Laporan Praktikum ASD
Pertemuan 10

Nama: Gegas Anugrah Derajat
Kelas: SIB-1F
NIM: 2341760140

Percobaan 8.2

Membuat class queue dan menambahkan konstruktor

```
public class Queue11 {
    int[] data;
    int front, rear, size, max;

    public Queue11(int n) {
        max = n;
        data = new int[max];
        size = 0;
        front = rear = -1;
    }
```

Menambahkan method IsEmpty dan IsFull

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

public boolean IsFull() {
    if (size == max) {
        return true;
    } else {
        return false;
    }
}
```

Membuat method peek dan print

```
public void peek() {
    if (!IsEmpty())  {
        System.out.println("Element terdepan: " + data[front]);
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}

public void print() {
    if (IsEmpty()) {
        System.out.println(x:"Queue masih kosong");
    } else {
        int i = front;
        while (i != rear) {
            System.out.println(data[i] + " ");
            i = (i + 1) % max;
        }
        System.out.println(data[i] + " ");
        System.out.println("Jumlah element = " + size);
    }
}
```

Membuat method clear

```
public void clear() {
    if (!IsEmpty()) {
        front = rear = -1;
        size = 0;
        System.out.println(x:"Queue berhasil dikossongkan");
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}
```

Membuat method Enqueue

```
public void Enqueue(int dt) {
    if (IsFull()) {
        System.out.println(x:"Queue sudah penuh");
    } else {
        if (IsEmpty()) {
            front = rear = 0;
        } else {
            if (rear == max -1) {
                rear = 0;
            } else {
                rear++;
            }
        }
        data[rear] = dt;
        size++;
    }
}
```

Membuat method Dequeue

```
public int Dequeue() {
    int dt = 0;
    if (IsEmpty()) {
        System.out.println(x:"Queue masih kosong");
    } else {
        dt = data[front];
        size--;
        if (IsEmpty()) {
            front = rear = -1;
        } else {
            if (front == max - 1) {
                front = 0;
            } else {
                front++;
            }
        }
    }
    return dt;
}
```

Membuat class QueueMain dan membuat method menu untuk memlilih menu program

```
import java.util.Scanner;

public class QueueMain11 {
    public static void menu() {
        System.out.println(x:"Masukkan operasi yang diinginlkan:");
        System.out.println(x:"1. Enqueue");
        System.out.println(x:"2. Dequeue");
        System.out.println(x:"3. Print");
        System.out.println(x:"4. Peek");
        System.out.println(x:"5. Clear");
        System.out.println(x:"------------------");
    }
```

Membuat fungsi main, menambahkan variabel n untuk menampung jumlah maksimal elemen dan melakukan instansiasi objek Queue

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

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

Melakukan perulangan menggunakan do-while dan di dalam perulangan terdapat pemilihan kondisi menggunakan switch-case

```
int pilih;
do {
    menu();
    pilih = sc.nextInt();
    switch (pilih) {
        case 1:
            System.out.print(s:"Masukkan data baru: ");
            int dataMasuk = sc.nextInt();
            Q.Enqueue(dataMasuk);
            break;
        case 2:
            int dataKeluar = Q.Dequeue();
            if (dataKeluar != 0) {
                System.out.println("Data yang dikeluarkan: " + dataKeluar);
                break;
            }
        case 3:
            Q.print();
            break;
        case 4:
            Q.peek();
            break;
        case 5:
            Q.clear();
            break;
    }
} while (pilih ==  1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
```

Hasil

```
Masukkan kapasitas queue: 6

Masukkan operasi yang diinginlkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
------------------
1
Masukkan data baru: 15

Masukkan operasi yang diinginlkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
------------------
1
Masukkan data baru: 23

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

```
Masukkan operasi yang diinginlkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
------------------
4
Element terdepan: 15

Masukkan operasi yang diinginlkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
------------------
2
Data yang dikeluarkan: 15

Masukkan operasi yang diinginlkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
------------------
3
23
Jumlah element = 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. Front dan rear bernilai -1 karena masih belum mengacu pada indeks berapapun sedangkan size bernilai 0 karena belum terdapat value pada queue.

2. Jika rear berada pada indeks terakhir, maka data baru akan ditambahkan di pada indeks ke-0.

3. Jika front berada pada indeks terakhir, maka data baru akan ditambahkan di pada indeks ke-0.

4. Karena front tidak selalu berada pada indeks ke-0.

5. Jika nilai i bukan rear, maka i akan melakukan increment dan di modulo dengan nilai kapasitas maksimum dari queue.

6. kode program yang merupakan queue overflow adalah

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

7.

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

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

Percobaan 8.3

Membuat class Nasabah dan konstruktornya

```
public class Nasabah11 {
    String norek, nama, alamat;
    int umur;
    double saldo;
    Nasabah11 (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;
    }

    Nasabah11() {

    }

    Nasabah11[] data;
    int front, rear, size, max;

    public Nasabah11(int n) {
        max = n;
        data = new Nasabah11[max];
        size = 0;
        front = rear = -1;
    }
```

Menambahkan method IsEmpty dan IsFull

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

public boolean IsFull() {
    if (size == max) {
        return true;
    } else {
        return false;
    }
}
```

Menambahkan method peek, print dan clear

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

public void print() {
    if (IsEmpty()) {
        System.out.println(x:"Queue masih kosong");
    } else {
        int i = front;
        while (i != rear) {
            System.out.println(data[i].norek + " " + data[i].nama + " " + data[i].alamat + " " + data[i].umur + " "
                + data[i].saldo);
            i = (i + 1) % max;
        }
        System.out.println(data[i].norek + " " + " " + data[i].nama + " " + data[i].alamat + " " + data[i].umur
            + " " + data[i].saldo);
        System.out.println("Jumlah element = " + size);
    }
}

public void clear() {
    if (!IsEmpty()) {
        front = rear = -1;
        size = 0;
        System.out.println(x:"Queue berhasil dikossongkan");
    } else {
        System.out.println(x:"Queue masih kosong");
    }
```

Menambahkan method Enqueue dan Dequeue

```
public void Enqueue(Nasabah11 dt) {
    if (IsFull()) {
        System.out.println(x:"Queue sudah penuh");
    } else {
        if (IsEmpty()) {
            front = rear = 0;
        } else {
            if (rear == max - 1) {
                rear = 0;
            } else {
                rear++;
            }
        }
        data[rear] = dt;
        size++;
    }
}

public Nasabah11 Dequeue() {
    Nasabah11 dt = new Nasabah11();
    if (IsEmpty()) {
        System.out.println(x:"Queue masih kosong");
    } else {
        dt = data[front];
        size--;
        if (IsEmpty()) {
            front = rear = -1;
        } else {
            if (front == max - 1) {
                front = 0;
            } else {
                front++;
            }
        }
    }
    return dt;
}
```

Membuat class NasabahMain dan membuat method menu untuk memlilih menu program

```
import java.util.Scanner;

public class NasabahMain11 {
    public static void menu() {
        System.out.println();
        System.out.println(x:"Pilih menu:");
        System.out.println(x:"1. Antrian baru");
        System.out.println(x:"2. Antrian keluar");
        System.out.println(x:"3. Cek antrian terdepan");
        System.out.println(x:"4. Cek semua antrian");
        System.out.println(x:"----------------------------");
    }
```

Membuat fungsi main, menambahkan variabel jumlah untuk menampung jumlah maksimal elemen dan melakukan instansiasi objek Nasabah

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.println();
    System.out.print(s:"Masukkan kapasitas queue: ");
    int jumlah = sc.nextInt();

    Nasabah11 antrian = new Nasabah11(jumlah);
```

Melakukan perulangan menggunakan do-while dan di dalam perulangan terdapat pemilihan kondisi menggunakan switch-case

```
int pilih;
do {
    menu();
    pilih = sc.nextInt();
    switch (pilih) {
        case 1:
            System.out.print(s:"No Rekening: ");
            String norek = sc.next();
            System.out.print(s:"Nama: ");
            String nama = sc.next();
            System.out.print(s:"Alamat: ");
            String alamat = sc.next();
            System.out.print(s:"Umur: ");
            int umur = sc.nextInt();
            System.out.print(s:"Saldo: ");
            double saldo = sc.nextDouble();
            Nasabah11 nb = new Nasabah11(norek, nama, alamat, umur, saldo);
            sc.nextLine();
            antrian.Enqueue(nb);
            break;
        case 2:
            Nasabah11 data = antrian.Dequeue();
            if (!"".equals(data.norek) && !"".equals(data.nama) && !"".equals(data.alamat)
                && !"".equals(data.umur) && !"".equals(data.saldo)) {
                System.out.println("Antrian yang keluar: " + data.norek + " " + data.nama + " " + data.alamat
                    + " " + data.umur + " " + data.saldo);
                break;
            }
        case 3:
            antrian.peek();
            break;
        case 4:
            antrian.print();
            break;
    }
} while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4);
```

Hasil

```
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 element = 2

Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
--------------------------
3
Element terdepan: 1200046675 Arif Sukun,Malang 25 1.2E7
```

```
Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
--------------------------
2
Antrian yang keluar: 1200046675 Arif Sukun,Malang 25 1.2E7

Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
--------------------------
4
1200198733  Dewi Rungkut,Surabaya 30 8600000.0
Jumlah element = 1
```

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. .equals digunakan untuk membandingkan dua string. Pada kode tersebut dibandingkan apakah list data berisikan string kosong, jika tidak berisi string kosong maka akan print antrian yang keluar.

2.

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

```
public class NasabahMain11 {
    public static void menu() {
        System.out.println();
        System.out.println(x:"Pilih menu:");
        System.out.println(x:"1. Antrian baru");
        System.out.println(x:"2. Antrian keluar");
        System.out.println(x:"3. Cek antrian terdepan");
        System.out.println(x:"4. Cek semua antrian");
        System.out.println(x:"4. Cek antrian terbelakang");
        System.out.println(x:"--------------------------");
    }
```

```
        case 5:
            antrian.peekRear();
            break;
    }
```

```
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
5. Cek antrian terbelakang
--------------------------
1
No Rekening: 1234
Nama: errr
Alamat: sd
Umur: 23
Saldo: 1200000

Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
5. Cek antrian terbelakang
--------------------------
1
No Rekening: 1235
Nama: weee
Alamat: af
Umur: 22
Saldo: 1000000

Pilih menu:
1. Antrian baru
2. Antrian keluar
3. Cek antrian terdepan
4. Cek semua antrian
5. Cek antrian terbelakang
--------------------------
5
Element terbelakang: 1235 weee af 22 1000000.0
```

Tugas

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

Jawaban:

Membuat class pembeli dan konstruktor

```
public class Pembeli11 {
    String nama;
    int noId, umur;
    char jK;

    Pembeli11() {

    }

    Pembeli11(String nama, int noId, int umur, char jK) {
        this.nama = nama;
        this.noId = noId;
        this.umur = umur;
        this.jK = jK;
    }
}
```

Membuat class Queue

```
public class Queue {
    Pembeli11[] data;
    int front, rear, size, max;

    public Queue(int n) {
        max = n;
        data = new Pembeli11[max];
        size = 0;
        front = rear = -1;
    }

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

    public boolean IsFull() {
        if (size == max) {
            return true;
        } else {
            return false;
        }
    }
```

Membuat method peek dan peekRear

```
public void peek() {
    if (!IsEmpty()) {
        System.out.println("Element terdepan: " + data[front].nama + " " + data[front].noId + " "
            + data[front].umur + " " + data[front].jK);
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}

public void peekRear() {
    if (!IsEmpty()) {
        System.out.println("Element terbelakang: " + data[rear].nama + " " + data[rear].noId + " "
            + data[rear].umur + " " + data[rear].jK);
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}
```

Membuat method clear dan print

```
public void print() {
    if (IsEmpty()) {
        System.out.println(x:"Queue masih kosong");
    } else {
        int i = front;
        while (i != rear) {
            System.out.println(data[i].nama + " " + data[i].noId + " " + data[i].umur + " " + data[i].jK);
            i = (i + 1) % max;
        }
        System.out.println(data[i].nama + " " + " " + data[i].noId + " " + data[i].umur + " " + data[i].jK);
        System.out.println("Jumlah element = " + size);
    }
}

public void clear() {
    if (!IsEmpty()) {
        front = rear = -1;
        size = 0;
        System.out.println(x:"Queue berhasil dikossongkan");
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}
```

Membuat method enqueue dan dequeue

```
public void Enqueue(Pembeli11 dt) {
    if (IsFull()) {
        System.out.println(x:"Queue sudah penuh");
    } else {
        if (IsEmpty()) {
            front = rear = 0;
        } else {
            if (rear == max - 1) {
                rear = 0;
            } else {
                rear++;
            }
        }
        data[rear] = dt;
        size++;
    }
}

public Pembeli11 Dequeue() {
    Pembeli11 dt = new Pembeli11();
    if (IsEmpty()) {
        System.out.println(x:"Queue masih kosong");
    } else {
        dt = data[front];
        size--;
        if (IsEmpty()) {
            front = rear = -1;
        } else {
            if (front == max - 1) {
                front = 0;
            } else {
                front++;
            }
        }
    }
    return dt;
}
```

Membuat method peekPosition dan daftarPasien

```
public int peekPosition(String nama) {
    if (IsEmpty()) {
        return -1;
    }
    int i = front;
    int position = 0;
    while (i != rear) {
        if (data[i].nama.equals(nama)) {
            System.out.println("Nama: " + data[i].nama + " No Id: " + data[i].noId + " Umur: " + data[i].umur
                    + " Jenis Kelamin: " + data[i].jK);
            return position;
        }
        position++;
        i = (i + 1) % max;
    }

    if (data[i].nama.equals(nama)) {
        System.out.println("Nama: " + data[i].nama + "No Id: " + data[i].noId + "Umur: " + data[i].umur
                + "Jenis Kelamin: " + data[i].jK);
        return position;
    }
    return -1;
}

public void daftarPasien() {
    if (IsEmpty()) {
        System.out.println(x:"Daftar pasien masih kosong");
    } else {
        int i = front;
        while (i != rear) {
            System.out.println("Nama: " + data[i].nama + "No Id: " + data[i].noId + "Umur: " + data[i].umur
                    + "Jenis Kelamin: " + data[i].jK);

            i = (i + 1) % max;
        }
        System.out.println("Nama: " + data[i].nama + "No Id: " + data[i].noId + "Umur: " + data[i].umur
                + "Jenis Kelamin: " + data[i].jK);
    }
}
```

Membuat class PembeliMain dan method menu

```
import java.util.Scanner;

public class PembeliMain11 {
    public static void menu() {
        System.out.println();
        System.out.println(x:"Pilih menu:");
        System.out.println(x:"1. Pasien baru");
        System.out.println(x:"2. Pasien keluar");
        System.out.println(x:"3. Cek pasien terdepan");
        System.out.println(x:"4. Cek semua pasien");
        System.out.println(x:"5. Cek pasien terbelakang");
        System.out.println(x:"6. Cek pasien berdasarkan nama: ");
        System.out.println(x:"--------------------------");
    }

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

        Queue antrian = new Queue(jumlah);
```

Membuat perulangan menu

```
public static void main(String[] args) {
    int pilih;
    do {
        menu();
        pilih = sc.nextInt();
        switch (pilih) {
            case 1:
                System.out.print(s:"Nama: ");
                String nama = sc.next();
                System.out.print(s:"No Id: ");
                int noId = sc.nextInt();
                System.out.print(s:"Umur: ");
                int umur = sc.nextInt();
                System.out.print(s:"Jenis Kelamin (L/P): ");
                String jK = sc.next();
                Pembeli11 nb = new Pembeli11(nama, noId, umur, jK.charAt(index:0));
                sc.nextLine();
                antrian.Enqueue(nb);
                break;
            case 2:
                Pembeli11 data = antrian.Dequeue();
                if (!"".equals(data.nama) && !"".equals(data.noId) && !"".equals(data.umur)
                    && !"".equals(data.jK)) {
                    System.out.println("Pasien yang keluar: " + data.nama + " " + data.noId + " " + data.umur
                        + " " + data.jK);
                    break;
                }
            case 3:
                antrian.peek();
                break;
            case 4:
                antrian.print();
                break;
            case 5:
                antrian.peekRear();
                break;
            case 6:
                System.out.print(s:"Masukkan nama pasien: ");
                String dptNama = sc.next();
                antrian.peekPosition(dptNama);
                break;
        }
    } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4);
```

Hasil

```
Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Cek pasien terdepan
4. Cek semua pasien
5. Cek pasien terbelakang
6. Cek pasien berdasarkan nama:
---------------------------
1
Nama: lontar
No Id: 1123
Umur: 20
Jenis Kelamin (L/P): L

Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Cek pasien terdepan
4. Cek semua pasien
5. Cek pasien terbelakang
6. Cek pasien berdasarkan nama:
---------------------------
1
Nama: wawu
No Id: 1124
Umur: 23
Jenis Kelamin (L/P): P
```

```
Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Cek pasien terdepan
4. Cek semua pasien
5. Cek pasien terbelakang
6. Cek pasien berdasarkan nama:
---------------------------
1
Nama: goklas
No Id: 1125
Umur: 45
Jenis Kelamin (L/P): L

Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Cek pasien terdepan
4. Cek semua pasien
5. Cek pasien terbelakang
6. Cek pasien berdasarkan nama:
---------------------------
4
lontar 1123 20 L
wawu 1124 23 P
goklas  1125 45 L
Jumlah element = 3
```

```
Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Cek pasien terdepan
4. Cek semua pasien
5. Cek pasien terbelakang
6. Cek pasien berdasarkan nama:
--------------------------
3
Element terdepan: lontar 1123 20 L

Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Cek pasien terdepan
4. Cek semua pasien
5. Cek pasien terbelakang
6. Cek pasien berdasarkan nama:
--------------------------
2
Pasien yang keluar: lontar 1123 20 L
```

```
Pilih menu:
1. Pasien baru
2. Pasien keluar
3. Cek pasien terdepan
4. Cek semua pasien
5. Cek pasien terbelakang
6. Cek pasien berdasarkan nama:
--------------------------
6
Masukkan nama pasien: wawu
Nama: wawu No Id: 1124 Umur: 23 Jenis Kelamin: P
```