

JOBSHEET X QUEUE

Satria Wiguna / TI_1D/26

10.1 Tujuan Praktikum

Setelah melakukan materi praktikum ini, mahasiswa mampu:

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

10.2 Praktikum 1

Waktu percobaan : 45 menit

Pada percobaan ini, kita akan mengimplementasikan penggunaan class Queue.

10.2.1 Langkah-langkah Percobaan

1. Perhatikan Diagram Class Queue berikut ini:

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

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

2. Buat package dengan nama **Praktikum1**, kemudian buat class baru dengan nama **Queue**.
3. Tambahkan atribut-atribut Queue sesuai diagram class, kemudian tambahkan pula konstruktornya seperti gambar berikut ini.



```
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() {
    if (size == 0) {
        return true;
    } else {
        return false;
    }
}
```

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

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

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

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

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

```
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);
    }
}
```

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("Queue berhasil dikosongkan");
    } else {
        System.out.println("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("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 belakang

```

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;
}

```

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

```

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("-----");
}

```

12. Buat fungsi **main**, kemudian deklarasikan Scanner dengan nama **sc**.
13. Buat variabel **n** untuk menampung masukan berupa jumlah maksimal elemen yang dapat disimpan pada queue.

```

System.out.print("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);

```

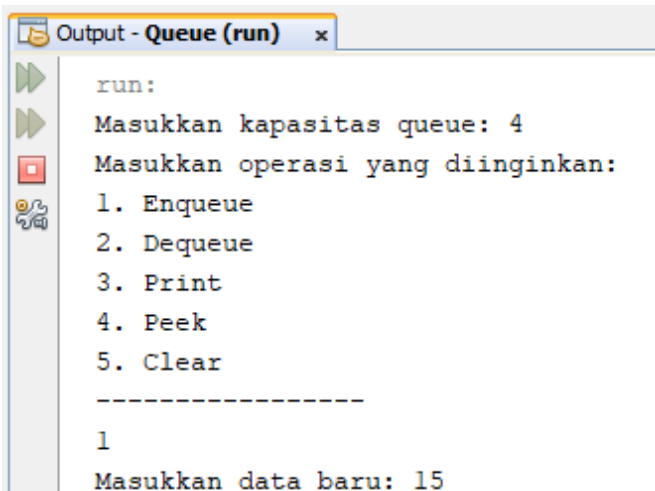
15. Deklarasikan variabel dengan nama **pilih** bertipe integer untuk menampung pilih menu dari pengguna.
16. Lakukan perulangan menggunakan do-while untuk menjalankan program secara terus menerus sesuai masukan yang diberikan. Di dalam perulangan tersebut, terdapat pemilihan kondisi menggunakan switch-case untuk menjalankan operasi queue sesuai dengan masukan pengguna.

```
do {
    menu();
    pilih = sc.nextInt();
    switch (pilih) {
        case 1:
            System.out.print("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);
```

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

10.2.2 Verifikasi Hasil Percobaan

Samakan hasil compile kode program Anda dengan gambar berikut ini.



```
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



Queue :

```

1  package Praktikum1;
2
3  public class Queue {
4      int[] data;
5      int front;
6      int rear;
7      int size;
8      int max;
9
10     public Queue(int n){
11         max = n;
12         data = new int [max];
13         size = 0;
14         front = rear = -1;
15     }
16
17     boolean isEmpty (){
18         if (size == 0){
19             return true;
20         } else {
21             return false;
22         }
23     }
24
25     public boolean isFull () {
26         if (size == max) {
27             return true;
28         } else {
29             return false;
30         }
31     }
32
33     public void peek() {
34         if (!isEmpty()){
35             System.out.println("Elemen Terdepan " + data [front]);
36         } else {
37             System.out.println("Queue masih kosong");

```



```

38     }
39 }
40
41 public void print() {
42     if (isEmpty()){
43         System.out.println(x:"Queue masih kosong");
44     } else {
45         int i = front;
46         while (i != rear ){
47             System.out.println(data[i] + " ");
48             i= (i+1) % max;
49         }
50         System.out.println(data[i] + " ");
51         System.out.println("Jumlah elemen = " + size);
52     }
53 }
54
55 public void clear() {
56     if (!isEmpty()) {
57         front = rear = -1;
58         size = 0;
59         System.out.println(x:"Queue berhasil dikosongkan");
60     } else {
61         System.out.println(x:"Queue masih kosong");
62     }
63 }
64
65 public void Enqueue (int dt) {
66     if (isFull()) {
67         System.out.println(x:"Queue sudah penuh");
68     } else {
69         if (isEmpty()){
70             front = rear = 0;
71         } else {
72             if (rear == max -1){
73                 rear =0;
74             } else {
75                 rear ++;
76             }
77         }
78         data [rear] = dt;
79         size ++ ;
80     }
81 }
82
83 int Dequeue (){
84     int dt = 0;
85     if (isEmpty()) {
86         System.exit(dt);
87     }else{
88         dt = data[front];
89         size --;
90         if (isEmpty()) {
91             front = rear = -1;
92         }else{
93             if (front == max -1) {
94                 front = 0;
95             }else{
96                 front ++;
97             }
98         }
99     }
100     return dt;
101 }
102 }

```




QueueMain :

```

1 package Praktikum1;
2 import java.util.Scanner;
3
4 public class QueueMain {
5     static void menu() {
6         System.out.println(x:"Masukkan operasi yang diinginkan");
7         System.out.println(x:"1. Enqueue");
8         System.out.println(x:"2. Dequeue");
9         System.out.println(x:"3. Print");
10        System.out.println(x:"4. Peek");
11        System.out.println(x:"5. Clear");
12        System.out.println(x:"-----");
13    }
14    public static void main(String[] args) {
15        Scanner sc = new Scanner (System.in);
16        System.out.println(x:"Masukkan kapasitas Queue");
17        int n = sc.nextInt();
18        Queue Q = new Queue(n);
19        int pilih;
20        do {
21            menu();
22            pilih = sc.nextInt();
23            switch (pilih) {
24                case 1:
25                    System.out.println(x:"Masukkan data baru");
26                    int datamasuk = sc.nextInt();
27                    Q.Enqueue(datamasuk);
28                    break;
29                case 2:
30                    int datakeluar = Q.Dequeue();
31                    if(datakeluar != 0) {
32                        System.out.println("Data yang dikeluarkan: "+ datakeluar);
33                    }
34                    break;
35                case 3:
36                    Q.print();
37                    break;
38                case 4:
39                    Q.peek();
40                    break;
41                case 5:
42                    Q.clear();
43                    break;
44            }
45        } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
46        sc.close();
47    }
48 }
49
50 }
51
52

```



Verifikasi Hasil :

```
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
```

10.2.3 Pertanyaan

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

Pada konstruktor, nilai awal atribut front dan rear diatur ke -1 sedangkan size diatur ke 0 untuk menandakan bahwa pada awalnya antrian (queue) dalam keadaan kosong.

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

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

Potongan kode if (rear == max - 1) { rear = 0; dalam method Enqueue memiliki maksud dan kegunaan untuk menangani kondisi di mana rear mencapai batas maksimal dari array antrian (max - 1) kode tersebut membantu dalam mengimplementasikan antrian sirkular

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

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

Potongan kode if (front == max - 1) { front = 0; } ini membantu dalam mengimplementasikan antrian sirkular, di mana penghapusan elemen dari depan antrian akan terus dilakukan dengan melanjutkan dari awal array setelah mencapai batas akhir dari array yang digunakan untuk menyimpan elemen-antrian.

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

pada method print, perulangan dimulai dari front karena antrian tidak selalu dimulai dari indeks 0. karena elemen mungkin dimulai dari posisi lain selain 0 tergantung pada bagaimana elemen-elemen dimasukkan dan dihapus dari antrian.

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

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

Potongan kode i = (i + 1) % max; dalam method print digunakan untuk menggerakkan variabel i ke elemen berikutnya dalam antrian, dengan memperhitungkan sifat sirkular dari antrian.

6. Tunjukkan potongan kode program yang merupakan queue overflow!

```
public void Enqueue (int dt) {
    if (isFull()) {
        System.out.println(x:"Queue sudah penuh");
    } else {
```

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!

```
if (isFull()) {
    System.out.println(x:"Queue sudah penuh");
    System.exit(status:0);
```

```
int Dequeue (){
    int dt = 0;
    if (isEmpty()) {
        System.out.println(x:"Queue Sudah kosong");
        System.exit(dt);
```

10.3 Praktikum 2

Waktu percobaan : 45 menit

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

10.3.1 Langkah-langkah Percobaan

1. Perhatikan Diagram Class berikut ini:

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

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

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

```
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;
}
```

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**.

```
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("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;
}

```

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 modifikasi perlu dilakukan pada method **peek** dan method **print**.

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

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

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

8. Buat fungsi **main**, deklarasikan Scanner dengan nama **sc**
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.nextInt();  
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 = 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);
```

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

10.3.2 Verifikasi Hasil Percobaan

Samakan hasil compile kode program Anda dengan gambar berikut ini.

```

Output - Queue (run) x
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 1300000.0
32940 Susan Surabaya 39 4.2E7
Jumlah elemen = 2
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek Antrian terdepan
4. Cek Semua Antrian
-----

```


Nasabah Class :

```

1 package Praktikum2;
2 public class Nasabah26 {
3     String norek, nama, alamat;
4     int umur;
5     double saldo;
6
7     Nasabah26(String norek, String nama, String alamat, int umur, double saldo) {
8         this.norek = norek;
9         this.nama = nama;
10        this.alamat = alamat;
11        this.umur = umur;
12        this.saldo = saldo;
13    }
14 }
15

```

Queue26 Class :

```

1 package Praktikum2;
2 public class Queue2 {
3     Nasabah26[] data;
4     int front, rear, size, max;
5
6     Queue2(int n) {
7         max = n;
8         data = new Nasabah26[max];
9         size = 0;
10        front = rear = -1;
11    }
12
13    boolean isEmpty() {
14        return size == 0;
15    }
16
17    boolean isFull() {
18        return size == max;
19    }
20
21    void peek() {
22        if (!isEmpty()) {
23            System.out.println("Antrian terdepan: " + data[front].norek + " " + data[front].nama + " " + data[front].alamat +
24                " " + data[front].umur + " " + data[front].saldo);
25        } else {
26            System.out.println(x:"Queue masih kosong");
27        }
28    }
29
30    void print() {
31        if (isEmpty()) {
32            System.out.println(x:"Queue masih kosong");
33        } else {
34            int i = front;
35            while (i != rear) {

```



```

36         System.out.println(data[i].norek + " " + data[i].nama + " " + data[i].alamat +
37             " " + data[i].umur + " " + data[i].saldo);
38         i = (i + 1) % max;
39     }
40     System.out.println(data[i].norek + " " + data[i].nama + " " + data[i].alamat +
41         " " + data[i].umur + " " + data[i].saldo);
42     System.out.println("Jumlah elemen = " + size);
43 }
44 }
45
46 void peekRear() {
47     if (!isEmpty()) {
48         System.out.println("Antrian paling belakang: " + data[rear].norek + " " + data[rear].nama + " " + data[rear].alamat +
49             " " + data[rear].umur + " " + data[rear].saldo);
50     } else {
51         System.out.println(x:"Queue masih kosong");
52     }
53 }
54
55 void clear() {
56     if (isEmpty()) {
57         front = rear = -1;
58         size = 0;
59         System.out.println(x:"Queue berhasil dikosongkan");
60     } else {
61         System.out.println(x:"Queue masih kosong");
62     }
63 }
64
65 void Enqueue(Nasabah26 dt) {
66     if (isFull()) {
67         System.out.println(x:"Queue sudah penuh");
68     } else {
69         if (isEmpty()) {
70             front = rear = 0;
71         } else {
72             rear = (rear + 1) % max;
73         }
74         data[rear] = dt;
75         size++;
76     }
77 }
78
79 Nasabah26 Dequeue() {
80     if (isEmpty()) {
81         System.out.println(x:"Queue masih kosong");
82         return null;
83     } else {
84         Nasabah26 dt = data[front];
85         size--;
86         if (isEmpty()) {
87             front = rear = -1;
88         } else {
89             front = (front + 1) % max;
90         }
91         return dt;
92     }
93 }
94 }
95

```



QueueMain26 :

```

1 package Praktikum2;
2 import java.util.Scanner;
3
4 public class QueueMain2 {
5
6     static void menu() {
7         System.out.println("Pilih Menu :");
8         System.out.println("1. Antrian Baru ");
9         System.out.println("2. Antrian Keluar ");
10        System.out.println("3. Cek antrian terdepan");
11        System.out.println("4. Cek semua Antrian");
12        System.out.println("5. Cek Antrian paling belakang");
13        System.out.println("-----");
14    }
15
16    public static void main(String[] args) {
17
18        Scanner sc = new Scanner(System.in);
19        System.out.print("Masukkan Kapasitas queue: ");
20        int Jumlah = sc.nextInt();
21        Queue2 antri = new Queue2(Jumlah);
22
23        int pilih;
24        do {
25            menu();
26            pilih = sc.nextInt();
27            switch (pilih) {
28                case 1:
29                    System.out.print("No Rekening: ");
30                    String norek = sc.next();
31                    System.out.print("Nama: ");
32                    String nama = sc.next();
33                    System.out.print("Alamat: ");
34                    String alamat = sc.next();
35                    System.out.print("Umur: ");
36                    int umur = sc.nextInt();
37                    System.out.print("Saldo: ");
38                    double saldo = sc.nextDouble();
39                    Nasabah26 nb = new Nasabah26(norek, nama, alamat, umur, saldo);
40                    sc.nextLine();
41                    antri.Enqueue(nb);
42                    break;
43                case 2:
44                    Nasabah26 data = antri.Dequeue();
45                    if (data != null) {
46                        System.out.println("Antrian yang keluar: " + data.norek + " " + data.nama + " " + data.alamat + " " + data.umur + " " + data.saldo);
47                    }
48                    break;
49                case 3:
50                    antri.peek();
51                    break;
52                case 4:
53                    antri.print();
54                    break;
55                case 5:
56                    antri.peekRear();
57                    break;
58            }
59        } while (pilih >= 1 && pilih <= 5);
60        sc.close();
61    }
62 }
63

```



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

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
5. Cek Antrian paling belakang

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
5. Cek Antrian paling belakang

4

12345 Dewi Malang 23 1300000.0

32940 Susan Surabaya 39 4.2E7

Jumlah elemen = 2

Pilih Menu :

1. Antrian Baru
2. Antrian Keluar
3. Cek antrian terdepan
4. Cek semua Antrian



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;
}
```

kondisi if tersebut memeriksa apakah semua atribut dari objek data memiliki nilai yang tidak null (untuk string) dan tidak sama dengan 0 (untuk integer dan double). Jika semua kondisi terpenuhi, maka pesan yang berisi informasi tentang nasabah yang keluar akan dicetak.

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!

Queue26 class

```
void peekRear() {
    if (!isEmpty()) {
        System.out.println("Antrian paling belakang: " + data[rear].norek + " " + data[rear].nama + " " + data[rear].alamat +
            " " + data[rear].umur + " " + data[rear].saldo);
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}
```

QueueMain26 class

```
24 do {
25     menu();
26     pilih = sc.nextInt();
27     switch (pilih) {
28         case 1:
29             System.out.print(s:"No Rekening: ");
30             String norek = sc.next();
31             System.out.print(s:"Nama: ");
32             String nama = sc.next();
33             System.out.print(s:"Alamat: ");
34             String alamat = sc.next();
35             System.out.print(s:"Umur: ");
36             int umur = sc.nextInt();
37             System.out.print(s:"Saldo: ");
38             double saldo = sc.nextDouble();
39             Nasabah26 nb = new Nasabah26(norek, nama, alamat, umur, saldo);
40             sc.nextLine();
41             antri.Enqueue(nb);
42             break;
43         case 2:
44             Nasabah26 data = antri.Dequeue();
45             if (data != null) {
46                 System.out.println("Antrian yang keluar: " + data.norek + " " + data.nama + " " + data.alamat + " " + data.umur + " "
47                     + data.saldo);
48             }
49             break;
50         case 3:
51             antri.peek();
52             break;
53         case 4:
54             antri.print();
55             break;
56         case 5:
57             antri.peekRear();
58             break;
59     } while (pilih >= 1 && pilih <= 5);
60 }
61 sc.close();
```



```

Masukkan kapasitas queue : 5
Pilih Menu :
1. Antrian Baru
2. Antrian Keluar
3. Cek antrian terdepan
4. Cek semua Antrian
5. Cek Antrian paling belakang
-----
    
```

10.4 Tugas

1. Buatlah program antrian untuk mengilustasikan pesanan disebuah warung. Ketika seorang pembeli akan mengantri, maka dia harus mendaftarkan nama, dan nomor HP seperti yang digambarkan pada Class diagram berikut:

| Pembeli |
|----------------------------------|
| nama: String noHP: int |
| Pembeli(nama: String, noHP: int) |

Class diagram Queue digambarkan sebagai berikut:

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

Keterangan:

- Method create(), isEmpty(), isFull(), enqueue(), dequeue() dan print(), kegunaannya sama seperti yang telah dibuat pada Praktikum
- Method peek(): digunakan untuk menampilkan data Pembeli yang berada di posisi antrian paling depan
- Method peekRear(): digunakan untuk menampilkan data Pembeli yang berada di posisi antrian paling belakang
- Method peekPosition(): digunakan untuk menampilkan seorang pembeli (berdasarkan nama) posisi antrian ke berapa
- Method daftarPembeli(): digunakan untuk menampilkan data seluruh pembeli

Pembeli Class

```

1  package Tugas;
2
3  public class Pembeli {
4      String nama;
5      int noHp;
6
7      Pembeli (String nama, int noHp){
8          this.nama = nama;
9          this.noHp = noHp;
10     }
11     Pembeli (){
12
13     }
14

```



PembeliMain Class

```

1  package Tugas;
2  import java.util.Scanner;
3
4  public class PembeliMain {
5
6      static void Menu (){
7          System.out.println("-----");
8          System.out.println("\tPilih menu: ");
9          System.out.println("1. Antrean baru");
10         System.out.println("2. Antrean keluar");
11         System.out.println("3. Cek antrean terdepan");
12         System.out.println("4. Cek antrean paling belakang");
13         System.out.println("5. Cek semua antrean");
14         System.out.println("6. Cek posisi pembeli");
15         System.out.println("-----");
16     }
17     public static void main(String[] args) {
18         Scanner sc = new Scanner(System.in);
19
20         System.out.print("Masukan kapasitas antrean: ");
21         int jumlah = sc.nextInt();
22         Queue pembeli = new Queue(jumlah);
23         int pilih;
24
25         do {
26             Menu();
27             System.out.print("Masukkan pilihan: ");
28             pilih = sc.nextInt();
29             switch (pilih) {
30                 case 1:
31                     System.out.println("-----");
32                     System.out.print("Nama      : ");
33                     String nama = sc.next();
34                     System.out.print("No HP      : ");
35                     int hp = sc.nextInt();
36                     Pembeli pbl = new Pembeli(nama, hp);
37                     pembeli.Enqueue(pbl);
38                     break;
39                 case 2:
40                     Pembeli data = pembeli.Dequeue();
41                     if (!"".equals(data.nama) && data.noHp != 0) {
42                         System.out.println("Antrian yang keluar: " + data.nama + " " + data.noHp);
43                     }
44                     break;
45                 case 3:
46                     pembeli.peek();
47                     break;
48                 case 4:
49                     pembeli.peekRear();
50                     break;
51                 case 5:
52                     pembeli.daftarPembeli();
53                     break;
54                 case 6:
55                     System.out.println("Masukkan nama pembeli: ");
56                     String namaPembeli = sc.next();
57                     pembeli.peekPosition(namaPembeli);
58             }
59         } while (pilih < 7 && pilih > 0);
60         sc.close();
61     }
62 }

```




Queue Class

```

1 package Tugas;
2
3 public class Queue {
4     Pembeli antrian[];
5     int front, rear, size, max;
6
7     Queue (int n){
8         max = n;
9         antrian = new Pembeli[max];
10        size = 0;
11        front = rear = -1;
12    }
13    boolean isEmpty () {
14        if (size == 0) {
15            return true;
16        }
17        else {
18            return false;
19        }
20    }
21    boolean isFull () {
22        if (size == max) {
23            return true;
24        }
25        else {
26            return false;
27        }
28    }
29    void Enqueue (Pembeli dt){
30        if (isFull()) {
31            System.out.println("Queue sudah penuh");
32        }
33        else {
34            if (isEmpty()) {
35                front = rear = 0;
36            }
37            else {
38                if (rear == max - 1) {
39                    rear = 0;
40                }
41                else {
42                    rear++;
43                }
44                antrian[rear] = dt;
45                size++;
46            }
47        }
48    }
49    Pembeli Dequeue () {
50        Pembeli dt = new Pembeli();
51        if (isEmpty()) {
52            System.out.println("Queue masih kosong");
53        }
54        else {
55            dt = antrian[front];
56            size--;
57            if (isEmpty()) {
58                front = rear = -1;
59            }
60            else {
61                if (front == max - 1) {
62                    front = 0;
63                }
64                else {
65                    front++;
66                }
67            }
68        }
69        return dt;
70    }
71    void peek () {
72        if (isEmpty()) {
73            System.out.println("Antrian terdapat: " + antrian[front].nama + " " + antrian[front].noHp);
74        }
75        else {
76            System.out.println("Queue masih kosong");
77        }
78    }
79    void print () {
80        if (isEmpty()) {
81            System.out.println("Queue masih kosong");
82        }
83        else {
84            int i = front;
85            while (i != rear) {
86                System.out.println(antrian[i].nama + " " + antrian[i].noHp);
87                i = (i + 1) % max;
88            }
89            System.out.println(antrian[front].nama + " " + antrian[front].noHp);
90            System.out.println("Jumlah elemen = " + size);
91        }
92    }
93    void peekrear () {
94        if (isEmpty()) {
95            System.out.println("Antrian paling belakang: " + antrian[rear].nama + " " + antrian[rear].noHp);
96        }
97        else {
98            System.out.println("Queue masih kosong");
99        }
100    }
101    void peekPosition (String pembeli){
102        if (isEmpty()) {
103            int position = -1;
104            for (int i = 0; i < size; i++) {
105                int index = (front + i) % max;
106                if (antrian[index].nama.equals(pembeli)) {
107                    position = i + 1;
108                    if (position != 1) {
109                        System.out.println("Posisi " + pembeli + " dalam antrian: " + position);
110                    }
111                    else {
112                        System.out.println("Posisi " + pembeli + " tidak diketahui");
113                    }
114                    break;
115                }
116            }
117        }
118        else {
119            System.out.println("Queue masih kosong");
120        }
121    }
122    void daftarPembeli () {
123        if (isEmpty()) {
124            int i = front;
125            while (i != rear) {
126                System.out.println(antrian[i].nama + "\t" + antrian[i].noHp);
127                i = (i + 1) % max;
128            }
129            System.out.println(antrian[i].nama + "\t" + antrian[i].noHp);
130            System.out.println("Jumlah elemen = " + size);
131        }
132        else {
133            System.out.println("Queue masih kosong");
134        }
135    }
136    }
137 }

```

Hasil :



```
Masukan kapasitas antrean: 2
-----
      Pilih menu:
1. Antrean baru
2. Antrean keluar
3. Cek antrean terdepan
4. Cek antrean paling belakang
5. Cek semua antrean
6. Cek posisi pembeli
-----
Masukkan pilihan: 1
-----
Nama      : joko
No HP     : 1234
-----
      Pilih menu:
1. Antrean baru
2. Antrean keluar
3. Cek antrean terdepan
4. Cek antrean paling belakang
5. Cek semua antrean
6. Cek posisi pembeli
-----
Masukkan pilihan: 1
-----
Nama      : Daa
No HP     : 34524
-----
      Pilih menu:
1. Antrean baru
2. Antrean keluar
3. Cek antrean terdepan
4. Cek antrean paling belakang
5. Cek semua antrean
6. Cek posisi pembeli
-----
Masukkan pilihan: 2
Antrian yang keluar: joko 1234
-----
```