LAPORAN PRAKTIKUM ALGORITMA & STRUKTUR DATA

Pertemuan – 10: Queue

Aqil Rahmat Alifiandi

2341760099

D-1V SISTEM INFORMASI BISNIS

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2023/2024

10.2 Praktikum 1

Kode program Queue05.java:

```java
public class Queue05 {

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

    public Queue05(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(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.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(x:"Queue berhasil dikosongkan");
        } else {
            System.out.println(x:"Queue masih kosong");
        }
    }

    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++;
        }
    }

    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; // Return the dequeued element
    }

}
```

Kode program QueueMain05.java:

```java
import java.util.Scanner;

/**
 * QueueMain05
 */
public class QueueMain05 {

        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:"-------------------");

    }

    Run | Debug
    public static void main(String[] args) {
        Scanner sc05 = new Scanner(System.in);
        System.out.print(s:"Masukkan kapasitas queue: ");
        int n = sc05.nextInt();
        Queue05 Q = new Queue05(n);
        int pilih;
        do {
            menu();
            pilih = sc05.nextInt();
            switch (pilih) {
                case 1:
                    System.out.print(s:"Masukkan data baru: ");
                    int dataMasuk = sc05.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);
    }
}
```

Hasil run program:

```
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
-----------------
```

10.2.3 Pertanyaan

1. Pada konstruktor, mengapa nilai awal atribut front dan rear bernilai -1, sementara atribut size bernilai 0?
   Jawaban: Front dan rear bernilai -1 karena tidak menunjuk ke data manapun sedangkan size bernilai 0 karena array masih kosong

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

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

   Jawaban: Untuk memastikan bahwa jika kita telah mencapai kapasitas maksimum antrian dan ingin menambahkan elemen baru, kita akan kembali ke awal antrian untuk menambahkan elemen tersebut.

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

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

   Jawaban: Untuk memastikan bahwa jika kita telah mencapai kapasitas maksimum antrian dan ingin menghapus elemen dari antrian, kita akan kembali ke awal antrian untuk mengambil elemen tersebut.

4. Pada method print, mengapa pada proses perulangan variabel i tidak dimulai dari 0 (int i=0), melainkan int i=front?
   Jawaban: Karena untuk mencetak elemen-elemen antrian dengan mempertahankan urutan yang benar, dimulai dari elemen yang paling depan.

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

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

   Jawaban: Digunakan untuk memperbarui nilai i ke elemen berikutnya dalam antrian dengan memperhatikan kemungkinan adanya siklik atau lingkaran, serta memastikan bahwa nilai i tetap dalam kisaran yang valid sesuai dengan kapasitas antrian.

6. Tunjukkan potongan kode program yang merupakan queue overflow!
   Jawab:

```
63      public void Enqueue(int dt) {
64          if(IsFull()) {
65              System.out.println(x:"Queue sudah penuh");
66          } else {
67              if (IsEmpty()) {
68                  front = rear = 0;
69              } else {
70                  if (rear == max - 1) {
71                      rear = 0;
72                  } else {
73                      rear++;
74                  }
75              }
76              data[rear] = dt;
77              size++;
78          }
79      }
```

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!

Jawaban:

Kode program:

```
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++;
    }
}
```

```
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; // Return the dequeued element
}
```

Hasil run kode program:

```
Masukkan kapasitas queue: 3
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----------------
1
Masukkan data baru: 34
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----------------
1
Masukkan data baru: 21
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----------------
1
Masukkan data baru: 32
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----------------
1
Masukkan data baru: 23
Queue sudah penuh
```

```
Masukkan kapasitas queue: 2
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----------------
1
Masukkan data baru: 12
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----------------
1
Masukkan data baru: 21
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----------------
2
Data yang dikeluarkan: 12
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----------------
2
Data yang dikeluarkan: 21
Masukkan operasi yang diinginkan
```

```
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
------------------
2
Queue masih kosong
aqilrahmat@Aqils-MacBook-Air Jobsheet 10 %
```

10.3 Praktikum 2

Kode program Nasabah05:

```java
public class Nasabah05 {
    String norek, nama, alamat;
    int umur;
    double saldo;

    Nasabah05() {}

    Nasabah05(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;
    }
}
```

Kode program QueueNasaba05:

```
public class QueueNasabah05 {
        Nasabah05[] data;
        int front;
        int rear;
        int size;
        int max;

        public QueueNasabah05(int n) {
            max = n;
            data = new Nasabah05[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(x:"Queue masih kosong");
            }
        }

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

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

        public void Enqueue(Nasabah05 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 Nasabah05 Dequeue() {
            Nasabah05 dt = new Nasabah05();
            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;
        }
}
```

Kode program QueueNasabahMain05:

```java
import java.util.Scanner;

public class QueueNasabahMain05 {
    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 keluar");
        System.out.println("4. Cek semua antrian");
        System.out.println("5. Cek antrian paling belakang");
        System.out.println("-------------------------");
    }

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

        System.out.print("Masukkan kapasitas queue: ");
        int jumlah = sc.nextInt();
        QueueNasabah05 antri = new QueueNasabah05(jumlah);

        int pilih;
        do {
            menu();
            pilih = sc.nextInt();

            switch (pilih) {
                case 1:
                    System.out.print("No. Rekening: ");
                    String norek = sc.nextLine();
                    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();

                    Nasabah05 nb = new Nasabah05(norek, nama, alamat, umur, saldo);
                    sc.nextLine();
                    antri.Enqueue(nb);
                    break;
                case
                    Nasabah05
                    Nasabah05 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;
            }
        } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4);
    }
}
```

10.3.3 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;
}
```

Jawaban: Untuk mencocokkan jika seluruh data tidak kosong.

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!

Jawab:

Kode program:

```
39      public void peekRear() {
40          if (!IsEmpty()) {
41              System.out.println("Elemen terbelakang: " + data[rear].norek + " " + data[rear].nama + " " + data[rear].alamat + " " + data[rear].um
42          } else {
43              System.out.println(x:"Queue masih kosong");
44          }
45      }
```

```
56                  case 5:
57                      antri.peekRear();
58                      break;
59              }
```

Hasil run kode program:

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

10.4 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

Jawaban:

Pasien05:

```
public class Pasien05 {
    String nama;
    int noID, umur;
    char jenisKelamin;


    public Pasien05(String nama, int noID, char jenisKelamin, int umur) {
        this.nama = nama;
        this.noID = noID;
        this.jenisKelamin = jenisKelamin;
        this.umur = umur;
    }
}
```

QueuePasien05:a

```java
public class QueuePasien05 {
    Pasien05[] antrian;
    int front, rear, size, max;

    public QueuePasien05(int n) {
        max = n;
        antrian = new Pasien05[max];
        size = 0;
        front = rear = -1;
    }

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

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

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

    public void enqueue(Pasien05 p) {
        if (isFull()) {
            System.out.println(x:"Antrian sudah penuh");
        } else {
            if (isEmpty()) {
                front = rear = 0;
            } else {
                rear = (rear + 1) % max;
            }
            antrian[rear] = p;
            size++;
            System.out.println("Pasien " + p.nama + " berhasil ditambahkan ke dalam antrian");
        }
    }

    public Pasien05 dequeue() {
        Pasien05 p = null;
        if (isEmpty()) {
            System.out.println(x:"Antrian masih kosong");
        } else {
            p = antrian[front];
            if (front == rear) {
                front = rear = -1;
            } else {
                front = (front + 1) % max;
            }
            size--;
        }
        return p;
    }

    public void print() {
        if (isEmpty()) {
            System.out.println(x:"Antrian masih kosong");
        } else {
            int i = front;
            while (i != rear) {
                System.out.println(antrian[i].nama + " " + antrian[i].noID + " " + antrian[i].jenisKelamin + " " + antrian[i].umur);
                i = (i + 1) % max;
            }
            System.out.println(antrian[i].nama + " " + antrian[i].noID + " " + antrian[i].jenisKelamin + " " + antrian[i].umur);
        }
    }

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

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

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

    public void daftarPasien() {
        if (!isEmpty()) {
            System.out.println(x:"Daftar Pasien:");
            int i = front;
            while (i != rear) {
                System.out.println(antrian[i].nama + " " + antrian[i].noID + " " + antrian[i].jenisKelamin + " " + antrian[i].umur);
                i = (i + 1) % max;
            }
            System.out.println(antrian[i].nama + " " + antrian[i].noID + " " + antrian[i].jenisKelamin + " " + antrian[i].umur);
        } else {
            System.out.println(x:"Antrian masih kosong");
        }
    }
}
```

QueuePasienMain05:

```java
import java.util.Scanner;

public class QueuePasienMain05 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print(s:"Masukkan kapasitas antrian: ");
        int capacity = sc.nextInt();
        QueuePasien05 antrian = new QueuePasien05(capacity);

        int pilihan;
        do {
            System.out.println(x:"============Pilihan Menu:==============");
            System.out.println(x:"1. Tambah Pasien");
            System.out.println(x:"2. Hapus Pasien Terdepan");
            System.out.println(x:"3. Lihat Pasien Terdepan");
            System.out.println(x:"4. Lihat Pasien Terbelakang");
            System.out.println(x:"5. Cari Posisi Pasien");
            System.out.println(x:"6. Daftar Pasien");
            System.out.println(x:"7. Keluar");
            System.out.println(x:"=======================================");
            System.out.print(s:"Pilihan: ");
            pilihan = sc.nextInt();

            switch (pilihan) {
                case 1:
                    sc.nextLine();
                    System.out.print(s:"Nama: ");
                    String nama = sc.nextLine();
                    System.out.print(s:"Nomor Identitas: ");
                    int noID = sc.nextInt();
                    System.out.print(s:"Jenis Kelamin (L/P): ");
                    char jenisKelamin = sc.next().charAt(index:0);
                    System.out.print(s:"Umur: ");
                    int umur = sc.nextInt();

                    Pasien05 pb = new Pasien05(nama, noID, jenisKelamin, umur);
                    sc.nextLine();
                    antrian.enqueue(pb);
                    break;
                case 2:
                    Pasien05 pasienHapus = antrian.dequeue();
                    if (pasienHapus != null) {
                        System.out.println("Pasien " + pasienHapus.nama + " berhasil dihapus dari antrian");
                    }
                    break;
                case 3:
                    antrian.peek();
                    break;
                case 4:
                    antrian.peekRear();
                    break;
                case 5:
                    sc.nextLine();
                    System.out.print(s:"Nama Pasien: ");
                    String namaCari = sc.nextLine();
                    antrian.peekPosition(namaCari);
                    break;
                case 6:
                    antrian.daftarPasien();
                    break;
                case 7:
                    System.out.println(x:"Terima kasih, program selesai.");
                    break;
                default:
                    System.out.println(x:"Pilihan tidak valid");
            }
        } while (pilihan != 7);

        sc.close();
    }
}
```