

LAPORAN PRAKTIKUM
ALGORITMA & STRUKTUR DATA

Pertemuan – 10: Queue

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



MARGA RETA NOVIA PUTRI

2341760017

D-1V SISTEM INFORMASI BISNIS

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2023/2024

10.2 Praktikum 1

Kode program Queue19.java:

```
1 public class Queue19 {
2     int[] data;
3     int front;
4     int rear;
5     int size;
6     int max;
7
8     public Queue19(int n) {
9         max = n;
10        data = new int[max];
11        size = 0;
12        front = rear = -1;
13    }
14
15    public boolean isEmpty() {
16        if (size == 0) {
17            return true;
18        } else {
19            return false;
20        }
21    }
22
23    public boolean IsFull() {
24        if (size == max) {
25            return true;
26        } else {
27            return false;
28        }
29    }
30
31    public void peek() {
32        if (!isEmpty()) {
33            System.out.println("Elemen terdepan: " + data[front]);
34        } else {
35            System.out.println("Queue masih kosong");
36        }
37    }
38
39    public void print() {
40        if (isEmpty()) {
41            System.out.println("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    }
52
53    public void clear() {
54        if (!isEmpty()) {
55            front = rear = -1;
56            size = 0;
57            System.out.println("Queue berhasil dikosongkan");
58        } else {
59            System.out.println("Queue masih kosong");
60        }
61    }
62
63    public void Enqueue(int dt) {
64        if (IsFull()) {
65            System.out.println("Queue sudah penuh");
66        } else {
67            if (isEmpty()) {
68                front = rear = 0;
69            } else {
70                if (rear == max - 1) {
71                    rear = 0;
72                } else {
73                    rear++;
74                }
75            }
76            data[rear] = dt;
77            size++;
78        }
79    }
80
81    public int Dequeue() {
82        int dt = 0;
83        if (isEmpty()) {
84            System.out.println("Queue masih kosong");
85        } else {
86            dt = data[front];
87            size--;
88            if (isEmpty()) {
89                front = rear = -1;
90            } else {
91                if (front == max - 1) {
92                    front = 0;
93                } else {
94                    front++;
95                }
96            }
97        }
98        return dt;
99    }
100 }
101
```

Kode program QueueMain19.java:

```
1  import java.util.Scanner;
2
3  public class QueueMain19 {
4      public static void menu() {
5          System.out.println("Masukkan operasi yang diinginkan");
6          System.out.println("1. Enqueue");
7          System.out.println("2. Dequeue");
8          System.out.println("3. Print");
9          System.out.println("4. Peek");
10         System.out.println("5. Clear");
11         System.out.println("-----");
12     }
13
14     public static void main(String[] args) {
15         Scanner sc = new Scanner(System.in);
16
17         System.out.print("Masukkan kapasitas queue: ");
18         int n = sc.nextInt();
19         Queue19 Q = new Queue19(n);
20         int pilih;
21
22         do {
23             menu();
24             pilih = sc.nextInt();
25             switch (pilih) {
26                 case 1:
27                     System.out.print("Masukkan data baru: ");
28                     int dataMasuk = sc.nextInt();
29                     Q.Enqueue(dataMasuk);
30                     break;
31                 case 2:
32                     int dataKeluar = Q.Dequeue();
33                     if (dataKeluar != 0) {
34                         System.out.println("Data yang dikeluarkan: " + dataKeluar);
35                         break;
36                     }
37                 case 3:
38                     Q.print();
39                     break;
40                 case 4:
41                     Q.peek();
42                     break;
43                 case 5:
44                     Q.clear();
45                     break;
46             }
47         } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
48     }
49 }
50
```

Hasil run program:

```
Masukkan kapasitas queue: 4
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----
1
Masukkan data baru: 15
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----
1
Masukkan data baru: 31
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----
4
Elemen terdepan: 15
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----
```

10.2.3 Pertanyaan

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

Jawab: Front dan rear bernilai -1 karena tidak menunjuk ke data manapun sedangkan size bernilai 0 karena array masih kosong

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

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

Jawab: Untuk memastikan bahwa jika kita telah mencapai kapasitas maksimum antrian dan ingin menambahkan elemen baru, kita akan kembali ke awal antrian untuk menambahkan elemen tersebut.

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

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

Jawab: Untuk memastikan bahwa jika kita telah mencapai kapasitas maksimum antrian dan ingin menghapus elemen dari antrian, kita akan kembali ke awal antrian untuk mengambil elemen tersebut.

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

Jawab: Karena untuk mencetak elemen-elemen antrian dengan mempertahankan urutan yang benar, dimulai dari elemen yang paling depan.

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

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

Jawab: Digunakan untuk memperbarui nilai i ke elemen berikutnya dalam antrian dengan memperhatikan kemungkinan adanya siklik atau lingkaran, serta memastikan bahwa nilai i tetap dalam kisaran yang valid sesuai dengan kapasitas antrian.

6. Tunjukkan potongan kode program yang merupakan queue overflow!

Jawab:

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

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:

Kode program:

```
63     public void Enqueue(int dt) {
64         if(IsFull()) {
65             System.out.println(x:"Queue sudah penuh");
66             System.exit(status:0);
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     }

81
82     public int Dequeue() {
83         int dt = 0;
84         if(IsEmpty()) {
85             System.out.println(x:"Queue masih kosong");
86             System.exit(status:0);
87         } else {
88             dt = data[front];
89             size--;
90             if (IsEmpty()) {
91                 front = rear = -1;
92             } else {
93                 if (front == max - 1) {
94                     front = 0;
95                 } else {
96                     front++;
97                 }
98             }
99         }
100         return dt;
101     }
102 }
```

Hasil run kode program:

```
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
```

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

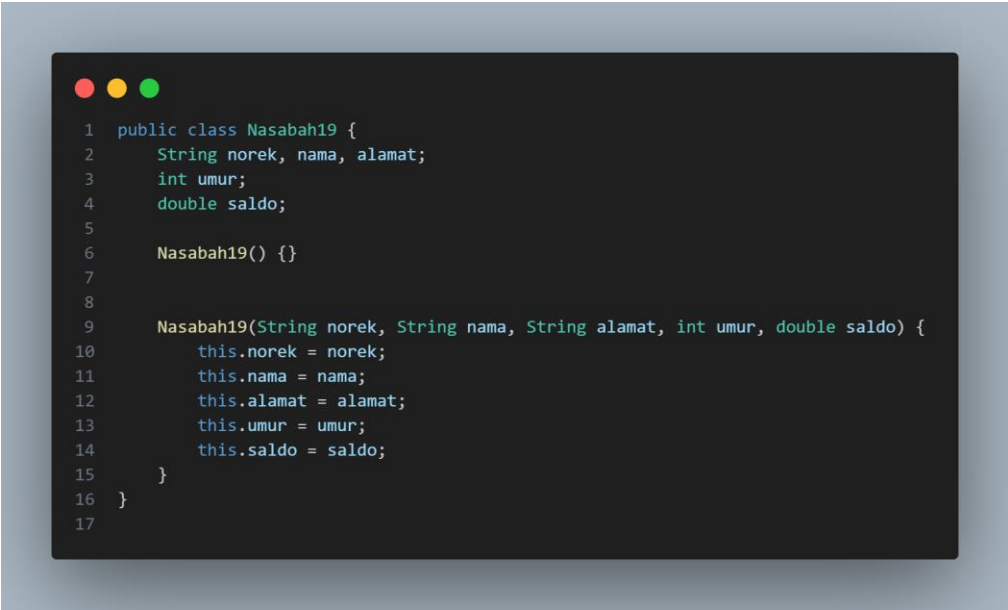
2

Queue masih kosong

PS D:\SEMESTER2\ASD\PRAKTIKUM\jobsheet 10>

10.3 Praktikum 2

Kode program Nasabah19:



```
1 public class Nasabah19 {
2     String norek, nama, alamat;
3     int umur;
4     double saldo;
5
6     Nasabah19() {}
7
8
9     Nasabah19(String norek, String nama, String alamat, int umur, double saldo) {
10         this.norek = norek;
11         this.nama = nama;
12         this.alamat = alamat;
13         this.umur = umur;
14         this.saldo = saldo;
15     }
16 }
17
```

Kode program QueueNasabah19:

```
1 public class QueueNasabah19 {
2     Nasabah19[] data;
3     int front;
4     int rear;
5     int size;
6     int max;
7
8     public QueueNasabah19(int n) {
9         max = n;
10        data = new Nasabah19[max];
11        size = 0;
12        front = rear = -1;
13    }
14
15    public boolean isEmpty() {
16        if (size == 0) {
17            return true;
18        } else {
19            return false;
20        }
21    }
22
23    public boolean IsFull() {
24        if (size == max) {
25            return true;
26        } else {
27            return false;
28        }
29    }
30
31    public void peek() {
32        if (!isEmpty()) {
33            System.out.println("Elemen terdepan: " + data[front].norek + " " + data[front].nama + " " + data[front].alamat + " " + data[front].umur + " " + data[front].saldo);
34        } else {
35            System.out.println("Queue masih kosong");
36        }
37    }
38
39    public void print() {
40        if (!isEmpty()) {
41            System.out.println("Queue masih kosong");
42        } else {
43            int i = front;
44            while (i != rear) {
45                System.out.println(data[i].norek + " " + data[i].nama + " " + data[i].alamat + " " + data[i].umur + " " + data[i].saldo);
46                i = (i + 1) % max;
47            }
48            System.out.println(data[i].norek + " " + data[i].nama + " " + data[i].alamat + " " + data[i].umur + " " + data[i].saldo);
49            System.out.println("Jumlah elemen = " + size);
50        }
51    }
52
53    public void Clear() {
54        if (!isEmpty()) {
55            front = rear = -1;
56            size = 0;
57            System.out.println("Queue berhasil dikosongkan ");
58        } else {
59            System.out.println("Queue masih kosong");
60        }
61    }
62
63    public void Enqueue(Nasabah19 dt) {
64        if (IsFull()) {
65            System.out.println("Queue sudah penuh");
66        } else {
67            if (!isEmpty()) {
68                front = rear = 0;
69            } else {
70                if (rear == max - 1) {
71                    rear = 0;
72                } else {
73                    rear++;
74                }
75            }
76            data[rear] = dt;
77            size++;
78        }
79    }
80
81    public Nasabah19 Dequeue() {
82        Nasabah19 dt = new Nasabah19();
83        if (!isEmpty()) {
84            System.out.println("Queue masih kosong");
85        } else {
86            dt = data[front];
87            size--;
88            if (!isEmpty()) {
89                front = rear = -1;
90            } else {
91                if (front == max - 1) {
92                    front = 0;
93                } else {
94                    front++;
95                }
96            }
97        }
98        return dt;
99    }
100 }
101
```

Kode program QueueNasabahMain:

```
1 import java.util.Scanner;
2
3 public class QueueNasabahMain19 {
4     public static void menu() {
5         System.out.println("Pilih menu: ");
6         System.out.println("1. Antrian baru");
7         System.out.println("2. Antrian keluar");
8         System.out.println("3. Cek antrian keluar");
9         System.out.println("4. Cek semua antrian");
10        System.out.println("-----");
11    }
12
13    public static void main(String[] args) {
14        Scanner sc = new Scanner(System.in);
15
16        System.out.print("Masukkan kapasitas queue: ");
17        int jumlah = sc.nextInt();
18        QueueNasabah19 antri = new QueueNasabah19(jumlah);
19
20        int pilih;
21        do {
22            menu();
23            pilih = sc.nextInt();
24
25            switch (pilih) {
26                case 1:
27                    System.out.print("No. Rekening: ");
28                    String norek = sc.nextLine();
29                    sc.nextLine();
30                    System.out.print("Nama: ");
31                    String nama = sc.nextLine();
32                    System.out.print("Alamat: ");
33                    String alamat = sc.nextLine();
34                    System.out.print("Umur: ");
35                    int umur = sc.nextInt();
36                    System.out.print("Saldo: ");
37                    double saldo = sc.nextDouble();
38
39                    Nasabah19 nb = new Nasabah19(norek, nama, alamat, umur, saldo);
40                    sc.nextLine();
41                    antri.Enqueue(nb);
42                    break;
43                case 2:
44                    Nasabah19 data = antri.Dequeue();
45                    if (!"".equals(data.norek) && !"".equals(data.nama) && !"".equals(data.alamat) && data.umur != 0 && data.saldo != 0) {
46                        System.out.println("Antrian yang keluar: " + data.norek + " " + data.nama + " " + data.alamat + " " + data.umur + " " + data.saldo);
47                        break;
48                    }
49                case 3:
50                    antri.peek();
51                    break;
52                case 4:
53                    antri.print();
54                    break;
55            }
56        } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4);
57    }
58 }
```

10.3.3 Pertanyaan

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

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

Jawab: Untuk mencocokkan jika seluruh data tidak kosong.

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:

Kode program:

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

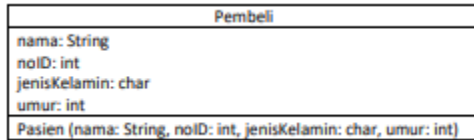
```
56 case 5:
57     antri.peekRear();
58     break;
59 }
```

Hasil run kode program:

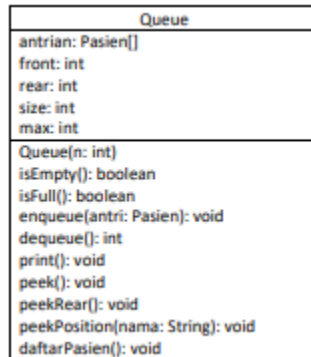
```
Masukkan kapasitas queue: 2
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian keluar
4. Cek semua antrian
5. Cek antrian paling belakang
-----
1
No. Rekening: 12345
Nama: tera
Alamat: kalam
Umur: 23
Saldo: 23000000
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian keluar
4. 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:

Kode program Pasien19.java:

```
1 public class Pasien19 {
2     String nama;
3     int noID, umur;
4     char jenisKelamin;
5
6
7
8     public Pasien19(String nama, int noID, char jenisKelamin, int umur) {
9         this.nama = nama;
10        this.noID = noID;
11        this.jenisKelamin = jenisKelamin;
12        this.umur = umur;
13    }
14 }
15
```

Kode program QueuePasien19.java:

```
1 public class QueuePasien19 {
2     Pasien19[] antrian;
3     int front, rear, size, max;
4
5     public QueuePasien19(int n) {
6         max = n;
7         antrian = new Pasien19[max];
8         size = 0;
9         front = rear = -1;
10    }
11
12    public boolean isEmpty() {
13        return size == 0;
14    }
15
16    public boolean isEmpty() {
17        return size == 0;
18    }
19
20    public boolean isFull() {
21        return size == max;
22    }
23
24    public void enqueue(Pasien19 p) {
25        if (isFull()) {
26            System.out.println("Antrian sudah penuh");
27        } else {
28            if (isEmpty()) {
29                front = rear = 0;
30            } else {
31                rear = (rear + 1) % max;
32            }
33            antrian[rear] = p;
34            size++;
35            System.out.println("Pasien " + p.nama + " berhasil ditambahkan ke dalam antrian");
36        }
37    }
38
39    public Pasien19 dequeue() {
40        Pasien19 p = null;
41        if (isEmpty()) {
42            System.out.println("Antrian masih kosong");
43        } else {
44            p = antrian[front];
45            if (front == rear) {
46                front = rear = -1;
47            } else {
48                front = (front + 1) % max;
49            }
50            size--;
51        }
52        return p;
53    }
54
55    public void print() {
56        if (isEmpty()) {
57            System.out.println("Antrian masih kosong");
58        } else {
59            int i = front;
60            while (i != rear) {
61                System.out.println(antrian[i].nama + " " + antrian[i].noID + " " + antrian[i].jenisKelamin + " " + antrian[i].umur);
62                i = (i + 1) % max;
63            }
64            System.out.println(antrian[i].nama + " " + antrian[i].noID + " " + antrian[i].jenisKelamin + " " + antrian[i].umur);
65        }
66    }
67
68    public void peek() {
69        if (isEmpty()) {
70            System.out.println("Pasien terdepan: " + antrian[front].nama);
71        } else {
72            System.out.println("Antrian masih kosong");
73        }
74    }
75
76    public void peekRear() {
77        if (isEmpty()) {
78            System.out.println("Pasien terbelakang: " + antrian[rear].nama);
79        } else {
80            System.out.println("Antrian masih kosong");
81        }
82    }
83
84    public void peekPosition(String nama) {
85        if (isEmpty()) {
86            for (int i = front; i != rear; i = (i + 1) % max) {
87                if (antrian[i].nama.equals(nama)) {
88                    System.out.println("Posisi antrian pasien " + nama + " : " + ((i - front + max) % max + 1));
89                    return;
90                }
91            }
92            if (antrian[rear].nama.equals(nama)) {
93                System.out.println("Posisi antrian pasien " + nama + " : " + ((rear - front + max) % max + 1));
94            } else {
95                System.out.println("Pasien " + nama + " tidak ditemukan dalam antrian");
96            }
97        } else {
98            System.out.println("Antrian masih kosong");
99        }
100    }
101
102    public void daftarPasien() {
103        if (isEmpty()) {
104            System.out.println("Daftar Pasien:");
105            int i = front;
106            while (i != rear) {
107                System.out.println(antrian[i].nama + " " + antrian[i].noID + " " + antrian[i].jenisKelamin + " " + antrian[i].umur);
108                i = (i + 1) % max;
109            }
110            System.out.println(antrian[i].nama + " " + antrian[i].noID + " " + antrian[i].jenisKelamin + " " + antrian[i].umur);
111        } else {
112            System.out.println("Antrian masih kosong");
113        }
114    }
115 }
116
```

Kode program QueuePasienMain19.java

```
1 import java.util.Scanner;
2
3 public class QueuePasienMain19 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         System.out.print("Masukkan kapasitas antrian: ");
7         int capacity = sc.nextInt();
8         QueuePasien19 antrian = new QueuePasien19(capacity);
9
10        int pilihan;
11        do {
12            System.out.println("=====Pilihan Menu:=====");
13            System.out.println("1. Tambah Pasien");
14            System.out.println("2. Hapus Pasien Terdepan");
15            System.out.println("3. Lihat Pasien Terdepan");
16            System.out.println("4. Lihat Pasien Terbelakang");
17            System.out.println("5. Cari Posisi Pasien");
18            System.out.println("6. Daftar Pasien");
19            System.out.println("7. Keluar");
20            System.out.println("=====");
21            System.out.print("Pilihan: ");
22            pilihan = sc.nextInt();
23
24            switch (pilihan) {
25                case 1:
26                    sc.nextLine();
27                    System.out.print("Nama: ");
28                    String nama = sc.nextLine();
29                    System.out.print("Nomor Identitas: ");
30                    int noID = sc.nextInt();
31                    System.out.print("Jenis Kelamin (L/P): ");
32                    char jenisKelamin = sc.next().charAt(0);
33                    System.out.print("Umur: ");
34                    int umur = sc.nextInt();
35
36                    Pasien19 pb = new Pasien19(nama, noID, jenisKelamin, umur);
37                    sc.nextLine();
38                    antrian.enqueue(pb);
39                    break;
40                case 2:
41                    Pasien19 pasienHapus = antrian.dequeue();
42                    if (pasienHapus != null) {
43                        System.out.println("Pasien " + pasienHapus.nama + " berhasil dihapus dari antrian");
44                    }
45                    break;
46                case 3:
47                    antrian.peek();
48                    break;
49                case 4:
50                    antrian.peekRear();
51                    break;
52                case 5:
53                    sc.nextLine();
54                    System.out.print("Nama Pasien: ");
55                    String namaCari = sc.nextLine();
56                    antrian.peekPosition(namaCari);
57                    break;
58                case 6:
59                    antrian.daftarPasien();
60                    break;
61                case 7:
62                    System.out.println("Terima kasih, program selesai.");
63                    break;
64                default:
65                    System.out.println("Pilihan tidak valid");
66            }
67        } while (pilihan != 7);
68
69        sc.close();
70    }
71 }
72
```

Hasil run 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: Margareta
Nomor Identitas: 123454
Jenis Kelamin (L/P): P
Umur: 19
Pasien Margareta 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: Najua
Nomor Identitas: 99876
Jenis Kelamin (L/P): P
Umur: 19
Pasien Najua 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: 2
Pasien Margareta berhasil dihapus dari antrian
=====Pilihan Menu:=====
```

Link Github: https://github.com/margaretanp/ASD_2024/tree/main/jobsheet%2010i