Laporan Praktikum Alogaritma dan Struktur Data

Jobsheet 10 : Queue

Alogaritma dan Struktur Data

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

Eka Putri Natalya Kabelen

2341760107

SIB 1E

PROGRAM STUDI SISTEM INFORMASI BISNIS

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2024

8.2 Praktikum 1

Pada percobaan ini, kita akan mengimplementasikan penggunaan class Queue.

8.2.1 Langkah-langkah Percobaan

1. Berdasarkan diagram class tersebut, akan dibuat program class Queue dalam Java dan Tambahkan atribut-atribut Queue sesuai diagram class, kemudian tambahkan pula konstruktornya seperti gambar berikut ini.

```
1  public class Queue11 {
2
3      int[] data;
4      int front;
5      int rear;
6      int size;
7      int max;
8
9      public Queue11(int n) {
10         max = n;
11         data = new int[max];
12         size = 0;
13         front = rear = -1;
14     }
```

2. Buat method IsEmpty bertipe boolean yang digunakan untuk mengecek apakah queue kosong dan Buat method IsFull bertipe boolean yang digunakan untuk mengecek apakah queue sudah penuh.

```
16     public boolean IsEmpty() {
17         if (size == 0) {
18             return true;
19         } else {
20             return false;
21         }
22
23     }
24
25     public boolean IsFull() {
26         if (size == max)  {
27             return true;
28         } else {
29             return false;
30         }
31     }
```

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

```
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      }
```

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

```
41      public void print() {
42          if (IsEmpty()) {
43              System.out.println("Queue masih kosong");
44          } else {
45              int i = front;
46              while (i != rear) {
47                  System.out.print(data[i] + " ");
48                  i = (i + 1) % max;
49              }
50              System.out.println(data[i] + " ");
51              System.out.println("Jumlah elemen = " + size);
52          }
53      }
```

5. Buat method clear bertipe void untuk menghapus semua elemen pada queue

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

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

```
65      public void Enqueue(int dt) {
66          if (IsFull()) {
67              System.out.println("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              }
79              data[rear] = dt;
80              size++;
81          }
82      }
83
```

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

```
84      public int Dequeue() {
85          int dt = 0;
86          if (IsEmpty()) {
87              System.out.println(x:"Queue masih kosong");
88          } else {
89              dt = data[front];
90              size--;
91              if (IsEmpty()) {
92                  front = rear = -1;
93              } else {
94                  if (front == max - 1) {
95                      front = 0;
96                  } else {
97                      front++;
98                  }
99              }
00          }
01          return dt; // Return the dequeued element
02      }
03  }
```

8. Selanjutnya, buat class baru dengan nama QueueMain tetap pada package Praktikum1. Buat method menu bertipe void untuk memilih menu program pada saat dijalankan dan Buat fungsi main, kemudian deklarasikan Scanner dengan nama sc.

```
QueueMain.java > QueueMain > main(String[])
1   import java.util.Scanner;
2   public class QueueMain {
3
4           public static void menu() {
5               System.out.println(x:"Masukkan operasi yang diinginkan:");
6               System.out.println(x:"1. Enqueue");
7               System.out.println(x:"2. Dequeue");
8               System.out.println(x:"3. Print");
9               System.out.println(x:"4. Peek");
10              System.out.println(x:"5. Clear");
11              System.out.println(x:"-----------------");
12
13      }
14
        Run | Debug
15      public static void main(String[] args) {
16          Scanner sc = new Scanner(System.in);
```

9. Buat variabel n untuk menampung masukan berupa jumlah maksimal elemen yang dapat disimpan pada queue.

```
System.out.print(s:"Masukkan kapasitas queue: ");
int n = sc.nextInt();
```

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

```
Queue11 Q = new Queue11(n);
```

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

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

12. Hasil Compile class QueueMain

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

```
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
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
------------------
5
Queue berhasil dikosongkan
```

8.2.3 Pertanyaan

1. Pada konstruktor, mengapa nilai awal atribut front dan rear bernilai -1, sementara atribut size bernilai 0?
   Jawab :
   Karena, diawal front dan rear tidak menunjuk index data tertentu, sehingga jika bernilai 0 maka akan menunjuk index awal sebuah array. Oleh karena itu, diawal front dan rear akan bernilai -1.

2. Pada method Enqueue, jelaskan maksud dan kegunaan dari potongan kode berikut!
   ```
   if (rear == max - 1) {
       rear = 0;
   ```
   Jawab :
   Maksud dari potongan kode diatas adalah untuk mereset nilai rear adalah untuk mengecek apakah queue dalam kondisi kosong. jika queue masih dalam kosong data yang akan masuk menjadi data yang paling depan dan sekaligus menjadi data yang paling akhir dalam queue dan jika queue dalam kondisi tidak kosong , cek apakah posisi rear berada pada index terakhir array. jika benar, maka posisi rear selanjutnya adalah di index 0. jika posisi rear tidak berada pada index terakhir array, maka posisi rear selanjutnya adalah rear + 1.

3. Pada method Dequeue, jelaskan maksud dan kegunaan dari potongan kode berikut!
   ```
   if (front == max - 1) {
       front = 0;
   ```
   Jawab :
   Maksud potongan kode diatas adalah untuk mereset nilai front apabila sudah berada di index terakhir array. Sehingga nilai front akan dibalikkan ke index awal array.

4. Pada method print, mengapa pada proses perulangan variabel i tidak dimulai dari 0 (int i=0), melainkan int i=front?
   Jawab :
   Karena, tidak semua nilai front akan dimulai dari index ke 0.

5. Perhatikan kembali method print, jelaskan maksud dari potongan kode berikut
   ```
   i = (i + 1) % max;
   ```
   Jawab :
   Potongan kode tersebut digunakan untuk memperbarui nilai I sebagai index yang dimana nantinya digunakan sebagai penentu pada index ke berapa informasi akan ditampilkan.

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

```
65      public void Enqueue(int dt) {
66          if (IsFull()) {
67              System.out.println(x:"Queue sudah penuh");
```

7. Pada saat terjadi queue overflow dan queue underflow, program tersebut tetap dapat berjalan dan hanya menampilkan teks informasi. Lakukan modifikasi program sehingga pada saat terjadi queue overflow dan queue underflow, program dihentikan!
   Jawab :

   Hasil modifikasi

```
65      public void Enqueue(int dt) {
66          if (IsFull()) {
67              System.out.println(x:"Queue sudah penuh");
68              System.exit(status:0);
69          } else {
70              if (IsEmpty()) {
71                  front = rear = 0;
72              } else {
73                  if (rear == max - 1) {
74                      rear = 0;
75                  } else {
76                      rear++;
77                  }
78
79              }
80              data[rear] = dt;
81              size++;
82          }
83      }
```

```
85      public int Dequeue() {
86          int dt = 0;
87          if (IsEmpty()) {
88              System.out.println(x:"Queue masih kosong");
89              System.exit(status:0);
90          } else {
91              dt = data[front];
92              size--;
93              if (IsEmpty()) {
94                  front = rear = -1;
95              } else {
96                  if (front == max - 1) {
97                      front = 0;
98                  } else {
99                      front++;
00                  }
01              }
02          }
03          return dt; // Return the dequeued element
04      }
05  }
```

Hasil run

```
Masukkan kapasitas queue: 3
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
------------------
1
Masukkan data baru: 34
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
------------------
1
Masukkan data baru: 21
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
------------------
1
Masukkan data baru: 32
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
------------------
1
Masukkan data baru: 23
Queue sudah penuh
PS C:\Users\ASUS\Documents\SEMESTER 2\A
```

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

```
Masukkan operasi yang diinginkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
------------------
2
Queue masih kosong
PS C:\Users\ASUS\Documents\SEMESTER 2\Alogaritma&StrukturData\JOBSHEET 10>
```

\

8.3 Praktikum 2

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

8.3.1 Langkah-langkah Percobaan

1. Buat program class Nasabah dalam Java, kemudian buat class baru dengan nama Nasabah. Dan tambahkan atribut-atribut Nasabah seperti pada Class Diagram, kemudian tambahkan pula konstruktornya seperti gambar berikut ini.

```
Nasabah11.java > Nasabah11 > Nasabah11()
1  public class Nasabah11 {
2      String norek, nama, alamat;
3      int umur;
4      double saldo;
5
6      Nasabah11() {}
7      Nasabah11(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 }
```

2. 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. 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.

```
1  public class QueueNasabah11 {
2          Nasabah11[] data;
3          int front;
4          int rear;
5          int size;
6          int max;
7
8          public QueueNasabah11(int n) {
9              max = n;
10             data = new Nasabah11[max];
11             size =  0;
12             front  = rear = -1;
13         }
```

```
71          public void Enqueue(Nasabah11 dt) {
72              if(IsFull()) {
73                  System.out.println(x:"Queue sudah penuh");
74              } else {
75                  if (IsEmpty()) {
76                      front = rear = 0;
77                  } else {
78                      if (rear == max - 1) {
79                          rear = 0;
80                      } else {
81                          rear++;
82                      }
83                  }
84                  data[rear] = dt;
85                  size++;
86              }
87          }
88          public Nasabah11 Dequeue() {
89              Nasabah11 dt = new Nasabah11();
90              if(IsEmpty()) {
91                  System.out.println(x:"Queue masih kosong");
92              } else {
93                  dt = data[front];
94                  size--;
95                  if (IsEmpty()) {
96                      front = rear = -1;
97                  } else {
98                      if (front == max - 1) {
99                          front = 0;
100                     } else {
101                         front++;
102                     }
103                 }
104             }
105             return dt;
106         }
```

3. 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.

```
31      public void peek() {
32          if (!IsEmpty()) {
33              System.out.println("Elemen terdepan: " + data[front].norek + " " + data[front].nama + " " + data[front].alamat + " "
34              + data[front].umur + " " + data[front].saldo);
35          } else {
36              System.out.println(x:"Queue masih kosong");
37          }
38      }
```

```
49      public void print() {
50          if (IsEmpty()) {
51              System.out.println(x:"Queue masih kosong");
52          } else {
53              int i = front;
54              while (i != rear) {
55                  System.out.println(data[i].norek + " " + data[i].nama + " " + data[i].alamat + " " + data[i].umur + " " + data[i].saldo);
56                  i = (i + 1) % max;
57              }
58               System.out.println(data[i].norek + " " + data[i].nama + " " + data[i].alamat + " " + data[i].umur + " " + data[i].saldo);
59              System.out.println("Jumlah elemen = " + size);
60          }
61      }
```

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

```
QueueNasabahMain11.java > QueueNasabahMain11 > main(String[])
1   import java.util.Scanner;
2
3   public class QueueNasabahMain11 {
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 keluar");
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      }
```

5. 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.

```
13
        Run | Debug
14      public static void main(String[] args) {
15          Scanner sc = new Scanner(System.in);
16
17          System.out.print(s:"Masukkan kapasitas queue: ");
18          int jumlah = sc.nextInt();
19          QueueNasabah11 antri = new QueueNasabah11(jumlah);
```

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

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

            switch (pilih) {
                case 1:
                    System.out.print("No. Rekening: ");
                    String norek = sc.nextLine();
                    sc.nextLine();
                    System.out.print("Nama: ");
                    String nama = sc.nextLine();
                    System.out.print("Alamat: ");
                    String alamat = sc.nextLine();
                    System.out.print("Umur: ");
                    int umur = sc.nextInt();
                    System.out.print("Saldo: ");
                    double saldo = sc.nextDouble();

                    Nasabah11 nb = new Nasabah11(norek, nama, alamat, umur, saldo);
                    sc.nextLine();
                    antri.Enqueue(nb);
                    break;
                case 2:
                    Nasabah11 data = antri.Dequeue();
                    if (!"".equals(data.norek) && !"".equals(data.nama) && !"".equals(data.alamat) && data.umur != 0 && data.saldo != 0) {
                        System.out.println("Antrian yang keluar: " + data.norek + " " + data.nama + " " + data.alamat + " " + data.umur + " " + data.saldo);
                        break;
                    }
                case 3:
                    antri.peek();
                    break;
                case 4:
                    antri.print();
                    break;
                case 5:
                    antri.peekRear();
                    break;
            }
        } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4);
    }
}
```

8.3.2 Pertanyaan

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

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

Jawab :

Pada potongan kode diatas adalah memastikan bahwa data yang akan dikeluarkan tidak kosong, sehingga data yang ingin keluar itu ada nilainya.

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

Jawab :

Modifikasi

```
40      public void peekRear() {
41          if (!IsEmpty()) {
42              System.out.println("Elemen terbelakang: " + data[rear].norek + " " + data[rear].nama + " " + data[rear].alamat +
43                  " " + data[rear].umur + " " + data[rear].saldo);
44          } else {
45              System.out.println(x:"Queue masih kosong");
46          }
47      }
```

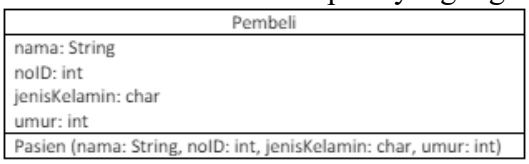
```
56                  case 5:
57                      antri.peekRear();
58                      break;
59              }
```

Hasil Run code program

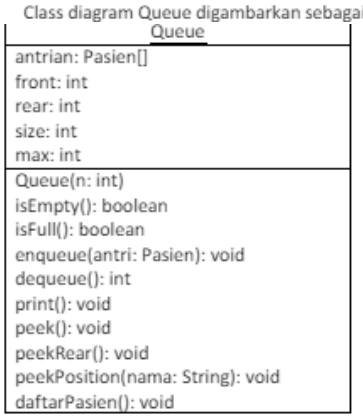
```
Masukkan kapasitas queue: 2
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian keluar
4. Cek semua antrian
5. Cek antrian paling belakang
-------------------------
1
No. Rekening: 12345
Nama: tera
Alamat: kaliam
Umur: 23
Saldo: 23000000
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian keluar
4. Cek semua antrian
5. Cek antrian paling belakang
5. Cek antrian paling belakang
-------------------------
1
No. Rekening: 43113
Nama: herra
Alamat: madura
Umur: 43
Saldo: 132000000
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian keluar
4. Cek semua antrian
5. Cek antrian paling belakang
-------------------------
5
Elemen terbelakang:  herra madura 43 1.32E8
PS D:\SEMESTER2\ASD\PRAKTIKUM\jobsheet 10>
```

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:



Class diagram Queue digambarkan sebagai berikut:



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

Jawab :

Class Pasian

Pasian11.java > Pasian11

```java
public class Pasian11 {
        String nama;
        int noID, umur;
        char jenisKelamin;

        public Pasian11 (String nama, int noID, char jenisKelamin, int umur) {
            this.nama = nama;
            this.noID = noID;
            this.jenisKelamin = jenisKelamin;
            this.umur = umur;
        }
```

Class QueuePasian

```java
public class QueuePasian11 {

        Pasian11[] antrian;
        int front, rear, size, max;

        public QueuePasian11(int n) {
            max = n;
            antrian = new Pasian11[max];
            size = 0;
            front = rear = -1;
        }

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

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

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

        public void enqueue(Pasian11 p) {
            if (isFull()) {
                System.out.println("Antrian sudah penuh");
            } else {
                if (isEmpty()) {
                    front = rear = 0;
                } else {
                    rear = (rear + 1) % max;
                }
                antrian[rear] = p;
                size++;
                System.out.println("Pasien " + p.nama + " berhasil ditambahkan ke dalam antrian");
            }
        }

        public Pasian11 dequeue() {
            Pasian11 p = null;
            if (isEmpty()) {
                System.out.println("Antrian masih kosong");
            } else {
                p = antrian[front];
                if (front == rear) {
                    front = rear = -1;
                } else {
                    front = (front + 1) % max;
                }
                size--;
            }
            return p;
        }

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

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

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

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

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

```

Class QueuePasienMain

```java
import java.util.Scanner;
public class QueuePasienMain11 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Masukkan kapasitas antrian: ");
        int capacity = sc.nextInt();
        QueuePasien11 antrian = new QueuePasien11(capacity);

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

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

                    Pasien11 pb = new Pasien11(nama, noID, jenisKelamin, umur);
                    sc.nextLine();
                    antrian.enqueue(pb);
                    break;
                case 2:
                    Pasien11 pasianHapus = antrian.dequeue();
                    if (pasianHapus != null) {
                        System.out.println("Pasien " + pasianHapus.nama + " berhasil dihapus dari antrian");
                    }
                    break;
                case 3:
                    antrian.peek();
                    break;
                case 4:
                    antrian.peekRear();
                    break;
                case 5:
                    sc.nextLine();
                    System.out.print("Nama Pasien: ");
                    String namaCari = sc.nextLine();
                    antrian.peekPosition(namaCari);
                    break;
                case 6:
                    antrian.daftarPasien();
                    break;
                case 7:
                    System.out.println("Terima kasih, program selesai.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid");
            }
        } while (pilihan != 7);

        sc.close();
    }
}
```

Hasil Run Code Program

```
Masukkan kapasitas antrian: 2
============Pilihan Menu:==============
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=======================================
Pilihan: 1
Nama: putriiiiii
Nomor Identitas: 232434
Jenis Kelamin (L/P): P
Umur: 19
Pasien putriiiiii berhasil ditambahkan ke dalam antrian
============Pilihan Menu:==============
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=======================================
Pilihan: 1
Nama: retaaa
Nomor Identitas: 34242
Jenis Kelamin (L/P): P
Umur: 19
Pasien retaaa berhasil ditambahkan ke dalam antrian
```

```
==========Pilihan Menu:==========
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=====================================
Pilihan: 3
Pasien terdepan: putriiiiii
==========Pilihan Menu:==========
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=====================================
Pilihan: 4
Pasien terbelakang: retaaa
==========Pilihan Menu:==========
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=====================================
Pilihan: 5
Nama Pasien: retaaa
Posisi antrian pasien retaaa : 2
```

```
==========Pilihan Menu:==========
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=======================================
Pilihan: 6
Daftar Pasien:
putriiiiii 232434 P 19
retaaa 34242 P 19
==========Pilihan Menu:==========
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=======================================
Pilihan: 2
Pasien putriiiiii berhasil dihapus dari antrian
==========Pilihan Menu:==========
1. Tambah Pasien
2. Hapus Pasien Terdepan
3. Lihat Pasien Terdepan
4. Lihat Pasien Terbelakang
5. Cari Posisi Pasien
6. Daftar Pasien
7. Keluar
=======================================
Pilihan: 7
Terima kasih, program selesai.
PS C:\Users\ASUS\Documents\SEMESTER 2\Alogaritma&Str
```