LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

PERTEMUAN KE 10 JOBSHEET 8 QUEUE

Dosen Pengajar : Triana Fatmawati, S.T., M.T.

Muhammad Afiq Firdaus

2341760189 / 21

SIB1E

PROGRAM STUDI D-IV SISTEM INFORMASI BISNIS

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2024

8.2 Praktikum 1

buat class baru dengan nama Queue.

```
Queue21.java > Queue21
1   public class Queue21{
```

Tambahkan atribut-atribut Queue sesuai diagram class, kemudian tambahkan pula konstruktornya

```
Queue21.java > Queue21
1   public class Queue21{
2       int data[];
3       int front;
4       int rear;
5       int size;
6       int max;
7
8       public Queue21(int n){
9           max = n;
10          data = new int[max];
11          size = 0;
12          front = rear = -1;
13      }
```

Buat method IsEmpty bertipe boolean yang digunakan untuk mengecek apakah queue kosong.

```
15      public boolean IsEmpty(){
16          if(size == 0){
17              return true;
18          }else{
19              return false;
20          }
21      }
```

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

```
23      public boolean IsFull(){
24          if(size == max){
25              return true;
26          }else{
27              return false;
28          }
29      }
```

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

```
31      public void peek(){
32          if(!IsEmpty()){
33              System.out.println("Elemen terdepan: " + data[front]);
34          }else{
35              System.out.println(x:"Queue masih kosong");
36          }
37      }
```

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

```
39      public void print(){
40          if(IsEmpty()){
41              System.out.println(x:"Queue masih kosong");
42          }else{
43              int i = front;
44              while (i != rear){
45                  System.out.println(data[i] + " ");
46                  i = (i + 1) % max;
47              }
48              System.out.println(data[i] + " ");
49              System.out.println("Jumlah elemen = " + size);
50          }
51      }
```

Buat method clear bertipe void untuk menghapus semua elemen pada queue

```
53      public void clear(){
54          if(!IsEmpty()){
55              front = rear = -1;
56              size = 0;
57              System.out.println(x:"Queue berhasil dikosongkan");
58
59          }else{
60              System.out.println(x:"Queue masih kosong");
61          }
62      }
63
```

Buat method Enqueue bertipe void untuk menambahkan isi queue dengan parameter dt yang bertipe integer

```
64      public void Enqueue(int dt){
65          if(IsFull()){
66              System.out.println(x:"Queue sudah penuh");
67          }else{
68              if(IsEmpty()){
69                  front =  rear = 0;
70              }else{
71                  if(rear == max -1){
72                      rear = 0;
73                  }else{
74                      rear++;
75                  }
76              }
77              data[rear] =dt;
78              size++;
79          }
80      }
```

Buat method Dequeue bertipe int untuk mengeluarkan data pada queue di posisi paling depan

```
82      public int Dequeue(){
83          int dt = 0;
84          if(IsEmpty()){
85              System.out.println(x:"Queue masih kosong");
86          }else{
87              dt = data[front];
88              size--;
89              if(IsEmpty()){
90                  front = rear = -1;
91              }else{
92                  if(front == max -1){
93                      front = 0;
94                  }else{
95                      front++;
96                  }
97              }
98          }
99          return dt;
100     }
101 }
```

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

```
QueueMain21.java > QueueMain21
1   import java.util.*;
2   public class QueueMain21 {
3       public static void menu(){
4           System.out.println(x:"Masukkan operasi yang diinginkan: ");
5           System.out.println(x:"1. Enqueue");
6           System.out.println(x:"2. Dequeue");
7           System.out.println(x:"3. Print");
8           System.out.println(x:"4. Peek");
9           System.out.println(x:"5. Clear");
10          System.out.println(x:"-----------------------------------");
11
12      }
```

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

```
        Run | Debug
13      public static void main(String[] args) {
14          Scanner sc = new Scanner(System.in);
15
16          System.out.print(s:"Masukkan kapasitas queue: ");
17          int n = sc.nextInt();
```

Lakukan instansiasi objek Queue dengan nama Q dengan mengirimkan parameter n sebagai kapasitas elemen queue.

```
Queue21 Q = new Queue21(n);
```

Deklarasikan variabel dengan nama pilih bertipe integer untuk menampung pilih menu dari pengguna. 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.

```
22          int pilih;
23          do {
24              menu();
25              pilih = sc.nextInt();
26              switch (pilih) {
27                  case 1:
28                      System.out.print(s:"Masukkan data baru: ");
29                      int dataMasuk = sc.nextInt();
30                      Q.Enqueue(dataMasuk);
31                      break;
32                  case 2:
33                      int dataKeluar = Q.Dequeue();
34                      if (dataKeluar != 0) {
35                          System.out.println("Data yang dikeluarkan: " + dataKeluar);
36                          break;
37                      }
38                  case 3:
39                      Q.print();
40                      break;
41                  case 4:
42                      Q.peek();
43                      break;
44                  case 5:
45                      Q.clear();
46                      break;
47              }
48
49          } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
50      }
51  }
```

Compile dan jalankan class QueueMain, kemudian amati hasilnya.

```
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Prakt
 ke 10>  & 'C:\Program Files\Java\jdk-21\bin\java.exe' '-XX:+ShowCodeDet
nMessages' '-cp' 'C:\Users\HP\AppData\Roaming\Code\User\workspaceStorage
a03e18129d7deb5c80a\redhat.java\jdt_ws\Pertemuan ke 10_788c0dca\bin' 'Qu
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
```

8.2.3 Pertanyaan

1. Pada konstruktor, mengapa nilai awal atribut front dan rear bernilai -1, sementara atribut size bernilai 0?
2. Pada method Enqueue, jelaskan maksud dan kegunaan dari potongan kode berikut!

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

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

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

4. Pada method print, mengapa pada proses perulangan variabel i tidak dimulai dari 0 (int i=0), melainkan int i=front?
5. Perhatikan kembali method print, jelaskan maksud dari potongan kode berikut!

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

6. Tunjukkan potongan kode program yang merupakan queue overflow!
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

1. nilai awal atribut front dan rear diatur sebagai -1 karena -1 menandakan bahwa antrian (queue) saat itu kosong. Ini mengindikasikan bahwa tidak ada elemen di dalam antrian pada awalnya.
2. Maksud dan kegunaan dari code tersebut adalah bagian dari operasi enqueue pada struktur data antrian (queue). Dalam kode tersebut, kondisi rear == max - 1 memeriksa apakah indeks rear sudah mencapai batas maksimum yang ditentukan (max - 1).ini berarti antrian sudah penuh.
3. kode tersebut merupakan bagian dari operasi dequeue pada struktur data antrian (queue). Dalam kode tersebut, kondisi front == max - 1 memeriksa apakah indeks front sudah mencapai batas maksimum yang ditentukan (max - 1). ini berarti kita sudah sampai di akhir antrian.
4. Pada method print dalam implementasi kelas Queue21, proses perulangan variabel i dimulai dari front bukan dari 0 karena kita ingin mencetak elemen-elemen yang ada dalam antrian (queue) sesuai dengan urutan dari front ke rear. ini dilakukan untuk memastikan bahwa kita mencetak elemen secara berurutan, sesuai dengan cara penyimpanan data dalam array circular yang digunakan untuk mewakili queue.
5. Potongan kode i = (i + 1) % max; dalam method print pada implementasi kelas Queue21 memiliki tujuan untuk melanjutkan indeks **i** ke elemen berikutnya dalam array secara circular.

6. Queue overflow terjadi ketika kita mencoba menambahkan elemen ke dalam antrian yang sudah penuh (mencapai kapasitas maksimumnya). Dalam implementasi queue menggunakan array seperti yang disediakan di kelas Queue21, queue overflow dapat terjadi saat mencoba melakukan operasi enqueue (Enqueue(int dt)) pada antrian yang sudah mencapai kapasitas maksimumnya (max).
   Berikut adalah code yang menunjukkan Queue overflow

```
63  ck to add a breakpoint
64          public void Enqueue(int dt){
65              if(IsFull()){
66                  System.out.println(x:"Queue sudah penuh");
67              }else{
68                  if(IsEmpty()){
69                      front =  rear = 0;
70                  }else{
71                      if(rear == max -1){
72                          rear = 0;
73                      }else{
74                          rear++;
75                      }
76                  }
77                  data[rear] =dt;
78                  size++;
79              }
80          }
```

7.

```
103
104  // MODIFIKASI NOMER 7
105  public class Queue21 {
106      int data[];
107      int front;
108      int rear;
109      int size;
110      int max;
111
112      public Queue21(int n) {
113          max = n;
114          data = new int[max];
115          size = 0;
116          front = rear = -1;
117      }
118
119      public boolean IsEmpty() {
120          return size == 0;
121      }
122
123      public boolean IsFull() {
124          return size == max;
125      }
126
127      public void peek() {
128          if (!IsEmpty()) {
129              System.out.println("Elemen terdepan: " + data[front]);
130          } else {
131              System.out.println(x:"Queue masih kosong");
132              System.exit(status:0); // Menghentikan program saat terjadi queue underflow
133          }
134      }
135
```

```java
public void print() {
    if (IsEmpty()) {
        System.out.println(x:"Queue masih kosong");
        System.exit(status:0); // Menghentikan program saat terjadi queue underflow
    } 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(x:"Queue berhasil dikosongkan");
    } else {
        System.out.println(x:"Queue masih kosong");
        System.exit(status:0); // Menghentikan program saat terjadi queue underflow
    }
}
```

```java
public void Enqueue(int dt) {
    if (IsFull()) {
        System.out.println(x:"Queue sudah penuh. Overflow terjadi.");
        System.exit(status:0); // Menghentikan program saat terjadi queue overflow
    } else {
        if (IsEmpty()) {
            front = rear = 0;
        } else {
            if (rear == max - 1) {
                rear = 0;
            } else {
                rear++;
            }
        }
        data[rear] = dt;
        size++;
    }
}
```

```java
public int Dequeue() {
    int dt = 0;
    if (IsEmpty()) {
        System.out.println(x:"Queue masih kosong");
        System.exit(status:0); // Menghentikan program saat terjadi queue underflow
    } else {
        dt = data[front];
        size--;
        if (IsEmpty()) {
            front = rear = -1;
        } else {
            if (front == max - 1) {
                front = 0;
            } else {
                front++;
            }
        }
    }
    return dt;
}
```

Praktikum 2

buat class baru dengan nama Nasabah.

Tambahkan atribut-atribut Nasabah seperti pada Class Diagram, kemudian tambahkan pula konstruktornya

```
    String norek, nama, alamat;
    int umur;
    double saldo;

    Nasabah21 (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;
    }
```

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.

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.

```
    public Nasabah21() {
    }
    Nasabah21[] data;
    int front;
    int rear;
    int size;
    int max;

    public void Queue(int n){
        max = n;
        data = new Nasabah21[max];
        size = 0;
        front = rear = -1;
    }

    public Nasabah21(int n){
        max = n;
        data = new Nasabah21[max];
        size = 0;
        front = rear = -1;
    }


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

```
45
46      public boolean IsFull(){
47          if(size == max){
48              return true;
49          }else{
50              return false;
51          }
52      }
53
54      public void peek(){
55          if(!IsEmpty()){
56              System.out.println("Elemen terdepan: " + data[front].norek + " " + data[front].nama + " "
57              + data[front].alamat + " " + data[front].umur + " " + data[front].saldo);
58          }else{
59              System.out.println(x:"Queue masih kosong");
60          }
61      }
62
63      public void print(){
64          if(IsEmpty()){
65              System.out.println(x:"Queue masih kosong");
66          }else{
67              int i = front;
68              while (i != rear){
69                  System.out.println(data[i].norek + " " + data[i].nama + " " + data[i].alamat
70                  + " " + data[i].umur + " " + data[i].saldo);
71                  i = (i + 1) % max;
72              }
73              System.out.println(data[i].norek + " " + data[i].nama + " " + data[i].alamat
74              + " " + data[i].umur + " " + data[i].saldo);
75              System.out.println("Jumlah elemen = " + size);
76          }
```

```
79      public void clear(){
80          if(!IsEmpty()){
81              front = rear = -1;
82              size = 0;
83              System.out.println(x:"Queue berhasil dikosongkan");
84
85          }else{
86              System.out.println(x:"Queue masih kosong");
87          }
88      }
89
90      public void Enqueue(Nasabah21 dt){
91          if(IsFull()){
92              System.out.println(x:"Queue sudah penuh");
93          }else{
94              if(IsEmpty()){
95                  front =  rear = 0;
96              }else{
97                  if(rear == max -1){
98                      rear = 0;
99                  }else{
100                     rear++;
101                 }
102             }
103             data[rear] =dt;
104             size++;
105         }
106     }
```

```
107
108     public Nasabah21 Dequeue(){
109         Nasabah21 dt = new Nasabah21();
110         if(IsEmpty()){
111             System.out.println(x:"Queue masih kosong");
112         }else{
113             dt = data[front];
114             size--;
115             if(IsEmpty()){
116                 front = rear = -1;
117             }else{
118                 if(front == max -1){
119                     front = 0;
120                 }else{
121                     front++;
122                 }
123             }
124         }
125         return dt;
126     }
127 }
```

Baris program Nasabah dt = new Nasabah(); akan ditandai sebagai error, untuk mengatasinya, tambahkan konstruktor default di dalam class Nasabah.

```
14      public Nasabah21() {
15      }
```

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.

```
54  public void peek(){
55      if(!IsEmpty()){
56          System.out.println("Elemen terdepan: " + data[front].norek + " " + data[front].nama + " "
57          + data[front].alamat + " " + data[front].umur + " " + data[front].saldo);
58      }else{
59          System.out.println(x:"Queue masih kosong");
60      }
61  }
62
63  public void print(){
64      if(IsEmpty()){
65          System.out.println(x:"Queue masih kosong");
66      }else{
67          int i = front;
68          while (i != rear){
69              System.out.println(data[i].norek + " " + data[i].nama + " " + data[i].alamat
70              + " " + data[i].umur + " " + data[i].saldo);
71              i = (i + 1) % max;
72          }
73          System.out.println(data[i].norek + " " + data[i].nama + " " + data[i].alamat
74          + " " + data[i].umur + " " + data[i].saldo);
75          System.out.println("Jumlah elemen = " + size);
76      }
77  }
```

buat class baru dengan nama QueueMain. Buat method menu untuk mengakomodasi pilihan menu dari masukan pengguna

NasabahMain21.java > ...

```
1   import java.util.*;
2   public class NasabahMain21 {
3       public static void menu(){
4           System.out.println(x:"Pilih Menu: ");
5           System.out.println(x:"1. Antrian baru");
6           System.out.println(x:"2. Antrian keluar");
7           System.out.println(x:"3. Cek Antrian terdepan");
8           System.out.println(x:"4. Cek Semua Antrian");
9           System.out.println(x:"-------------------------------------");
10      }
```

Buat fungsi main, deklarasikan Scanner dengan nama sc. Buat variabel max untuk menampung kapasitas elemen pada queue. Kemudian lakukan instansiasi objek queue dengan nama antri dan nilai parameternya adalah variabel jumlah.

```
    Run | Debug
12  public static void main(String[] args) {
13      Scanner sc = new Scanner(System.in);
14
15      System.out.print(s:"Masukkan kapasitas queue: ");
16      int jumlah = sc.nextInt();
17
```

Deklarasikan variabel dengan nama pilih bertipe integer untuk menampung pilih menu dari pengguna. Tambahkan kode berikut untuk melakukan perulangan menu sesuai dengan masukan yang diberikan oleh pengguna.

```
19      Nasabah21 antri = new Nasabah21(jumlah);
20      int pilih;
21
22      do {
23          menu();
24          pilih = sc.nextInt();
25          sc.nextLine();
26          switch (pilih) {
27              case 1:
28                  System.out.println(x:"No Rekening: ");
29                  String norek = sc.nextLine();
30                  System.out.println(x:"Nama: ");
31                  String nama = sc.nextLine();
32                  System.out.println(x:"Alamat: ");
33                  String alamat = sc.nextLine();
34                  System.out.println(x:"Umur: ");
35                  int umur = sc.nextInt();
36                  System.out.println(x:"Saldo: ");
37                  double saldo = sc.nextDouble();
38                  Nasabah21 nb = new Nasabah21(norek, nama, alamat, umur, saldo);
39                  sc.nextLine();
40                  antri.Enqueue(nb);
41                  break;
```

```
42              case 2:
43                  Nasabah21 data = antri.Dequeue();
44                  if (!"".equals(data.norek) && !"".equals(data.nama) && !"".equals(data.alamat) && data.umur != 0
45                          && data.saldo != 0) {
46                      System.out.println("Antrian yang keluar: " + data.norek + " " + data.nama + " " + data.alamat
47                              + " " + data.umur + " " + data.saldo);
48                      break;
49                  }
50              case 3:
51                  antri.peek();
52                  break;
53              case 4:
54                  antri.print();
55                  break;
56
57          }
58      } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4);
59  }
60  }
```

Compile dan jalankan class QueueMain, kemudian amati hasilnya

```
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Prak
eptionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\Code\User\workspaceS
hMain21'
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
```

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;
}
```

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!

Jawaban

1. Fungsi dari kondisi if ini adalah untuk memeriksa apakah data yang dikeluarkan dari antrian (data) valid atau tidak sebelum melakukan operasi cetak (print).

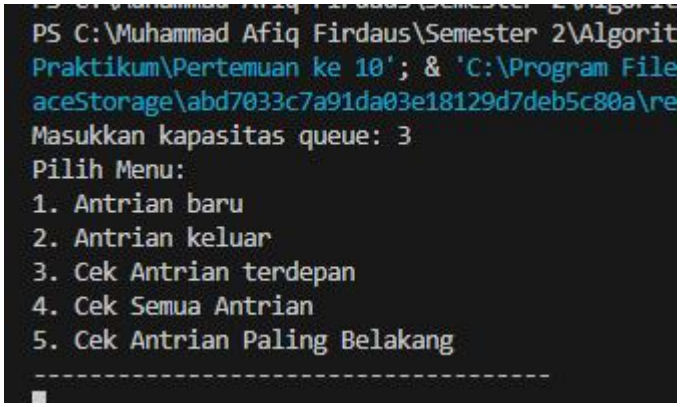
2.

```
128     public void peekRear(){
129         if(!IsEmpty()){
130             System.out.println("Elemen paling belakang: " + data[rear].norek + " " + data[rear].nama + " "
131             + data[rear].alamat + " " + data[rear].umur + " " + data[rear].saldo);
132         } else {
133             System.out.println(x:"Queue masih kosong");
134         }
135     }
136
137 }
```

```
3  public class NasabahMain21 {
4      public static void menu() {
5          System.out.println(x:"Pilih Menu: ");
6          System.out.println(x:"1. Antrian baru");
7          System.out.println(x:"2. Antrian keluar");
8          System.out.println(x:"3. Cek Antrian terdepan");
9          System.out.println(x:"4. Cek Semua Antrian");
10         System.out.println(x:"5. Cek Antrian Paling Belakang");
11         System.out.println(x:"-------------------------------------");
12     }
```

```
                break;
            case 5:
                antri.peekRear();
                break;

            }
        } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
    }
}
```



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

Berikut adalah hasil dari tugas diatas

AntrianPasien21.java > AntrianPasien21

```java
public class AntrianPasien21 {
    private static final int MAX_SIZE = 10;
    private Pasien21[] queueArray;
    private int front;
    private int rear;
    private int size;

    public AntrianPasien21() {
        queueArray = new Pasien21[MAX_SIZE];
        front = 0;
        rear = -1;
        size = 0;
    }

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

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

    public void enqueue(Pasien21 pasien) {
        if (!isFull()) {
            rear = (rear + 1) % MAX_SIZE;
            queueArray[rear] = pasien;
            size++;
        } else {
            System.out.println(x:"Antrian sudah penuh");
        }
    }
```

```java
    }

    public int peekPosition(String nama) {
        for (int i = front, count = 1; count <= size; i = (i + 1) % MAX_SIZE, count++) {
            if (queueArray[i].nama.equals(nama)) {
                return count;
            }
        }
        return -1; // Jika tidak ditemukan
    }

    public void print() {
        if (!isEmpty()) {
            System.out.println(x:"Antrian Pasien:");
            System.out.println(x:"-----------------------------------------------------------------");
            System.out.println(String.format(format:"%-20s %-20s %-10s %-5s", ...args:"Nama", "Nomor Identitas", "Jenis Kelamin", "Umur"));
            System.out.println(x:"-----------------------------------------------------------------");
            for (int i = front, count = 0; count < size; i = (i + 1) % MAX_SIZE, count++) {
                System.out.println(queueArray[i]);
            }
            System.out.println(x:"-----------------------------------------------------------------");
        } else {
            System.out.println(x:"Antrian sudah kosong");
        }
    }
}
```

Pasien21.java > Pasien21

```java
public class Pasien21 {
    String nama;
    String nomorIdentitas;
    String jenisKelamin;
    int umur;

    public Pasien21(String nama, String nomorIdentitas, String jenisKelamin, int umur) {
        this.nama = nama;
        this.nomorIdentitas = nomorIdentitas;
        this.jenisKelamin = jenisKelamin;
        this.umur = umur;
    }

    public String toString() {
        return String.format(format:"%-20s %-20s %-10s %-5d", nama, nomorIdentitas, jenisKelamin, umur);
    }
}
```

```java
PasienMain21.java > PasienMain21 > main(String[])
import java.util.Scanner;
public class PasienMain21 {
    Run | Debug
    public static void main(String[] args) {
        AntrianPasien21 antrian = new AntrianPasien21();
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println(x:"\nMenu:");
            System.out.println(x:"1. Tambah Pasien");
            System.out.println(x:"2. Panggil Pasien");
            System.out.println(x:"3. Lihat Antrian");
            System.out.println(x:"4. Lihat Pasien Paling Depan");
            System.out.println(x:"5. Lihat Pasien Paling Belakang");
            System.out.println(x:"6. Cari Posisi Pasien");
            System.out.println(x:"7. Keluar");
            System.out.print(s:"Pilih: ");
            int choice = scanner.nextInt();
            scanner.nextLine();

```

```java
                case 1:
                    System.out.print(s:"Nama: ");
                    String nama = scanner.nextLine();
                    System.out.print(s:"Nomor Identitas: ");
                    String nomorIdentitas = scanner.nextLine();
                    System.out.print(s:"Jenis Kelamin: ");
                    String jenisKelamin = scanner.nextLine();
                    System.out.print(s:"Umur: ");
                    int umur = scanner.nextInt();
                    scanner.nextLine(); // Consume newline
                    Pasien21 pasien = new Pasien21(nama, nomorIdentitas, jenisKelamin, umur);
                    antrian.enqueue(pasien);
                    System.out.println(x:"Pasien telah ditambahkan ke antrian.");
                    break;
                case 2:
                    Pasien21 panggilPasien = antrian.dequeue();
                    if (panggilPasien != null) {
                        System.out.println("Pasien yang dipanggil: " + panggilPasien.nama);
                    }
                    break;
                case 3:
                    antrian.print();
                    break;
```

```java
                case 4:
                    Pasien21 pasienDepan = antrian.peek();
                    if (pasienDepan != null) {
                        System.out.println("Pasien paling depan: " + pasienDepan.nama);
                    }
                    break;
                case 5:
                    Pasien21 pasienBelakang = antrian.peekRear();
                    if (pasienBelakang != null) {
                        System.out.println("Pasien paling belakang: " + pasienBelakang.nama);
                    }
                    break;
                case 6:
                    System.out.print(s:"Masukkan nama pasien: ");
                    String cariNama = scanner.nextLine();
                    int posisi = antrian.peekPosition(cariNama);
                    if (posisi != -1) {
                        System.out.println("Posisi pasien " + cariNama + " dalam antrian: " + posisi);
                    } else {
                        System.out.println("Pasien " + cariNama + " tidak ditemukan dalam antrian.");
                    }
                    break;
                case 7:
                    System.out.println(x:"Keluar dari program.");
                    System.exit(status:0);
                default:
                    System.out.println(x:"Pilihan tidak valid.");
            }
        }
    }
}
```