IsEmpty()) {
        System.out.println("Elemen terdepan: " + data[front].norek + " " + data[front].nama + " "
                + data[front].alamat + " " + data[front].umur  + " " + data[front].saldo);
    } 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(data[i].norek + " " + data[i].nama + " " + data[i].alamat + " " + data[i].umur + " " + data[i].saldo);
            i = (i + 1) % max;
        }
        System.out.println(data[i].norek + " " + data[i].nama + " " + data[i].alamat + " " + data[i].umur + " " + data[i].saldo);
        System.out.println("Jumlah elemen = " + size);
    }
}
```

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

```
public static void menu() {
    System.out.println("------------------------------");
    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("------------------------------");
}
```

8. Buat fungsi **main**, deklarasikan Scanner dengan nama **sc**

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

9. Buat variabel **max** untuk menampung kapasitas elemen pada queue. Kemudian lakukan instansiasi objek queue dengan nama **antri** dan nilai parameternya adalah variabel **jumlah**.

```
        System.out.println();
        System.out.print("Masukkan kapasitas queue: ");
        int jumlah = sc.nextInt();
        Queue15 antri = new Queue15(jumlah);
```

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

```
        int pilih;
```

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

```
        do {
            menu();
            pilih = sc.nextInt();
            switch (pilih) {
                case 1:
                    System.out.print("No Rekening : ");
                    String norek = sc.next();
                    System.out.print("Nama        : ");
                    String nama = sc.next();
                    System.out.print("Alamat      : ");
                    String alamat = sc.next();
                    System.out.print("Umur        : ");
                    int umur = sc.nextInt();
                    System.out.print("Saldo       : ");
                    Double saldo = sc.nextDouble();
                    Nasabah15 nb = new Nasabah15(norek, nama, alamat, umur, saldo);
                    antri.Enqueue(nb);
                    break;
                case 2:
                    Nasabah15 data = antri.Dequeue();
                    if (!"".equals(data.norek) && !"".equals(data.nama) && !"".equals(data.alamat)
                    && !"".equals(data.umur) && !"".equals(data.saldo)) {
                        System.out.println("Antrian yang keluar: " + data.norek + " " + data.nama + " "
                                           + data.alamat + " "+ data.umur  + " " + data.saldo);
                        break;
                    }
                case 3:
                    antri.peek();
                    break;
                case 4:
                    antri.print();
                    break;
            }
        } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4);
    }
}
```

12. Compile dan jalankan class **QueueMain**, kemudian amati hasilnya.

8.3.2 Verifikasi Hasil Percobaan

Samakan hasil compile kode program Anda dengan gambar berikut ini.

```
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       : 8600000
------------------------------
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
------------------------------
4
1200046675 Arif Sukun,Malang 25 1.2E7
1200198733 Dewi Rungkut,Surabaya 30 8600000.0
Jumlah elemen = 2
```

```
------------------------------
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
------------------------------
3
Elemen terdepan: 1200046675 Arif Sukun,Malang 25 1.2E7
------------------------------
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
------------------------------
2
Antrian yang keluar: 1200046675 Arif Sukun,Malang 25 1.2E7
------------------------------
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
------------------------------
4
1200198733 Dewi Rungkut,Surabaya 30 8600000.0
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 mengecek apakah semua data pada objek data sudah terisi dengan nilai yang valid. Jika semua data sudah terisi, maka kode di dalam blok IF akan dijalankan.

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("Queue masih kosong");
    }
}
```

```
        System.out.println("------------------------------");
        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("------------------------------");
```

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

```
------------------------------
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
5. Cek antrian belakang
------------------------------
1
No Rekening : 111111
Nama        : Ahmad
Alamat      : Malang
Umur        : 29
Saldo       : 2000000
------------------------------
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
5. Cek antrian belakang
------------------------------
1
No Rekening : 222222
Nama        : Bambang
Alamat      : Surabaya
Umur        : 49
Saldo       : 1000000
------------------------------
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
5. Cek antrian belakang
------------------------------
5
Elemen terbelakang : 222222  Bambang Surabaya 49 1000000.0
```

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:

Pembeli
nama: String noID: int jenisKelamin: char umur: int
Pasien (nama: String, noID: int, jenisKelamin: char, umur: int)

Class diagram Queue digambarkan sebagai berikut:

Queue
antrian: Pasien[] front: int rear: int size: int max: int
Queue(n: int) isEmpty(): boolean isFull(): boolean enqueue(antri: Pasien): void dequeue(): int print(): void peek(): void peekRear(): void peekPosition(nama: String): void daftarPasien(): void

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

```java
package Tugas15;

public class Pembeli15 {
    String nama;
    int noId, umur;
    char jk;

    Pembeli15(){}

    Pembeli15 (String nama, int noId, char jk, int umur) {
        this.nama = nama;
        this.noId = noId;
        this. jk = jk;
        this.umur = umur;
    }
}
```

```java
package Tugas15;

public class Queue15 {
    Pembeli15[] data;
    int front;
    int rear;
    int size;
    int max;

    public Queue15(int n) {
        max = n;
        data = new Pembeli15[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: " + data[front].nama + " " + data[front].noId
                                    + " "  + data[front].jk   + " " + data[front].umur);
        } else {
            System.out.println("Queue masih kosong");
        }
    }

    public void peekRear(){
        if(!IsEmpty()){
            System.out.println("Elemen terbelakang: " + data[rear].nama + " " + data[rear].noId
                                    + " " + data[rear].jk   + " " + data[rear].umur);
        }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("Elemen terdepan: " + data[front].nama + " " + data[front].noId
                                        + " "  + data[front].jk   + " " + data[front].umur);
                i = (i + 1) % max;
            }
            System.out.println("Elemen terdepan: " + data[front].nama + " " + data[front].noId
                                    + " "  + data[front].jk   + " " + data[front].umur);
            System.out.println("Jumlah elemen = " + size);
        }
    }
```

```
77      public void Enqueue(Pembeli15 dt) {
78          if (IsFull()) {
79              System.out.println("Queue sudah penuh");
80          } else {
81              if (IsEmpty()) {
82                  front = rear = 0;
83              } else {
84                  if (rear == max - 1) {
85                      rear = 0;
86                  } else {
87                      rear++;
88                  }
89              }
90              data[rear] = dt;
91              size++;
92          }
93      }
94
95      public Pembeli15 Dequeue() {
96          Pembeli15 dt = new Pembeli15();
97          if (IsEmpty()) {
98              System.out.println("Queue masih kosong");
99          } else {
100             dt = data[front];
101             size--;
102             if (IsEmpty()) {
103                 front = rear = -1;
104             } else {
105                 if (front == max - 1) {
106                     front = 0;
107                 } else {
108                     front++;
109                 }
110             }
111         }
112         return dt;
113     }
114
115     public int peekPosition(String nama) {
116         if (IsEmpty()) {
117             return -1;
118         }
119         int i = front;
120         int position = 0;
121         while (i != rear) {
122             if (data[i].nama.equals(nama)) {
123                 System.out.println("Nama: " + data[i].nama + ", No. ID: " + data[i].noId +
124                                    ", JK: "    + data[i].jk   + ", Umur: "   + data[i].umur);
125                 return position;
126             }
127             position++;
128             i = (i + 1) % max;
129             }
130             if (data[i].nama.equals(nama)) {
131                 System.out.println("Nama: " + data[i].nama + ", No. ID: " + data[i].noId +
132                                    ", JK: "    + data[i].jk   + ", Umur: "   + data[i].umur);
133             return position;
134             }
135             return -1;
136         }
137
138         public void daftarPasien() {
139             if (IsEmpty()) {
140             System.out.println("Daftar pasien masih kosong");
141             } else {
142             int i = front;
143             while (i != rear) {
144                 System.out.println("Nama: " + data[i].nama + ", No. ID: " + data[i].noId
145                                    + ", JK: "  + data[i].jk   + ", Umur: "   + data[i].umur);
146                 i = (i + 1) % max;
147             }
148
149             System.out.println("Nama: " + data[i].nama + ", No. ID: " + data[i].noId
150                                + ", JK: "  + data[i].jk   + ", Umur: "   + data[i].umur);
151         }
152     }
153 }
```

```java
package Tugas15;

import java.util.Scanner;

public class QueueMain15 {
    public static void menu() {
        System.out.println("--------------------------------");
        System.out.println("Pilih menu: ");
        System.out.println("1. Pasien baru");
        System.out.println("2. Pasien keluar");
        System.out.println("3. Daftar Semua Pasien");
        System.out.println("4. Cek Pasien terdepan");
        System.out.println("5. Cek Pasien belakang");
        System.out.println("6. Cek Pasien berdasarkan nama");
        System.out.println("--------------------------------");
    }

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

        System.out.println();
        System.out.print("Masukkan kapasitas queue: ");
        int jumlah = sc.nextInt();

        Queue15 antri = new Queue15(jumlah);
        int pilih;

        do {
            menu();
            pilih = sc.nextInt();
            switch (pilih) {
                case 1:
                    System.out.print("Nama              : ");
                    String nama = sc.next();
                    System.out.print("No ID             : ");
                    int noId = sc.nextInt();
                    System.out.print("Jenis Kelamin (L/P): ");
                    String jk =  sc.next();
                    System.out.print("Umur              : ");
                    int umur = sc.nextInt();
                    Pembeli15 nb = new Pembeli15(nama, noId, jk.charAt(0), umur);
                    antri.Enqueue(nb);
                    break;
                case 2:
                    Pembeli15 data = antri.Dequeue();
                    if (data.nama.isEmpty() || data.noId == 0
                            || data.umur == 0) {
                        System.out.println("Pembeli masih kosong");
                    } else {
                        System.out.println("Pembeli yang keluar: " + data.nama + " " + data.noId + " "
                                + data.jk + " " + data.umur );
                    }
                    break;
                case 3:
                    antri.daftarPasien();
                    break;
                case 4:
                    antri.peek();
                    break;
                case 5:
                    antri.peekRear();
                    break;
                case 6:
                    System.out.print("Masukkan Nama :");
                    String getNama = sc.next();
                    antri.peekPosition(getNama);
                    break;
            }
        } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5 || pilih == 6);
    }
}
```

Output :

```
Masukkan kapasitas queue: 4
--------------------------------
Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Daftar Semua Pasien
4. Cek Pasien terdepan
5. Cek Pasien belakang
6. Cek Pasien berdasarkan nama
--------------------------------
1
Nama               : kela
No ID              : 555
Jenis Kelamin (L/P): p
Umur               : 15
--------------------------------
Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Daftar Semua Pasien
4. Cek Pasien terdepan
5. Cek Pasien belakang
6. Cek Pasien berdasarkan nama
--------------------------------
1
Nama               : cipung
No ID              : 222
Jenis Kelamin (L/P): l
Umur               : 3
--------------------------------
Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Daftar Semua Pasien
4. Cek Pasien terdepan
5. Cek Pasien belakang
6. Cek Pasien berdasarkan nama
--------------------------------
```

```
Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Daftar Semua Pasien
4. Cek Pasien terdepan
5. Cek Pasien belakang
6. Cek Pasien berdasarkan nama
--------------------------------
1
Nama               : abe
No ID              : 333
Jenis Kelamin (L/P): l
Umur               : 5
--------------------------------
Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Daftar Semua Pasien
4. Cek Pasien terdepan
5. Cek Pasien belakang
6. Cek Pasien berdasarkan nama
--------------------------------
3
Nama: kela, No. ID: 555, JK: p, Umur: 15
Nama: cipung, No. ID: 222, JK: l, Umur: 3
Nama: abe, No. ID: 333, JK: l, Umur: 5
--------------------------------
Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Daftar Semua Pasien
4. Cek Pasien terdepan
5. Cek Pasien belakang
6. Cek Pasien berdasarkan nama
--------------------------------
4
Elemen terdepan: kela 555 p 15
```


POLITEKNIK NEGERI MALANG