

LAPORAN PRAKTIKUM
MATA KULIAH PRAKTIKUM ALGORITMA STRUKTUR DATA

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

PERTEMUAN 10 : QUEUE



Nama : Yonanda Mayla Rusdiaty

NIM : 2341760184

Prodi : D-IV Sistem Informasi Bisnis

JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG

2024

PRAKTIKUM 1

Kode program class Queue29 :

```
1 public class Queue29 {
2     int[] data;
3     int front;
4     int rear;
5     int size;
6     int max;
7
8     public Queue29 (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 direset");
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 }
```

Kode program class QueueMain29 :

```
1  import java.util.Scanner;;
2
3  public class QueueMain29 {
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 sc29 = new Scanner(System.in);
16         System.out.print("Masukkan kapasitas queue : ");
17         int n = sc29.nextInt();
18
19         Queue29 Q = new Queue29(n);
20
21         int pilih;
22         do {
23             menu();
24             pilih = sc29.nextInt();
25             switch (pilih) {
26                 case 1:
27                     System.out.print("Masukkan data baru : ");
28                     int dataMasuk = sc29.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                 default:
46                     break;
47             }
48         } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
49         sc29.close();
50     }
51 }
52
```

Output :

```
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 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

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

Pertanyaan :

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

Jawab :

Karena hal tersebut merepresentasikan keadaan awal dari antrian (queue) yang masih kosong. Size = 0 berarti queue tidak berisi elemen apapun / kosong, begitu pula dengan front dan rear = - 1 yang berarti queue kosong, karena indeks array dimulai dari 0

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

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

Jawab :

Enqueue berarti menambahkan data / elemen pada indeks paling belakang dari queue. Kode program tersebut berfungsi untuk mengecek apakah rear (elemen paling belakang) bernilai max - 1 (masih ada space). Jadi, hal tersebut mencegah kondisi ketika rear mencapai batas max array, ketika rear mencapai batas maksimum maka rear akan diatur kembali ke indeks awal array yaitu indeks 0.

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

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

Jawab :

Dequeue berarti mengambil elemen terdepan dari queue. Kode program tersebut berfungsi untuk mengecek apakah front (elemen terdepan) == max - 1 (apakah front sudah mencapai batas maksimum indeks terakhir dari array). Hal tersebut bertujuan untuk memastikan bahwa jika front mencapai batas maksimum, kemudian front akan diatur Kembali ke indeks awal array yaitu indeks 0. Jadi, jika array queue tidak kosong, maka operasi dequeue dapat dilakukan

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

Jawab :

Hal tersebut karena dalam proses print (menampilkan seluruh elemen array queue), i tidak selalu dimulai dari indeks 0, tetapi i dimulai dari indeks front yang bisa berada di Tengah maupun di belakang.

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

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

Jawab :

Kode program tersebut digunakan untuk mengatur peningkatan variabel *i* dalam iterasi *while* pada method *print*. Dalam mencetak nilai *i*, *i* akan diincrement dan akan di mod *max* untuk mengeset agar indeks *i* kembali ke 0 untuk mencegah *i* melebihi *max*, jadi hal mod *max* tersebut bertujuan untuk memastikan bahwa nilai *i* tetap dalam rentang yang valid dalam array.

6. Tunjukkan potongan kode program yang merupakan queue overflow!

Jawab :

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

Pada program tersebut, kondisi *isFull* bernilai *true* dan akan mencetak "Queue sudah penuh", jika ukuran *size* sudah mencapai batas *max* array. Hal tersebut menunjukkan kondisi queue overflow, yaitu mencoba untuk menambahkan elemen ke dalam antrian yang 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 :

Berikut adalah hasil modifikasinya :

➔ Pada class *Queue29* :

```
public void enqueue(int dt) {  
    if (isFull()) {  
        throw new IllegalStateException(s:"Queue sudah penuh");  
    } else {  
        if (isEmpty()) {  
            front = rear = 0;  
        } else {  
            if (rear == max - 1) {  
                rear = 0;  
            } else {  
                rear++;  
            }  
        }  
        data[rear] = dt;  
        size++;  
    }  
}
```

```

public int dequeue() {
    if (isEmpty()) {
        throw new IllegalStateException(s:"Queue masih kosong");
    } else {
        int dt = data[front];
        size--;
        if (isEmpty()) {
            front = rear = -1;
        } else {
            if (front == max - 1) {
                front = 0;
            } else {
                front++;
            }
        }
        return dt;
    }
}

```

➔ Pada class QueueMain29 :

```

switch (pilih) {
    case 1:
        System.out.print(s:"Masukkan data baru : ");
        int dataMasuk = sc29.nextInt();
        try {
            Q.enqueue(dataMasuk);
        } catch (IllegalStateException e) {
            System.out.println(e.getMessage());
            return; // Menghentikan program
        }
        break;
    case 2:
        try {
            int dataKeluar = Q.dequeue();
            System.out.println("Data yang dikeluarkan : " + dataKeluar);
        } catch (IllegalStateException e) {
            System.out.println(e.getMessage());
            return; // Menghentikan program
        }
        break;
}

```


Output :

```
Masukkan kapasitas queue : 2
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
-----
1
Masukkan data baru : 60
Queue sudah penuh
```

PRAKTIKUM 2

Kode program class Nasabah29 :

```
1 public class Nasabah29 {
2     String norek, nama, alamat;
3     int umur;
4     double saldo;
5
6     Nasabah29(String norek, String nama, String alamat, int umur, double saldo) {
7         this.norek = norek;
8         this.nama = nama;
9         this.alamat = alamat;
10        this.umur = umur;
11        this.saldo = saldo;
12    }
13
14    Nasabah29[] data;
15    int front;
16    int rear;
17    int size;
18    int max;
19
20    Nasabah29() {
21    }
22 }
23
24
```

Kode program class Queue29 :

```
1 public class Queue29 {
2     Mahasiswa[] data;
3     int front;
4     int rear;
5     int size;
6     int max;
7
8     public Queue29(int n) {
9         max = n;
10        data = new Mahasiswa[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].nores + " " + data[front].nama + " "
34                + data[front].alamat + " " + data[front].umur + " " + data[front].saldo);
35        } else {
36            System.out.println("Queue masih kosong");
37        }
38    }
39
40    public void print() {
41        if (!isEmpty()) {
42            System.out.println("Queue masih kosong");
43        } else {
44            int i = front;
45            while (i <= rear) {
46                System.out.println(data[i].nores + " " + data[i].nama + " "
47                    + data[i].alamat + " " + data[i].umur + " " + data[i].saldo);
48                i = (i + 1) % max;
49            }
50            System.out.println(data[i].nores + " " + data[i].nama + " "
51                + data[i].alamat + " " + data[i].umur + " " + data[i].saldo);
52            System.out.println("banyak elemen = " + size);
53        }
54    }
55
56    public void clear() {
57        if (!isEmpty()) {
58            front = rear = -1;
59            size = 0;
60            System.out.println("Queue berhasil direset");
61        } else {
62            System.out.println("Queue masih kosong");
63        }
64    }
65
66    public void enqueue(Mahasiswa dt) {
67        if (!isFull()) {
68            System.out.println("Queue sudah penuh");
69        } else {
70            if (!isEmpty()) {
71                front = rear + 1;
72            } else {
73                if (rear == max - 1) {
74                    rear = 0;
75                } else {
76                    rear++;
77                }
78            }
79            data[rear] = dt;
80            size++;
81        }
82    }
83
84    public Mahasiswa dequeue() {
85        Mahasiswa dt = new Mahasiswa20();
86        if (!isEmpty()) {
87            System.out.println("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            }
100        }
101        return dt;
102    }
103
104    public void peekRear() {
105        if (!isEmpty()) {
106            System.out.println("Elemen terbelakang : " + data[rear].nores + " " + data[rear].nama
107                + " " + data[rear].alamat + " " + data[rear].umur + " " + data[rear].saldo);
108        } else {
109            System.out.println("Queue masih kosong");
110        }
111    }
112 }
```

Kode program class QueueMain29 :

```
1  import java.util.Scanner;
2
3  public class QueueMain29 {
4      public static void menu() {
5          System.out.println("-----");
6          System.out.println("Pilih menu : ");
7          System.out.println("1. Antrian baru");
8          System.out.println("2. Antrian keluar");
9          System.out.println("3. Cek antrian terdepan");
10         System.out.println("4. Cek semua antrian");
11         System.out.println("5. Cek antrian belakang");
12         System.out.println("-----");
13     }
14
15     public static void main(String[] args) {
16         Scanner sc29 = new Scanner(System.in);
17
18         System.out.println();
19         System.out.print("Masukkan kapasitas queue : ");
20         int jumlah = sc29.nextInt();
21         Queue29 antri = new Queue29(jumlah);
22
23         int pilih;
24         do {
25             menu();
26             pilih = sc29.nextInt();
27             switch (pilih) {
28                 case 1:
29                     System.out.print("No Rekening : ");
30                     String norek = sc29.next();
31                     System.out.print("Nama : ");
32                     String nama = sc29.next();
33                     System.out.print("Alamat : ");
34                     String alamat = sc29.next();
35                     System.out.print("Umur : ");
36                     int umur = sc29.nextInt();
37                     System.out.print("Saldo : ");
38                     Double saldo = sc29.nextDouble();
39                     Nasabah29 nb = new Nasabah29(norek, nama, alamat, umur, saldo);
40                     antri.enqueue(nb);
41                     break;
42
43                 case 2:
44                     Nasabah29 data = antri.Dequeue();
45                     if (!"".equals(data.norek) && !"".equals(data.nama) && !"".equals(data.alamat)
46                         && !"".equals(data.umur) && !"".equals(data.saldo)) {
47                         System.out.println("Antrian yang keluar: " + data.norek + " " + data.nama + " "
48                             + data.alamat + " " + data.umur + " " + data.saldo);
49                     }
50                     break;
51                 case 3:
52                     antri.print();
53                     break;
54                 case 4:
55                     antri.peek();
56                     break;
57                 case 5:
58                     antri.clear();
59                 default:
60                     break;
61             }
62         } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
63         sc29.close();
64     }
65 }
66
```

Output :

Masukkan kapasitas queue: 4

Pilih menu :

1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
5. Cek antrian belakang

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
5. Cek antrian belakang

1

1

No Rekening : 1200198733

Nama : Dewi

Alamat : Rungkut,Surabaya

Umur : 30

Saldo : 860000

Pilih menu :

1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
5. Cek antrian belakang

4

1200046675 Arif Sukun,Malang 25 1.2E7

1200198733 Dewi Rungkut,Surabaya 30 860000.0

Jumlah elemen = 2

Pilih menu :

1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
5. Cek antrian belakang

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
5. Cek antrian belakang

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
5. Cek antrian belakang

2

Antrian yang keluar: 1200046675 Arif Sukun,Malang 25 1.2E7

Pilih menu :

1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
5. Cek antrian belakang

4

1200198733 Dewi Rungkut,Surabaya 30 860000.0

Jumlah elemen = 1

Pilih menu :

1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
5. Cek antrian belakang

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 :

Kondisi if digunakan untuk memeriksa apakah data yang dikeluarkan dari queue memiliki nilai yang tidak kosong atau tidak. Jika semua kondisi bernilai true, berarti semua nilai (norek, nama, alamat, umur, dan saldo) pada objek tidak kosong. Dengan menggunakan kondisi tersebut, kode tersebut memastikan bahwa hanya ketika data yang dikeluarkan memiliki nilai yang tidak kosong, maka informasi tentang data tersebut akan dicetak. Hal ini bertujuan untuk menghindari pencetakan data yang tidak valid atau kosong dari antrian.

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 :


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

Output :

```
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
5. Cek antrian belakang
-----
1
No Rekening : 123456788
Nama       : Mayla
Alamat    : Malang
Umur      : 19
Saldo     : 10000000
-----
Pilih menu :
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
5. Cek antrian belakang
-----
5
Elemen terbelakang : 123456788 Mayla Malang 19 1.0E7
```

TUGAS

Kode program Pembeli29 :



```
1 public class Pembeli29 {  
2     String nama;  
3     int noID;  
4     char jenisKelamin;  
5     int umur;  
6  
7     Pembeli29 () {  
8  
9     }  
10  
11     Pembeli29 (String nama, int noID, char jenisKelamin, int umur) {  
12         this.nama = nama;  
13         this.noID = noID;  
14         this.jenisKelamin = jenisKelamin;  
15         this.umur = umur;  
16     }  
17 }
```

Kode program Queue29 :

Kode program QueueMain29 :

```
1  import java.util.Scanner;
2
3  public class QueueMain15 {
4      public static void menu() {
5          System.out.println("-----");
6          System.out.println("Pilih menu: ");
7          System.out.println("1. Pasien baru");
8          System.out.println("2. Pasien keluar");
9          System.out.println("3. Daftar Semua Pasien");
10         System.out.println("4. Cek Pasien terdepan");
11         System.out.println("5. Cek Pasien belakang");
12         System.out.println("6. Cek Pasien berdasarkan nama");
13         System.out.println("-----");
14     }
15
16     public static void main(String[] args) {
17         Scanner sc = new Scanner(System.in);
18
19         System.out.println();
20         System.out.print("Masukkan kapasitas queue: ");
21         int jumlah = sc.nextInt();
22
23         Queue29 antri = new Queue29(jumlah);
24         int pilih;
25
26         do {
27             menu();
28             pilih = sc.nextInt();
29             switch (pilih) {
30                 case 1:
31                     System.out.print("Nama          : ");
32                     String nama = sc.next();
33                     System.out.print("No ID          : ");
34                     int noId = sc.nextInt();
35                     System.out.print("Jenis Kelamin (L/P): ");
36                     String jk = sc.next();
37                     System.out.print("Umur           : ");
38                     int umur = sc.nextInt();
39                     Pembeli29 nb = new Pembeli29(nama, noId, jk.charAt(0), umur);
40                     antri.Enqueue(nb);
41                     break;
42                 case 2:
43                     Pembeli29 data = antri.Dequeue();
44                     if (data.nama.isEmpty() || data.noId == 0
45                         || data.umur == 0) {
46                         System.out.println("Pembeli masih kosong");
47                     } else {
48                         System.out.println("Pembeli yang keluar: " + data.nama + " " + data.noId + " "
49                             + data.jenisKelamin + " " + data.umur);
50                     }
51                     break;
52                 case 3:
53                     antri.daftarPasien();
54                     break;
55                 case 4:
56                     antri.peek();
57                     break;
58                 case 5:
59                     antri.peekRear();
60                     break;
61                 case 6:
62                     System.out.print("Masukkan Nama :");
63                     String getNama = sc.next();
64                     antri.peekPosition(getNama);
65                     break;
66             }
67         } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5 || pilih == 6);
68         sc.close();
69     }
70 }
```

Output :

```
Masukkan kapasitas queue: 2
-----
Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Daftar Semua Pasien
4. Cek Pasien terdepan
5. Cek Pasien belakang
6. Cek Pasien berdasarkan nama
-----
1
Nama           : Yonanda
No ID          : 12345
Jenis Kelamin (L/P): P
Umur           : 18
-----
Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Daftar Semua Pasien
4. Cek Pasien terdepan
5. Cek Pasien belakang
6. Cek Pasien berdasarkan nama
-----
1
Nama           : Mayla
No ID          : 12344
Jenis Kelamin (L/P): P
Umur           : 19
-----
```

```
Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Daftar Semua Pasien
4. Cek Pasien terdepan
5. Cek Pasien belakang
6. Cek Pasien berdasarkan nama
-----
2
Pembeli yang keluar: Yonanda 12345 P 18
-----
Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Daftar Semua Pasien
4. Cek Pasien terdepan
5. Cek Pasien belakang
6. Cek Pasien berdasarkan nama
-----
3
Nama: Mayla, No. ID: 12344, JK: P, Umur: 19
-----
Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Daftar Semua Pasien
4. Cek Pasien terdepan
5. Cek Pasien belakang
6. Cek Pasien berdasarkan nama
-----
```