Nama : Abdullah Azzam
Kelas : 1F
Absen : 01

8.2.3 Pertanyaan

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

Jawab :

- atribut front dan rear bernilai -1,dimaksutkan untuk menunjukkan bahwa sizenya masih dalam konsisi kosong/0,
- tribut size bernilai 0, Karena setiap array di mulai dari indexs ke 0,

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

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

Jawab :

potongan kode tersebut berguna untuk jika rear/data berada pada posisi max-1/indexs terakhir dari array, maka disaat ada penambahan data baru, maka akan di tempatkan pada index ke -0.

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

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

Jawab :

potongan kode tersebut berguna untuk jika front berada pada posisi max-1 atau indexs terakhir dari array, maka disaat ada penambahan data baru, maka akan di tempatkan pada index ke -0.

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

Jawab :

Karena posisi front atau data terdepan tidak selalu pada indeks ke-0, sedangkan perulangan dimulai dengan posisi frontnya

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

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

Jawab :

maksud dari kode tersebut adalah, nilai i(front) jika tidak berposisi sebagai rear, maka akan dilakukan penambahan pada variable tersebut lalu akan dimodulus dengan nilai max atau kapasitas dari Queue tersebut. Gunanya untuk mencegah melakukan print melebihi max, sehingga di lakukan modulo max

6. Tunjukkan potongan kode program yang merupakan queue overflow!

Jawab :

Overflow :

```
public void enqueue(int dt){
    if(IsFull()){
        System.out.println("Queue sudah penuh");
```

Underflow :

```
public int dequeue(){
    int dt = 0;
    if(IsEmpty()){
        System.out.println("Queue masih kosong");
```

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!

Jawab :

```
public class queue {
    int max, size, front, rear, data[];

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

8.3.3 Pertanyaan

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

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

Jawab :

equals adalah membandingkan dua string jika kedua string sama maka akan mereturn true dan sebaliknya

!"".equals(data.norek) ... digunakan untuk mengecek apakah norek pada data tidak sama dengan string kosong ("") dst

jika semua kondisi bernilai true maka akan menampilkan pada konsol isi dari data tersebut lalu berhenti (break)

break berfungsi untuk mengakhiri kode program akgar tidak terus melakukan proses

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 :

```java
public class queue {
    int max, size, front, rear;
    nasabah[] data;

    public queue(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
        }else{
            System.out.println("Queue masih kosong");
        }
    }
    public void peekRear(){
        if(!IsEmpty()){
            System.out.println("Elemen terdepan : "+data[rear].norek+ " "+data[rear].nama+" "+data[rear].alamat+" "+data[rear].um
        }else{
            System.out.println("Queue masih kosong");
        }
    }
    public void print(){
        if(IsEmpty()){
            System.out.println("Queue masih kosong");
        }else {
            int i = front;
            while(i != rear){
                System.out.println(data[i].norek + " "+data[i].nama + " "+data[i].alamat + " "+data[i].umur + " "+data[i].saldo);
                i = (i + 1) % max;
            }
            System.out.println(data[i].norek + " "+data[i].nama + " "+data[i].alamat + " "+data[i].umur + " "+data[i].saldo);
            System.out.println("Jumlah elemen = "+ size);
        }
    }
    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
public class queueMain {
/**
* @param args the command line arguments
*/
public static void menu(){
    System.out.println("\nPilih Menu ");
    System.out.println(" 1. Antrian baru\n 2. Antrian Keluar\n 3. Cek Antrian Terdepan\n 4. Cek Semua Antrian\n 5. Cek Antrian Pa
    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();
    queue antri = new queue(jumlah);

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

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