# LAPORAN PRAKTIKUM JOBSHEET

# QUEUE

# MATA KULIAH ALGORITMA DAN STRUKTUR DATA


**Disusun Oleh :**

**Jami'atul Afifah (2341760102)**

**SIB-1F**

**PROGRAM STUDI D4 SISTEM INFOEMASI BISNIS**

**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI MALANG**

**2024**

# PRAKTIKUM 1

1. Perhatikan Diagram Class Queue berikut ini:

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

   Berdasarkan diagram class tersebut, akan dibuat program class Queue dalam Jawa.
2. Buat package dengan nama Praktikum1, kemudian buat class baru dengan nama Queue.
3. Tambahkan atribut-atribut Queue sesuai diagram class, kemudian tambahkan pula konstruktornya seperti gambar berikut ini.

```
package Praktikum1;
import java.util.Scanner;

class Queue {
    int[] data;
    int front;
    int rear;
    int size;
    int max;

    public Queue(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.

```
public boolean IsEmpty() {
    return size == 0;
}
```

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

```
public boolean IsFull() {
    return size == max;
}
```

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

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

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

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

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

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

9. 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(x:"Queue sudah penuh");
    } else {
        if (IsEmpty()) {
            front = rear = 0;
        } else {
            if (rear == max - 1) {
                rear = 0;
            } else {
                rear++;
            }
        }
        data[rear] = dt;
        size++;
    }
}
```

10. 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(x:"Queue masih kosong");
    } else {
        dt = data[front];
        size--;
        if (IsEmpty()) {
            front = rear = -1;
        } else {
            if (front == max - 1) {
                front = 0;
            } else {
                front++;
            }
        }
    }
    return dt;
}
```

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

```
package Praktikum1;
import java.util.Scanner;

public class QueueMain {
    public static void menu() {
        System.out.println(x:"Masukkan operasi yang diinginkan:");
        System.out.println(x:"1. Enqueue");
        System.out.println(x:"2. Dequeue");
        System.out.println(x:"3. Print");
        System.out.println(x:"4. Peek");
        System.out.println(x:"5. Clear");
        System.out.println(x:"---------------------------------");
    }
```

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

```
package Praktikum1;
import java.util.Scanner;

public class QueueMain {
    public static void menu() {
        System.out.println(x:"Masukkan operasi yang diinginkan:");
        System.out.println(x:"1. Enqueue");
        System.out.println(x:"2. Dequeue");
        System.out.println(x:"3. Print");
        System.out.println(x:"4. Peek");
        System.out.println(x:"5. Clear");
        System.out.println(x:"---------------------------------");
    }
```

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

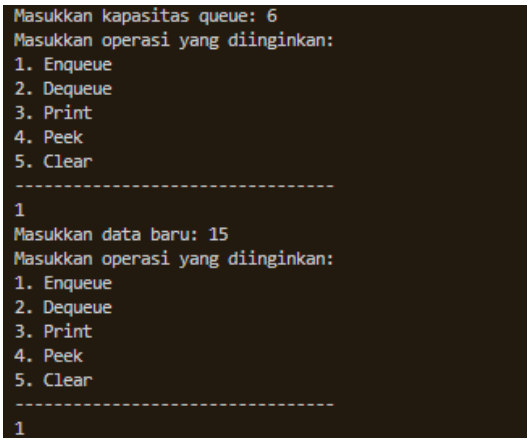
```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print(s:"Masukkan kapasitas queue: ");
    int n = sc.nextInt();
```

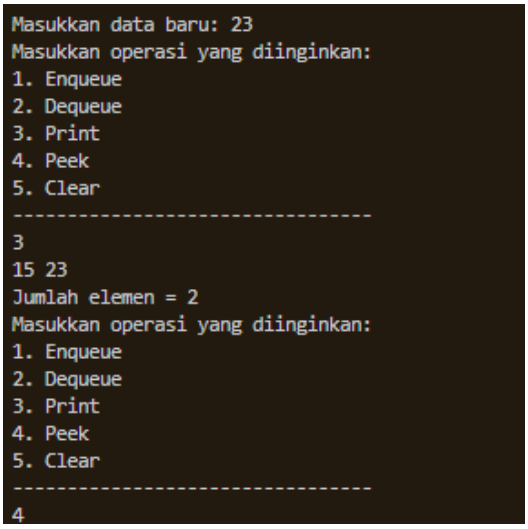
14. Lakukan instansiasi objek Queue dengan nama Q dengan mengirimkan parameter n sebagai kapasitas elemen queue
    Queue Q = new Queue(n);
15. Deklarasikan variabel dengan nama pilih bertipe integer untuk menampung pilih menu dari pengguna.
16. Lakukan perulangan menggunakan do-while untuk menjalankan program secara terus menerus sesuai masukan yang diberikan. Di dalam perulangan tersebut, terdapat pemilihan kondisi menggunakan switch-case untuk menjalankan operasi queue sesuai dengan masukan pengguna.

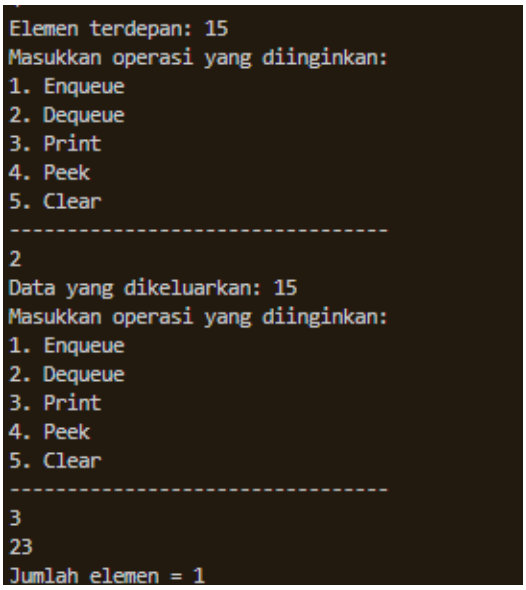
```
do {
    menu();
    pilih = sc.nextInt();
    switch (pilih) {
        case 1:
            System.out.print(s:"Masukkan data baru: ");
            int dataMasuk = sc.nextInt();
            Q.Enqueue(dataMasuk);
            break;
        case 2:
            int dataKeluar = Q.Dequeue();
            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;
    }
} while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
sc.close();
```

**Verifikasi Hasil Percobaan**

Samakan hasil compile kode program Anda dengan gambar berikut ini



**Pertanyaan**

1. Pada konstruktor, mengapa nilai awal atribut front dan rear bernilai -1, sementara atribut size bernilai 0?
   Dalam struktur data queue, nilai awal atribut front dan rear yang disetel ke -1 menunjukkan bahwa queue tersebut kosong. Ini karena dalam representasi array, posisi -1 menunjukkan bahwa tidak ada elemen yang ada.
   Sementara itu, atribut size disetel ke 0 karena pada awalnya tidak ada elemen yang disimpan di dalam queue tersebut. Dengan demikian, ketika queue dibuat, tidak ada elemen yang dianggap sebagai elemen pertama atau terakhir di dalamnya, sehingga front dan rear disetel ke -1 dan size disetel ke 0.
   Kemudian, saat elemen dimasukkan ke dalam queue, nilai-nilai front, rear, dan size akan diperbarui sesuai dengan operasi-operasi enqueue dan dequeue.
2. Pada method Enqueue, jelaskan maksud dan kegunaan dari potongan kode berikut!
   ```
   if (rear == max - 1) {
       rear = 0;
   ```
   Potongan kode tersebut memiliki tujuan dan kegunaan sebagai berikut:
   - rear == max - 1: Potongan kode ini memeriksa apakah posisi rear saat ini berada pada indeks terakhir dari array data. Jika demikian, artinya rear sudah berada di ujung maksimum array dan tidak ada lagi slot kosong di belakangnya untuk menambahkan elemen baru.
   - Jika kondisi tersebut terpenuhi, maka rear direset ke 0: Jika rear sudah berada di ujung maksimum array, maka elemen selanjutnya harus dimasukkan di awal array (indeks 0), karena queue menggunakan konsep circular array. Dengan mereset rear ke 0, kita akan kembali ke awal array dan mulai menambahkan elemen dari awal.
3. Pada method Dequeue, jelaskan maksud dan kegunaan dari potongan kode berikut!
   ```
   if (front == max - 1) {
       front = 0;
   ```
   Potongan kode tersebut memiliki maksud dan kegunaan sebagai berikut:
   - front == max - 1: Potongan kode ini memeriksa apakah posisi front saat ini berada pada indeks terakhir dari array data. Jika kondisi ini terpenuhi, artinya elemen yang akan di-dequeue adalah elemen terakhir dalam array data.
   - Jika kondisi tersebut terpenuhi, maka front direset ke 0: Dalam implementasi circular queue, setelah elemen terakhir di-dequeue, kita perlu kembali ke awal array untuk mencari elemen selanjutnya yang akan menjadi front dari queue. Dengan mereset front ke 0, kita menandakan bahwa elemen pertama dalam array akan menjadi elemen pertama dalam queue setelah proses dequeue.
4. Pada method print, mengapa pada proses perulangan variabel i tidak dimulai dari 0 (int i=0), melainkan int i=front?
   Variabel `i` dimulai dari `front` karena kita ingin mencetak elemen-elemen dalam urutan yang benar sesuai dengan struktur queue. Dengan memulai dari `front`, kita mencetak

elemen-elemen dari front queue hingga rear queue, sesuai dengan urutan mereka dalam array data circular.

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

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

Pada method print, variabel `i` tidak dimulai dari 0 (`i=0`), melainkan dari `front`, karena kita ingin mencetak elemen-elemen queue yang sesungguhnya, yang mungkin dimulai dari indeks selain 0 dalam array circular.

6. Tunjukkan potongan kode program yang merupakan queue overflow!

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

7. Pada saat terjadi queue overflow dan queue underflow, program tersebut tetap dapat berjalan dan hanya menampilkan teks informasi. Lakukan modifikasi program sehingga pada saat terjadi queue overflow dan queue underflow, program dihentikan!

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

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

Hasil Modifikasi :

```
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
--------------------------------
1
Masukkan data baru: 1
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
--------------------------------
1
Masukkan data baru: 1
Queue sudah penuh
PS D:\Matkul\SEM 2\ASD\Jobsheet9>
```

## PRAKTIKUM 2

1. Perhatikan Diagram Class berikut ini:

| Nasabah |
| --- |
| norek: String<br>nama: String<br>alamat: String<br>umur: int<br>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.

```
public class Nasabah {
    String norek;
    String nama;
    String alamat;
    int umur;
    double 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.

```
public class Queue {
    Nasabah[] data;
    int front;
    int rear;
    int size;
    int max;

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

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

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

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

```
Nasabah() {


}
```

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(x:"Queue masih kosong");
    }
}

public void print() {
    if (IsEmpty()) {
        System.out.println(x:"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);
    }
}

public boolean IsEmpty() {
    return size == 0;
}
```

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

```
package Praktikum2;
import java.util.Scanner;

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

8. Buat fungsi main, deklarasikan Scanner dengan nama sc
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.print("Masukkan kapasitas queue: ");
int jumlah = sc.ns:extInt();
Queue antri = new Queue(jumlah);
```

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

```
do {
    menu();
    pilih = sc.nextInt();
    sc.nextLine();
    switch (pilih) {
        case 1:
            System.out.print("No Rekening: ");
            String norek = scs:.nextLine();
            System.out.print("Nama: ");
            String nama = sc.s:nextLine();
            System.out.print("Alamat: ");
            String alamat = ss:c.nextLine();
            System.out.print("Umur: ");
            int umur = sc.nexs:tInt();
            System.out.print("Saldo: ");
            double saldo = scs:.nextDouble();
            Nasabah nb = new Nasabah(norek, nama, alamat, umur, saldo);
            sc.nextLine();
            antri.Enqueue(nb);
            break;
        case 2:
            Nasabah data = antri.Dequeue();
            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;
        case 3:
            antri.peek();
            break;
        case 4:
            antri.print();
            break;
    }
} while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4);
sc.close();
```

**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: 1213141516171819
Nama: Afifah
Alamat: Lowokwaru, malang
Umur: 19
Saldo: 19000000
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
-------------------------
1
No Rekening: 1314151617181912
Nama: Dewi
Alamat: Darmo, Surabaya
Umur: 19
Saldo: 2300000
```

```
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
-------------------------
4
1213141516171819 Afifah Lowokwaru, malang 19 1.9E7
1314151617181912 Dewi Darmo, Surabaya 19 2300000.0
Jumlah elemen = 2
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
-------------------------
3
Elemen terdepan: 1213141516171819 Afifah Lowokwaru, malang 19 1.9E7
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
-------------------------
2
Antrian yang keluar: 1213141516171819 Afifah Lowokwaru, malang 19 1.9E7
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
-------------------------
4
1314151617181912 Dewi Darmo, Surabaya 19 2300000.0
Jumlah elemen = 1
```

**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;
}
```

   Potongan kode tersebut digunakan untuk memastikan bahwa data yang di-dequeue dari antrian memiliki nilai yang valid sebelum dicetak. Jika semua atribut data seperti nomor rekening, nama, alamat, umur, dan saldo tidak kosong atau tidak nol, maka data tersebut dianggap valid dan dicetak sebagai "Antrian yang keluar" bersama dengan informasi terkait. Setelah itu, pernyataan `break` digunakan untuk keluar dari loop switch case.

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!

   **Menambahkan peekRear**

```
public Nasabah peekRear() {
    if (!IsEmpty()) {
        return data[rear];
    } else {
        System.out.println(x:"Queue masih kosong");
        return null;
    }
}
```

   **Menambahkan Cek antrian paling belakang**

```
public class QueueMain {
    public static void menu() {
        System.out.println(x:"Pilih menu: ");
        System.out.println(x:"1. Antrian baru");
        System.out.println(x:"2. Antrian keluar");
        System.out.println(x:"3. Cek Antrian terdepan");
        System.out.println(x:"4. Cek Semua Antrian");
        System.out.println(x:"5. Cek Antrian paling belakang");
        System.out.println(x:"--------------------------------");
    }
```

```
case 5:
    Nasabah rearData = antri.peekRear();
    if (rearData != null) {
        System.out.println("Antrian paling belakang: " + rearData.alamat + " " + rearData.umur + " " + rearData.norek + " " + rearData.saldo + " " + rearData.nama);
    }
    break;
```

**Hasil Modifikasi:**

```
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. Cek Antrian paling belakang
------------------------------
4
1213141516171819 Afifah Malang 19 1000000.0
2324252627282921 Dewi Surabaya 18 2000000.0
Jumlah elemen = 2
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. Cek Antrian paling belakang
------------------------------
5
Antrian paling belakang: Surabaya 18 2324252627282921 2000000.0 Dewi
```

## TUGAS

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:

| Pasien |
| --- |
| nama: String<br>noID: int<br>jenisKelamin: char<br>umur: int |
| Pasien (nama: String, noID: int, jenisKelamin: char, umur: int) |

| Queue |
| --- |
| antrian: Pasien[]<br>front: int<br>rear: int<br>size: int<br>max: int |

| |
|---|
| Queue(n: int)<br>isEmpty(): boolean<br>isFull(): boolean<br>enqueue(antri: Pasien): void<br>dequeue(): int<br>print(): void<br>peek(): void<br>peekRear(): void<br>peekPosition(nama: String): void<br>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

Buat Folder Tugas14 yang berisi:

Pasien.java

```
package Tugas14;

public class Pasien {
    String nama;
    int noID;
    char jenisKelamin;
    int umur;

    public Pasien(String nama, int noID, char jenisKelamin, int umur) {
        this.nama = nama;
        this.noID = noID;
        this.jenisKelamin = jenisKelamin;
        this.umur = umur;
    }
}
```

Queue.java

```java
package Tugas14;

public class Queue {
    Pasien[] antrian;
    int front;
    int rear;
    int size;
    int max;

    public Queue(int n) {
        max = n;
        antrian = new Pasien[max];
        size = 0;
        front = rear = -1;
    }

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

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

    public void enqueue(Pasien pasien) {
        if (isFull()) {
            System.out.println(x:"Queue sudah penuh");
        } else {
            if (isEmpty()) {
                front = rear = 0;
            } else {
                rear = (rear + 1) % max;
            }
            antrian[rear] = pasien;
            size++;
        }
    }
```

```java
public Pasien dequeue() {
    Pasien pasien = null;
    if (isEmpty()) {
        System.out.println(x:"Queue masih kosong");
    } else {
        pasien = antrian[front];
        size--;
        if (isEmpty()) {
            front = rear = -1;
        } else {
            front = (front + 1) % max;
        }
    }
    return pasien;
}

public void print() {
    if (isEmpty()) {
        System.out.println(x:"Queue masih kosong");
    } else {
        int i = front;
        do {
            System.out.println("Nama: " + antrian[i].nama + ", No. ID: " + antrian[i].noID + ", Jenis Kelamin: " + antrian[i].jenisKelamin + ", Umur: " + antrian[i].umur);
            i = (i + 1) % max;
        } while (i != (rear + 1) % max);
    }
}

public void peek() {
    if (!isEmpty()) {
        System.out.println("Pasien terdepan: " + antrian[front].nama);
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}
```

```java
public void peekRear() {
    if (!isEmpty()) {
        System.out.println("Pasien paling belakang: " + antrian[rear].nama);
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}

public void peekPosition(String nama) {
    if (!isEmpty()) {
        int i = front;
        int position = 1;
        do {
            if (antrian[i].nama.equals(nama)) {
                System.out.println("Posisi " + nama + " dalam antrian: " + position);
                return;
            }
            position++;
            i = (i + 1) % max;
        } while (i != (rear + 1) % max);
        System.out.println("Pasien " + nama + " tidak ditemukan dalam antrian");
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}
```

```java
public void daftarPasien() {
    if (!isEmpty()) {
        System.out.println(x:"Daftar Pasien dalam Antrian:");
        int i = front;
        do {
            System.out.println("Nama: " + antrian[i].nama + ", No. ID: " + antrian[i].noID + ", Jenis Kelamin: " + antrian[i].jenisKelamin + ", Umur: " + antrian[i].umur);
            i = (i + 1) % max;
        } while (i != (rear + 1) % max);
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}
```

QueueMain.java

```java
package Tugas14;
import java.util.Scanner;

public class QueueMain {
    public static void main(String[] args) {
        Run | Debug
        Scanner sc = new Scanner(System.in);
        System.out.print("Masukkan kapasitas queue: ");
        int jumlah = sc.ns:extInt();
        Queue antri = new Queue(jumlah);
        int pilih;
```

```java
do {
    menu();
    pilih = sc.nextInt();
    sc.nextLine();
    switch (pilih) {
        case 1:
            System.out.print("Nama: ");
            String nama = sc.s:nextLine();
            System.out.print("No. ID: ");
            int noID = sc.nexs:tInt();
            sc.nextLine();
            System.out.print("Jenis Kelamin (L/P): ");
            char jenisKelamins: = sc.nextLine().charAt(0);
            System.out.print("Umur: ");
            int umur = sc.nexs:tInt();
            Pasien pasien = new Pasien(nama, noID, jenisKelamin, umur);
            antri.enqueue(pasien);
            break;
        case 2:
            Pasien keluar = antri.dequeue();
            if (keluar != null) {
                System.out.println("Pasien yang keluar: " + keluar.nama);
            }
            break;
        case 3:
            antri.peek();
            break;
        case 4:
            antri.peekRear();
            break;
        case 5:
            System.out.print("Masukkan nama Pasien: ");
            String namaCari =s: sc.nextLine();
            antri.peekPosition(namaCari);
            break;
        case 6:
            antri.daftarPasien();
            break;
    }
} while (pilih >= 1 && pilih <= 6);
}
```

```java
public static void menu() {
    System.out.println("Pilih menu: ");
    System.out.println(x:"1. Antri Pasien");
    System.out.println(x:"2. Panggil Pasien");
    System.out.println(x:"3. Cek Pasien terdepan");
    System.out.println(x:"4. Cek Pasien paling belakang");
    System.out.println(x:"5. Cek posisi Pasien berdasarkan nama");
    System.out.println(x:"6. Daftar Pasien dalam antrian");
    System.out.println(x:"7. Keluar");
    System.out.println(x:"----------------------------------------");
}
```

## Hasil Program:

```
Masukkan kapasitas queue: 2
Pilih menu:
1. Antri Pasien
2. Panggil Pasien
3. Cek Pasien terdepan
4. Cek Pasien paling belakang
5. Cek posisi Pasien berdasarkan nama
6. Daftar Pasien dalam antrian
7. Keluar
----------------------------------------
1
Nama: Sila
No. ID: 4
Jenis Kelamin (L/P): P
Umur: 19
Pilih menu:
1. Antri Pasien
2. Panggil Pasien
3. Cek Pasien terdepan
4. Cek Pasien paling belakang
5. Cek posisi Pasien berdasarkan nama
6. Daftar Pasien dalam antrian
7. Keluar
----------------------------------------
1
Nama: Afifah
No. ID: 14
Jenis Kelamin (L/P): P
Umur: 19
```

```
Pilih menu:
1. Antri Pasien
2. Panggil Pasien
3. Cek Pasien terdepan
4. Cek Pasien paling belakang
5. Cek posisi Pasien berdasarkan nama
6. Daftar Pasien dalam antrian
7. Keluar
----------------------------------------
3
Pasien terdepan: Afifah
Pilih menu:
1. Antri Pasien
2. Panggil Pasien
3. Cek Pasien terdepan
4. Cek Pasien paling belakang
5. Cek posisi Pasien berdasarkan nama
6. Daftar Pasien dalam antrian
7. Keluar
----------------------------------------
4
Pasien paling belakang: Afifah
Pilih menu:
1. Antri Pasien
2. Panggil Pasien
3. Cek Pasien terdepan
4. Cek Pasien paling belakang
5. Cek posisi Pasien berdasarkan nama
6. Daftar Pasien dalam antrian
7. Keluar
----------------------------------------
5
Masukkan nama Pasien: Afifah
Posisi Afifah dalam antrian: 1
Pilih menu:
1. Antri Pasien
2. Panggil Pasien
3. Cek Pasien terdepan
4. Cek Pasien paling belakang
5. Cek posisi Pasien berdasarkan nama
6. Daftar Pasien dalam antrian
7. Keluar
----------------------------------------
6
Daftar Pasien dalam Antrian:
Nama: Afifah, No. ID: 14, Jenis Kelamin: P, Umur: 19
```

```
Pilih menu:
1. Antri Pasien
2. Panggil Pasien
3. Cek Pasien terdepan
4. Cek Pasien paling belakang
5. Cek posisi Pasien berdasarkan nama
6. Daftar Pasien dalam antrian
7. Keluar
----------------------------------------
7
PS D:\Matkul\SEM 2\ASD\Jobsheet9>
```