NAMA : DEANISSA SHERLY SABILLA

KELAS : 1B SIB

ABSEN : 06

HEET VIII
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

Waktu percobaan : 45 menit

Pada percobaan ini, kita akan mengimplementasikan penggunaan class Queue.

8.2.1 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 behasil 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 paling depan

```
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.

8.2.2 Verifikasi Hasil Percobaan

Samakan hasil compile kode program Anda dengan gambar berikut ini.

```
Masukkan kapasitas queue : 6
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: 23
```

```
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----------------------------------
3
15
23
Jumlah elemen = 2

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
23
Jumlah elemen = 1
```

INPUT :

CLASS QUEUE :

Pratikum01 > Queue06.java > Press 'Enter' to

```java
public class Queue06 {
int [] data;
int front;
int rear;
int size;
int max;
public Queue06(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;
    }
}
```

```java
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 masih kosong");
    }
```

```java
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++;
    }
}
```

```
75    public int Dequeue(){
76        int dt = 0;
77        if (IsEmpty()) {
78            System.out.println(x:"Queue masih kosong");
79        } else {
80            dt = data[front];
81            size--;
82            if (IsEmpty()) {
83                front = rear = -1;
84            } else {
85                if (front == max -1) {
86                    front = 0;
87                } else {
88                    front++;
89                }
90            }
91        }
92        return dt;
93    }
94    }
```

Hasil Output :

```
Masukkan kapasitas queue :
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 : 23
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
---------------------------------
3
15 23
Jumlah elemen = 2
```

```
Masukkan operasi yang diing
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
23
Jumlah elemen = 1
```

8.2.3 Pertanyaan

1. Pada konstruktor, mengapa nilai awal atribut front dan rear bernilai -1, sementara atribut size bernilai 0?
    - nilai awal atribut front dan rear diatur sebagai -1 karena -1 merupakan nilai yang tidak valid atau menandakan bahwa antrian (queue) kosong.
    - Sementara itu, atribut size diatur sebagai 0 karena pada awalnya antrian kosong, sehingga tidak ada elemen yang ada di dalamnya.
2. Pada method **Enqueue**, jelaskan maksud dan kegunaan dari potongan kode berikut!

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

    - Digunakan untuk memeriksa apakah posisi rear telah mencapai atau melewati indeks maksimum yang diizinkan dalam array yang mewakili antrian.
3. Pada method **Dequeue**, jelaskan maksud dan kegunaan dari potongan kode berikut!

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

    - Digunakan untuk memeriksa apakah posisi front telah mencapai atau melewati indeks maksimum yang diizinkan dalam array yang mewakili antrian.
4. Pada method **print**, mengapa pada proses perulangan variabel i tidak dimulai dari 0 (**`int i=0`**), melainkan **`int i=front`**?
    - Karena disini ingin mencetak elemen-elemen yang sebenarnya ada dalam antrian, bukan semua elemen dalam array yang mungkin sudah diisi sebelumnya dan kemudian dihapus.
5. Perhatikan kembali method **print**, jelaskan maksud dari potongan kode berikut!

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

    - Digunakan untuk memindahkan indeks i ke elemen berikutnya dalam array yang mewakili antrian.
6. Tunjukkan potongan kode program yang merupakan queue overflow!
    -

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

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!

   - Menambah **System.exit(0);** pada kode method Enqueue dan Dequeue

```
58  public void Enqueue (int dt){
59      if (IsFull()) {
60          System.out.println(x:"Queue sudah penuh");
61          System.exit(status:0);
62      } else {
```

```
76  public int Dequeue(){
77      int dt = 0;
78      if (IsEmpty()) {
79          System.out.println(x:"Queue masih kosong");
80          System.exit(status:0);
```

   - **Hasil Output Enqueue**

```
Masukkan kapasitas queue : 3
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
---------------------------------
1
Masukkan data baru : 10
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
---------------------------------
1
Masukkan data baru : 20
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
---------------------------------
1
Masukkan data baru : 30
```

```
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
---------------------------------
1
Masukkan data baru : 10
Queue sudah penuh
```

   Hasil Output Dequeue

```
Masukkan kapasitas queue : 2
Masukkan operasi yang diinginkan :
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
---------------------------------
2
Queue masih kosong
```

8.3 Praktikum 2

Waktu percobaan : 45 menit

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

8.3.1 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 meodifikasi perlu dilakukan pada method **peek** dan method **print**.

```
public void peek() {
    if (!IsEmpty()) {
        System.out.println("Elemen terdepan: " + data[front].norek + " " + data[front].nama
                + " " + data[front].alamat + " " + data[front].umur + " " + data[front].saldo);
    } else {
        System.out.println("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.

8.3.2 Verifikasi Hasil Percobaan

Samakan hasil compile kode program Anda dengan gambar berikut ini.

```
Masukkan kapasitas queue : 4
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
---------------------------------
1
No rekening: 1200046675
Nama: Arif
Alamat: Sukun, Malang
Umur: 25
Saldo: 12000000

Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
---------------------------------
1
No rekening: 1200198733
Nama: Dewi
Alamat: Rungkut, Surabaya
Umur: 30
Saldo: 8600000

Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
---------------------------------
4
1200046675 Arif Sukun, Malang 25 1.2E7
1200198733 Dewi Rungkut, Surabaya 30 8600000.0
Jumlah elemen = 2
```

```
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
---------------------------------
3
Elemen terdepan : 1200046675 Arif Sukun, Malang 25 1.2E7
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
---------------------------------
2
Antrian yang keluar : 1200046675 Arif Sukun, Malang 25 1.2E7
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
---------------------------------
4
1200198733 Dewi Rungkut, Surabaya 30 8600000.0
Jumlah elemen = 1
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
---------------------------------
```

INPUT :

CIASS NASABAH :

```java
public class Nasabah06 {
    String norek, nama, alamat;
    int umur;
    double saldo;
    Nasabah06 (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;
    }

    Nasabah06(){

    }
}
```

CLASS QUEUE :

```
3  public class QueueNasabah06 {
4      Nasabah06 [] data;
5      int front;
6      int rear;
7      int size;
8      int max;
9      public QueueNasabah06(int n){
10         max = n;
11         data = new Nasabah06 [max];
12         size = 0;
13         front = rear = -1;
14     }
15     public boolean IsEmpty(){
16         if (size == 0) {
17             return true;
18         } else {
19             return false;
20         }
21     }
22     public boolean IsFull() {
23         if (size == max) {
24             return true;
25         } else {
26             return false;
27         }
```

```
30     public void peek() {
31         if (!IsEmpty()) {
32             System.out.println("Elemen terdepan : " + data[front].norek + " " + data[front].n
33         } else {
34             System.out.println(x:"Queue masih kosong");
35         }
36     }
37     public void print() {
38         if (IsEmpty()) {
39             System.out.println(x:"Queue masih kosong");
40         } else {
41             int i = front;
42             while (i != rear){
43                 System.out.println(data[i].norek + " " +data[i].nama + " " + data[i].alamat +
44                 i = (i + 1) % max;
45             }
46             System.out.print(data[i].norek + " " +data[i].nama + " " + data[i].alamat + " " +
47             System.out.println("Jumlah elemen = "+size);
48         }
49     }
```

```
57     public void Enqueue (Nasabah06 dt){
58         if (IsFull()) {
59             System.out.println(x:"Queue sudah penuh");
60         } else {
61             if (IsEmpty()) {
62             front = rear = 0;
63             } else {
64                 if (rear == max -1){
65                     rear = 0;
66                 } else {
67                     rear++;
68                 }
69             }
70             data[rear] = dt;
71             size++;
72         }
73     }
```

```
74          public Nasabah06 Dequeue(){
75              Nasabah06 dt = new Nasabah06();
76              if (IsEmpty()) {
77                  System.out.println(x:"Queue masih kosong");
78                  System.exit(status:0);
79              } else {
80                  dt = data[front];
81                  size--;
82                  if (IsEmpty()) {
83                      front = rear = -1;
84                  } else {
85                      if (front == max -1) {
86                          front = 0;
87                      } else {
88                          front++;
89                      }
90                  }
91              }
92              return dt;
93          }
94  }
```

CLASS MAIN :

```
5   public class Queue06 {
6       public static void menu () {
7           System.out.println(x:"Pilih Menu :");
8           System.out.println(x:"1. Antrian Baru");
9           System.out.println(x:"2. Antrian Keluar");
10          System.out.println(x:"3. Cek Antrian Terdepan");
11          System.out.println(x:"4. Cek Semua Antrian");
12          System.out.println(x:"---------------------------------");
13      }
            Run | Debug
14          public static void main(String[] args) {
15              Scanner sc = new Scanner (System.in);
16              System.out.print(s:"Masukkan kapasitas : ");
17              int n = sc.nextInt();
18              QueueNasabah06 antri = new QueueNasabah06(n);
19              int pilih;
20              do {
21                  menu();
22                  pilih = sc.nextInt();
23                  sc.nextLine();
24                  switch (pilih){
25                      case 1 :
26                      System.out.print(s:"No Rekening : ");
27                      String norek = sc.nextLine();
28                      System.out.print(s:"Nama : ");
```

```
32                      System.out.print(s:"Umur : ");
33                      int umur = sc.nextInt();
34                      System.out.print(s:"Saldo : ");
35                      double saldo = sc.nextDouble();
36                      Nasabah06 nb = new Nasabah06 (norek, nama, alamat, umur, saldo);
37                      sc.nextLine();
38                      antri.Enqueue(nb);
39                      break;
40                      case 2 :
41                      Nasabah06 data = antri.Dequeue();
42                      if (!"".equals(data.norek) && !"".equals(data.nama) && !"".equals(data.alamat
43                      && data.umur !=0 && data.saldo !=0) {
44                          System.out.println("Antrian yang keluar : " + data.norek + " " +data.nama
45                          break;
46                      }
47                      case 3 :
48                      antri.peek();
49                      break;
50                      case 4 :
51                      antri.print();
52                      break;
53                  }
54              } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4);
```

Hasil Output :

```
Masukkan kapasitas : 4
Pilih Menu :
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
---------------------------------
1
No Rekening : 1200046675
Nama : Arif
Alamat : Sukun, Malang
Umur : 25
Saldo : 12000000
Pilih Menu :
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
---------------------------------
1
No Rekening : 1200198733
Nama : Dewi
Alamat : Rungkut, Surabaya
Umur : 30
Saldo : 8600000
```

```
Pilih Menu :
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
---------------------------------
4
1200046675 Arif Sukun, Malang 25 1.2E7
1200198733 Dewi Rungkut, Surabaya 30 8600000.0Jumlah elemen = 2
```

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;
}
```

   - Digunakan untuk memeriksa apakah objek data yang diperoleh dari operasi Dequeue memiliki nilai yang valid sebelum melakukan tindakan lebih lanjut.

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

- **Menambahkan method peek Rear ke Class Queue**

```
93          }
94          public void peekRear() {
95              if (!IsEmpty()) {
96                  System.out.println("Elemen terbelakang : " + data[rear].norek + " " + data[rear].nama + " " + data[rea
97                  data[rear].umur + " " + data[rear].saldo);
98              } else {
99                  System.out.println(x:"Queue masih kosong");
100             }
101         }
102 }
```

- **Hasil Output :**

```
Masukkan kapasitas : 4
Pilih Menu :
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
5. Cek Antrian Terbelakang
---------------------------------
1
No Rekening : 120000000
Nama : Arif
Alamat : Sukun
Umur : 20
Saldo : 1250000
Pilih Menu :
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
5. Cek Antrian Terbelakang
---------------------------------
1
No Rekening : 1300000000
Nama : Dewi
Alamat : Rungkut
Umur : 23
Saldo : 1350000
```

```
Pilih Menu :
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
5. Cek Antrian Terbelakang
---------------------------------
5
Elemen terbelakang : 1300000000 Dewi Rungkut 23 1350000.0
```

8.4 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

INPUT :

CLASS Pembeli :

Pratikum03 > Pembeli06.java > Pembeli06 > Pembeli06()

```java
package Pratikum03;
public class Pembeli06 {
    String nama;
    int noID;
    char jenisKelamin;
    int umur;

    Pembeli06 (String nama, int noID, char jenisKelamin, int umur){
        this.nama = nama;
        this.noID =noID;
        this.jenisKelamin = jenisKelamin;
        this.umur = umur;
    }
    Pembeli06 (){

    }
}
```

Class Queue :

```java
public class QueuePembeli06 {
    Pembeli06 [] antrian;
        int front;
        int rear;
        int size;
        int max;
        public QueuePembeli06(int n){
            max = n;
            antrian = new Pembeli06 [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;
            }
        }
```

```
29      public void peek() {
30          if (!IsEmpty()) {
31              System.out.println("Elemen terdepan : " + antrian[front].nama + " " + antrian[front].noID + " " + antr
32          } else {
33              System.out.println(x:"Queue masih kosong");
34          }
35      }
36      public void print() {
37          if (IsEmpty()) {
38              System.out.println(x:"Queue masih kosong");
39          } else {
40              int i = front;
41              while (i != rear){
42                  System.out.println(antrian[i].nama + " " +antrian[i].noID + " " + antrian[i].jenisKelamin + " " +
43                  i = (i + 1) % max;
44              }
45              System.out.print(antrian[i].nama + " " + antrian[i].noID + " " + antrian[i].jenisKelamin + " " + antri
46              System.out.println("Jumlah elemen = "+size);
47          }
48      }
```

```
49      public void Enqueue (Pembeli06 antri){
50          if (IsFull()) {
51              System.out.println(x:"Queue sudah penuh");
52          } else {
53              if (IsEmpty()) {
54              front = rear = 0;
55              } else {
56                  if (rear == max -1){
57                      rear = 0;
58                  } else {
59                      rear++;
60                  }
61              }
62              antrian[rear] = antri;
63              size++;
64          }
65      }
```

```
66      public Pembeli06 Dequeue(){
67          Pembeli06 antri = new Pembeli06();
68          if (IsEmpty()) {
69              System.out.println(x:"Queue masih kosong");
70              System.exit(status:0);
71          } else {
72              antri = antrian[front];
73              size--;
74              if (IsEmpty()) {
75                  front = rear = -1;
76              } else {
77                  if (front == max -1) {
78                      front = 0;
79                  } else {
80                      front++;
81                  }
82              }
83          }
84          return antri;
85      }
```

```
86          public void peekRear() {
87              if (!IsEmpty()) {
88                  System.out.println("Elemen terbelakang : " + antrian[rear].nama + " " + antrian[r
89                  antrian[rear].umur);
90              } else {
91                  System.out.println(x:"Queue masih kosong");
92              }
93          }
94          public void peekPosition(String nama) {
95              if (IsEmpty()) {
96                  System.out.println(x:"Queue masih kosong");
97              } else {
98                  int posisi = -1;
99                  int count = 0;
100                 int i = front;
101                 while (i != rear) {
102                     if (antrian[i].nama.equals(nama)) {
103                         posisi = count;
104                         break;
105                     }
106                     count++;
107                     i = (i + 1) % max;
108                 }
```

```
112                 if (posisi != -1) {
113                     System.out.println("Posisi antrian " + nama + " adalah: " + posisi);
114                 } else {
115                     System.out.println(nama + " tidak ditemukan dalam antrian");
116                 }
117             }
118         }
119         public void daftarPembeli() {
120             if (IsEmpty()) {
121                 System.out.println(x:"Queue masih kosong");
122             } else {
123                 int i = front;
124                 int count = 1;
125                 while (i != rear) {
126                     System.out.println("Antrian ke-" + count + ": " + antrian[i].nama + " " + anti
127                     count++;
128                     i = (i + 1) % max;
129                 }
130                 System.out.println("Antrian ke-" + count + ": " + antrian[i].nama + " " + antrian
131                 System.out.println("Jumlah elemen = " + size);
132             }
133         }
134 }
```

CLASS Main :

```
1   package Pratikum03;
2   import java.util.Scanner;
3   public class PembeliMain06 {
4
5       public static void menu () {
6           System.out.println(x:"Pilih Menu :");
7           System.out.println(x:"1. Antrian Baru");
8           System.out.println(x:"2. Antrian Keluar");
9           System.out.println(x:"3. Cek Antrian Terdepan");
10          System.out.println(x:"4. Cek Semua Antrian");
11          System.out.println(x:"5. Cek Antrian Terbelakang");
12          System.out.println(x:"6. Menampilkan Antrian Berdasarkan (nama)");
13          System.out.println(x:"7. Menampilkan Semua Antrian");
14          System.out.println(x:"--------------------------------");
15      }
            Run | Debug
16          public static void main(String[] args) {
17              Scanner sc = new Scanner (System.in);
18              System.out.print(s:"Masukkan kapasitas : ");
19              int n = sc.nextInt();
20              QueuePembeli06 antri = new QueuePembeli06(n);
21              int pilih;
22              do {
23                  menu();
24                  pilih = sc.nextInt();
```

```java
Pratikum03 > PembeliMain06.java > PembeliMain06 > main(String[])
3   public class PembeliMain06 {
16          public static void main(String[] args) {
26              switch (pilih){
27                  case 1 :
28                  System.out.print(s:"Nama : ");
29                  String nama = sc.nextLine();
30                  System.out.print(s:"No ID : ");
31                  int noID = sc.nextInt();
32                  System.out.print(s:"Jenis Kelamin (L/P) : ");
33                  char jenisKelamin = sc.next().charAt(index:0);
34                  System.out.print(s:"Umur : ");
35                  int umur = sc.nextInt();
36                  Pembeli06 nb = new Pembeli06 (nama, noID, jenisKelamin, umur);
37                  sc.nextLine();
38                  antri.Enqueue(nb);
39                  break;
40                  case 2 :
41                  Pembeli06 antrian = antri.Dequeue();
42                  if (!"".equals(antrian.nama) && !"".equals(antrian.noID) && !"".equals(antrian.jenisKelamin)
43                  && antrian.umur !=0) {
44                      System.out.println("Antrian yang keluar : " + antrian.nama + " " +antrian.noID + " " + ant
45                      break;
46                  }
47                  case 3 :
```

```java
47                  case 3 :
48                  antri.peek();
49                  break;
50                  case 4 :
51                  antri.print();
52                  break;
53                  case 5:
54                  antri.peekRear();
55                  break;
56                  case 6 :
57                  System.out.print(s:"Nama yang dicari: ");
58                  String namaCari = sc.nextLine();
59                  antri.peekPosition(namaCari);
60                  break;
61                  case 7 :
62                  antri.daftarPembeli();
63
64              }
65          } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5 || pilih == 6 || pilih ==
66      }
67  }
```

OUTPUT :

```
Masukkan kapasitas : 3
Pilih Menu :
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
5. Cek Antrian Terbelakang
6. Menampilkan Antrian Berdasarkan (nama)
7. Menampilkan Semua Antrian
---------------------------------
1
Nama : Dea
No ID : 1001
Jenis Kelamin (L/P) : P
Umur : 19
Pilih Menu :
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
5. Cek Antrian Terbelakang
6. Menampilkan Antrian Berdasarkan (nama)
7. Menampilkan Semua Antrian
---------------------------------
1
Nama : Nissa
No ID : 1002
Jenis Kelamin (L/P) : P
Umur : 20
```

```
Pilih Menu :
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
5. Cek Antrian Terbelakang
6. Menampilkan Antrian Berdasarkan (nama)
7. Menampilkan Semua Antrian
---------------------------------
1
Nama : Joko
No ID : 1003
Jenis Kelamin (L/P) : L
Umur : 26
```

Hasil pada Pilihan No 3 : Cek Antrian Terdepan

```
Pilih Menu :
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
5. Cek Antrian Terbelakang
6. Menampilkan Antrian Berdasarkan (nama)
7. Menampilkan Semua Antrian
---------------------------------
3
Elemen terdepan : Dea 1001 P 19
```

Hasil pada pilihan no 4 : Cek Semua Antrian

```
Pilih Menu :
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
5. Cek Antrian Terbelakang
6. Menampilkan Antrian Berdasarkan (nama)
7. Menampilkan Semua Antrian
---------------------------------
4
Dea 1001 P 19
Nissa 1002 P 23
Joko 1003 L 26Jumlah elemen = 3
```

Hasil pada pilihan no 5 : Menampilkan Antrian Terbelakang

```
Pilih Menu :
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
5. Cek Antrian Terbelakang
6. Menampilkan Antrian Berdasarkan (nama)
7. Menampilkan Semua Antrian
---------------------------------
5
Elemen terbelakang : Joko 1003 L 26
```

Hasil pada pilihan no 6 : Menampilkan Antrian berdasarkan (nama)

```
Pilih Menu :
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
5. Cek Antrian Terbelakang
6. Menampilkan Antrian Berdasarkan (nama)
7. Menampilkan Semua Antrian
---------------------------------
6
Nama yang dicari: Joko
Posisi antrian Joko adalah: 2
```

Hasil pada pilihan no 7 : Menampilkan Semua Antrian

```
Pilih Menu :
1. Antrian Baru
2. Antrian Keluar
3. Cek Antrian Terdepan
4. Cek Semua Antrian
5. Cek Antrian Terbelakang
6. Menampilkan Antrian Berdasarkan (nama)
7. Menampilkan Semua Antrian
---------------------------------
7
Antrian ke-1: Dea 1001 P 19
Antrian ke-2: Nissa 1002 P 20
Antrian ke-3: Joko 1003 L 26
Jumlah elemen = 3
```