PRAKTIKUM STRUKTUR DATA

JOBSHEET 8

Oleh:

Alfina Salsabilla

2141720044

TI – 1G

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2022

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

8.2.1 Langkah-langkah Percobaan

```
package Praktikum1;

/**
 *
 * @author adesta
 */
public 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;
    }

    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]);
    }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(data[i] + " ");
            i = (i+1) % max;
        }
        System.out.println(data[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 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++;
        }
    }

    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;
    }
}
```

```java
package Praktikuml;

import java.util.Scanner;
/**
 *
 * @author adesta
 */
public class QueueMain {
    public static void menu(){
        System.out.println("Masukkan operasi yang diinginkan : ");
        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("----------------------");
    }

    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("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;
            }
        }while(pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
    }
}
```

8.2.2 Verifikasi Hasil Percobaan

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

```
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
----------------------
2
Data yang dikeluarkan : 15
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
----------------------
3
31
Jumlah elemen = 1
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
----------------------
5
Queue berhasil dikosongkan
```

8.2.3 Pertanyaan!

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

   Karena atribut front dan rear tidak menunjuk ke data manapun, sedangkan size merupakan banyaknya nilai yang diinputkan ke dalam array

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

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

   Apabila rear berada pada indeks paling belakang maka rear akan dipindahkan ke indeks paling depan

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

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

   Apabila front berada pada indeks paling belakang maka front akan dipindahkan ke indeks paling depan

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

   Karena looping tidak selalu mulai dari indeks ke-0 dan front tidak selalu berada di indeks ke-0

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

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

   Potongan kode diatas digunakan agar i tidak melebihi max

6. Tunjukkan potongan kode program yang merupakan queue overflow!

```
if(IsFull()){
    System.out.println("Queue sudah penuh");
```

7. Pada saat terjadi queue overflaw 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!

   Kode Program

```
package Praktikum1;

import java.util.Scanner;
/**
 *
 * @author adesta
 */
public class QueueMain {
    public static void menu(){
        System.out.println("Masukkan operasi yang diinginkan : ");
        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");
        //modifikasi
        System.out.println("6. Keluar program");
        System.out.println("----------------------");
    }

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

```
            do{
                menu();
                //pilih = sc.nextInt();
                int 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;
                    //modifikasi
                    case 6:
                        System.exit(0);
                    default:
                    System.out.println("Maaf, anda salah memasukkan menu pilihan");
                }
                System.out.println("Apakah ingin kembali ke menu utama ? [Y/T]");
                ulang = sc.next();
            }//while(pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
            while(ulang.equalsIgnoreCase("Y"));
        }
    }
```

Output

```
run:
Masukkan kapasitas queue : 4
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Keluar program
----------------------
1
Masukkan data baru : 15
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Keluar program
----------------------
1
Masukkan data baru : 31
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Keluar program
----------------------
4
Elemen terdepan : 15
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Keluar program
----------------------
6
BUILD SUCCESSFUL (total time: 49 seconds)
```

8.3 Praktikum 2

8.3.1 Langkah-langkah Percobaan

```java
package Praktikum2;

/**
 *
 * @author adesta
 */
public class Nasabah {
    String norek;
    String nama;
    String alamat;
    int umur;
    double saldo;

    Nasabah(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;
    }

    Nasabah[] data;
    int front;
    int rear;
    int size;
    int max;

    Nasabah(){

    }
```

```
    public Nasabah(int n){
        max = n;
        data = new Nasabah[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].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.print(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 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(Nasabah 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++;
        }
    }
```

```
    public Nasabah Dequeue(){
        Nasabah dt = new Nasabah();
        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;
    }
}
```

```java
package Praktikum2;

import java.util.Scanner;
/**
 *
 * @author adesta
 */
public class NasabahMain {
    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.print("Masukkan kapasitas queue : ");
        int jumlah = sc.nextInt();
        Nasabah antri = new Nasabah(jumlah);
        int pilih;
        do{
            menu();
            pilih = sc.nextInt();
            sc.nextLine();

            switch(pilih){
                case 1:
                    System.out.print("No Rekening : ");
                    String norek = sc.nextLine();
                    System.out.print("Nama : ");
                    String nama= sc.nextLine();
                    System.out.print("Alamat : ");
                    String alamat = sc.nextLine();
                    System.out.print("Umur : ");
                    int umur = sc.nextInt();
                    System.out.print("Saldo : ");
                    double saldo = sc.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);
    }
}
```

8.3.2 Verifikasi Hasil Percobaan

```
run:
Masukkan kapasitas queue : 8
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
----------------------
1
No Rekening : 12345
Nama : Dewi
Alamat : Malang
Umur : 23
Saldo : 1300000
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
----------------------
1
No Rekening : 32940
Nama : Susan
Alamat : Surabaya
Umur : 39
Saldo : 42000000
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
----------------------
4
12345 Dewi Malang 23 1300000.032940 Susan Surabaya 39 4.2E7
Jumlah elemen = 2
```

8.3.3 Pertanyaan!

1. Pada class NasabahMain, 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;
}
```

Kode program diatas digunakan untuk mengecek apakah norek, nama, alamat, umur, saldo kosong atau tidak, apabila salah satu dari kelima atribut tersebut ada yang kosong maka tidak akan menampilkan antrian yang keluar

2. Lakukan modifikasi program dengan menambahkan method baru bernama **peekRear** pada class Nasabah yang digunakan untuk mengecek antrian yang berada di posisi belakang! Tambahkan pula daftar menu **5. Cek Antrian paling belakang** pada class **NasabahMain** sehingga method **peekRear** dapat dipanggil!

   Kode Program

```
    public void peekRear(){
        double min = data[0].rear;
        int idx = 0;
        for (int i=0; i<= front; i++){
            if(min > data[i].rear){
                min = data[i].rear;
                idx = i;
            }
        }
        System.out.println("Antrian paling belakang " + data[idx].norek +
                " " + data[idx].nama + " " + data[idx].alamat + " " + data[idx].umur +
                " " + data[idx].saldo);
    }
}
```

```java
package Praktikum2;

import java.util.Scanner;
/**
 *
 * @author adesta
 */
public class NasabahMain {
    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");
        //modifikasi
        System.out.println("5. cek Antrian paling belakang");
        System.out.println("6. Keluar dari program");
        System.out.println("----------------------");
    }

    public static void main(String[] args){
        Scanner sc =new Scanner(System.in);
        System.out.print("Masukkan kapasitas queue : ");
        int jumlah = sc.nextInt();
        Nasabah antri = new Nasabah(jumlah);
        //int pilih;
        String ulang;
        do{
            menu();
            //pilih = sc.nextInt();
```

```
            int pilih = sc.nextInt();
            sc.nextLine();
            switch(pilih){
                case 1:
                    System.out.print("No Rekening : ");
                    String norek = sc.nextLine();
                    System.out.print("Nama : ");
                    String nama= sc.nextLine();
                    System.out.print("Alamat : ");
                    String alamat = sc.nextLine();
                    System.out.print("Umur : ");
                    int umur = sc.nextInt();
                    System.out.print("Saldo : ");
                    double saldo = sc.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;
                case 5:
                    antri.peekRear();
                    break;
                case 6:
                    System.exit(0);
                    break;
                default:
                    System.out.println("Maaf, anda salah memasukkan menu pilihan");
            }
            System.out.println("Apakah ingin kembali ke menu utama ? [Y/T]");
            ulang = sc.next();
        }//while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4);
        while(ulang.equalsIgnoreCase("Y"));
    }
}
```

Output

```
run:
Masukkan kapasitas queue : 8
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. cek Antrian paling belakang
6. Keluar dari program
----------------------
1
No Rekening : 12345
Nama : Dewi
Alamat : Malang
Umur : 23
Saldo : 1300000
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. cek Antrian paling belakang
6. Keluar dari program
----------------------
1

No Rekening : 32940
Nama : Susan
Alamat : Surabaya
Umur : 39
Saldo : 42000000
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. cek Antrian paling belakang
6. Keluar dari program
----------------------
4
12345 Dewi Malang 23 1300000.032940 Susan Surabaya 39 4.2E7
Jumlah elemen = 2
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. cek Antrian paling belakang
6. Keluar dari program
----------------------
2
```

```
Antrian yang keluar : 12345 Dewi Malang 231300000.0
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. cek Antrian paling belakang
6. Keluar dari program
----------------------
3
Elemen terdepan : 32940 Susan Surabaya 39 4.2E7
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. cek Antrian paling belakang
6. Keluar dari program
----------------------
5
Antrian paling belakang 12345 Dewi Malang 23 1300000.0
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
5. cek Antrian paling belakang
6. Keluar dari program
----------------------
6
BUILD SUCCESSFUL (total time: 1 minute 46 seconds)
```

8.4 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 ke berapa

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!

Kode Program

```java
package Praktikum1;

/**
 *
 * @author adesta
 */
public 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;
    }

    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]);
        }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(data[i] + " ");
                i = (i+1) % max;
            }
            System.out.println(data[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 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++;
    }
}

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;
}
```

```
    //Tugas1
    public void peekPosition(int m){
        if(IsEmpty()){
            System.out.println("Queue masih kosong");
        }else{
            int counter = 0;
            int i = front;
            while(i != rear){
                if(data[i] == m){
                    System.out.println("Nilai " + data[i] + " terletak pada indeks ke-" + i);
                    counter++;
                }
                i=(i+1)%max;
            }
            if(counter !=1){
                System.out.println("Nilai " + data[i] + " terletak pada indeks ke-" + i);
            }
        }
    }
```

```
    public void peekAt(int pos){
        if(IsEmpty()){
            System.out.println("Queue masih kosong");
        }else{
            int counter = 0;
            int i = front;
            while(i != rear){
                if(i==pos){
                    System.out.println("Pada indeks ke-" + i + " terdapat nilai " + data[i]);
                    counter++;
                }
                i=(i+1)%max;
            }
            if(counter !=1){
                System.out.println("Pada indeks ke-" + i + " terdapat nilai " + data[i]);
            }
        }
    }
}
```

```java
package Praktikuml;

import java.util.Scanner;
/**
 *
 * @author adesta
 */
public class QueueMain {
    public static void menu(){
        System.out.println("Masukkan operasi yang diinginkan : ");
        System.out.println("1. Enqueue");
        System.out.println("2. Dequeue");
        System.out.println("3. Print");
        System.out.println("4. Peek");
        //Tugas1
        System.out.println("5. Cari posisi indeks");
        System.out.println("6. Tampilkan posisi indeks");
        System.out.println("7. Clear");
        //modifikasi
        System.out.println("8. Keluar program");
        System.out.println("----------------------");
    }

    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("Masukkan kapasitas queue : ");
        int n = sc.nextInt();
        Queue Q = new Queue(n);
        //int pilih;
        String ulang;
        do{
            menu();
            //pilih = sc.nextInt();
            int 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;
```

```
                //Tugas1
                case 5:
                    System.out.print("Masukkan data yang dicari : ");
                    int dataCari = sc.nextInt();
                    Q.peekPosition(dataCari);
                    break;
                case 6:
                    System.out.print("Masukkan data yang ditampilkan : ");
                    int dataTampil = sc.nextInt();
                    Q.peekPosition(dataTampil);
                    break;
                case 7:
                    Q.clear();
                    break;
                //modifikasi
                case 8:
                    System.exit(0);
                default:
                    System.out.println("Maaf, anda salah memasukkan menu pilihan");
            }
            System.out.println("Apakah ingin kembali ke menu utama ? [Y/T]");
            ulang = sc.next();
        }//while(pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
        while(ulang.equalsIgnoreCase("Y"));
    }
}
```

Output

```
run:
Masukkan kapasitas queue : 8
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Cari posisi indeks
6. Tampilkan posisi indeks
7. Clear
8. Keluar program
----------------------
1
Masukkan data baru : 31
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Cari posisi indeks
6. Tampilkan posisi indeks
7. Clear
8. Keluar program
----------------------
1
Masukkan data baru : 15
Apakah ingin kembali ke menu utama ? [Y/T]
Y
```

```
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Cari posisi indeks
6. Tampilkan posisi indeks
7. Clear
8. Keluar program
---------------------
4
Elemen terdepan : 31
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Cari posisi indeks
6. Tampilkan posisi indeks
7. Clear
8. Keluar program
---------------------
5
Masukkan data yang dicari : 31
Nilai 31terletak pada indeks ke-0
Apakah ingin kembali ke menu utama ? [Y/T]
Y
```

```
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Cari posisi indeks
6. Tampilkan posisi indeks
7. Clear
8. Keluar program
---------------------
6
Masukkan data yang ditampilkan : 31
Nilai 31terletak pada indeks ke-0
Apakah ingin kembali ke menu utama ? [Y/T]
T
BUILD SUCCESSFUL (total time: 1 minute 26 seconds)
```

2. Buatlah program antrian untuk mengilustrasikan 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
antiran: mahasiswa[] front: int rear: int size: int max: int
Queue(n: int) isEmpty(): boolean isFull(): boolean enqueue(antri: Mahasiswa): void dequeue(): int print(): void peek(): void peekRear(): void peekPosition(nim: String): void printMahasiswa(posisi: int): void

Keterangan :

- Method create(), isEmpty(), isFull(), enqueue(), dequeue() dan print(), kegunaannya sama seperti yang telah dibuat pada praktikum
- Method peek(): digunakan untuk menampilkan data Mahasiswa yang berasa di posisi antrian paling depan
- Method peekRear(): digunakan untuk menampilkan posisi antrian ke berapa, seorang Mahasiswa berada. Pengecekan dilakukan berdasarkan NIM
- Method printMahasiswa(): digunakan untuk menampikan data mahasiswa pada suatu posisi tertentu dalam antrian

Kode Program

```java
package Tugas2;

/**
 *
 * @author adesta
 */
public class Mahasiswa {
    String nim;
    String 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;
    }

    Mahasiswa[] data;
    int front;
    int rear;
    int size;
    int max;

    Mahasiswa(){

    }

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

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

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

    public void peek(){
        if(!IsEmpty()){
            System.out.println("Antrian terdepan : " + data[front].nim + " " +
                    data[front].nama + " " + data[front].absen + " " + data[front].ipk);
        }else{
            System.out.println("Antrian masih kosong");
        }
    }
```

```
public void peekRear(){
    if (!IsEmpty()) {
        System.out.println("Antrian paling belakang : " + data[rear].nim + " " + data[rear].nama + " "
        + data[rear].absen + " " + data[rear].ipk );
    }else{
        System.out.println("Antrian masih kosong");
    }

}

public void peekPosition(String nim){
    if(IsEmpty()){
        System.out.println("Antrian kosong");
    }else{
        for(int i=0; i<data.length; i++){
            if(nim.equalsIgnoreCase(data[i].nim)){
                System.out.println(data[i].nim + " " + data[i].nama + " " +
                    data[i].absen + " " + data[i].ipk);
                break;
            }
        }
    }
}
```

```
    public void printMahasiswa(int m){
        if(IsEmpty()){
            System.out.println("Antrian masih kosong");
        }else{
            int counter = 0;
            int i = front;
            while(i != rear){
                if(i==m){
                    System.out.println("Antrian indeks ke-" + i + " yaitu " + data[i].nim
                    + " " + data[i].nama + " " + data[i].absen + " " + data[i].ipk);
                    counter++;
                }
                i=(i+1)%max;
            }
            if(counter !=1){
                System.out.println("Antrian indeks ke-" + i + " yaitu " + data[i].nim
                    + " " + data[i].nama + " " + data[i].absen + " " + data[i].ipk);
            }
        }
    }
}
```

```
package Tugas2;

import java.util.Scanner;
/**
 *
 * @author adesta
 */
public class MahasiswaMain {
    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 Antrian paling belakang");
        System.out.println("5. Cek Semua antrian");
        System.out.println("6. Cek posisi antrian berdasarkan NIM");
        System.out.println("7. Tampilkan data antrian Mahasiswa");
        System.out.println("8. Keluar dari program");
        System.out.println("----------------------");
    }
```

```
public static void main(String[] args){
    Scanner sc =new Scanner(System.in);
    System.out.print("Masukkan Jumlah Mahasiswa : ");
    int jumlah = sc.nextInt();
    Mahasiswa antri = new Mahasiswa(jumlah);
    String ulang;
    do{
        menu();
        int pilih = sc.nextInt();
        sc.nextLine();
        switch(pilih){
            case 1:
                System.out.print("NIM Mahasiswa : ");
                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 ms = new Mahasiswa(nim, nama, absen, ipk);
                sc.nextLine();
                antri.Enqueue(ms);
                break;
```

```
            case 2:
                Mahasiswa data = antri.Dequeue();
                if(!"".equals(data.nim) && !"".equals(data.nama) && !"".equals(data.absen)
                        && data.ipk != 0){
                    System.out.println("Antrian yang keluar : " + data.nim + " " +
                            data.nama + " " + data.absen + " " + data.ipk);
                    break;
                }
            case 3:
                antri.peek();
                break;
            case 4:
                antri.peekRear();
                break;
            case 5:
                antri.print();
                break;
            case 6:
                System.out.print("Masukkan NIM Mahasiswa yang dicari : ");
                String a = sc.nextLine();
                antri.peekPosition(a);
                break;
            case 7:
                System.out.print("Masukkan data Mahasiswa yang ingin ditampilkan : ");
                int b = sc.nextInt();
                antri.printMahasiswa(b);
                break;
            case 8:
                System.exit(0);
                break;
```

```
                default:
                    System.out.println("Maaf, anda salah memasukkan menu pilihan");
                }
                System.out.println("Apakah ingin kembali ke menu utama ? [Y/T]");
                ulang = sc.next();
            }while(ulang.equalsIgnoreCase("Y"));
        }
    }
```

Output

```
run:
Masukkan Jumlah Mahasiswa : 5
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Antrian paling belakang
5. Cek Semua antrian
6. Cek posisi antrian berdasarkan NIM
7. Tampilkan data antrian Mahasiswa
8. Keluar dari program
----------------------
1
NIM Mahasiswa : 123
Nama : Adesta
Absen : 2
IPK : 3,95
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Antrian paling belakang
5. Cek Semua antrian
6. Cek posisi antrian berdasarkan NIM
7. Tampilkan data antrian Mahasiswa
8. Keluar dari program
----------------------
1
```

```
NIM Mahasiswa : 1234
Nama : Mirabell
Absen : 14
IPK : 3,9
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Antrian paling belakang
5. Cek Semua antrian
6. Cek posisi antrian berdasarkan NIM
7. Tampilkan data antrian Mahasiswa
8. Keluar dari program
----------------------
3
Antrian terdepan : 123 Adesta 2 3.95
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Antrian paling belakang
5. Cek Semua antrian
6. Cek posisi antrian berdasarkan NIM
7. Tampilkan data antrian Mahasiswa
8. Keluar dari program
----------------------
4
```

```
-
Antrian paling belakang : 1234 Mirabell 14 3.9
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Antrian paling belakang
5. Cek Semua antrian
6. Cek posisi antrian berdasarkan NIM
7. Tampilkan data antrian Mahasiswa
8. Keluar dari program
----------------------
5
123 Adesta 2 3.951234 Mirabell 14 3.9
Jumlah Mahasiswa = 2
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Antrian paling belakang
5. Cek Semua antrian
6. Cek posisi antrian berdasarkan NIM
7. Tampilkan data antrian Mahasiswa
8. Keluar dari program
----------------------
6
```

```
Masukkan NIM Mahasiswa yang dicari : 123
123 Adesta 2 3.95
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Antrian paling belakang
5. Cek Semua antrian
6. Cek posisi antrian berdasarkan NIM
7. Tampilkan data antrian Mahasiswa
8. Keluar dari program
----------------------
7
Masukkan data Mahasiswa yang ingin ditampilkan : 1234
Antrian indeks ke-1 yaitu 1234 Mirabell 14 3.9
Apakah ingin kembali ke menu utama ? [Y/T]
Y
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Antrian paling belakang
5. Cek Semua antrian
6. Cek posisi antrian berdasarkan NIM
7. Tampilkan data antrian Mahasiswa
8. Keluar dari program
----------------------
8
BUILD SUCCESSFUL (total time: 1 minute 22 seconds)
```