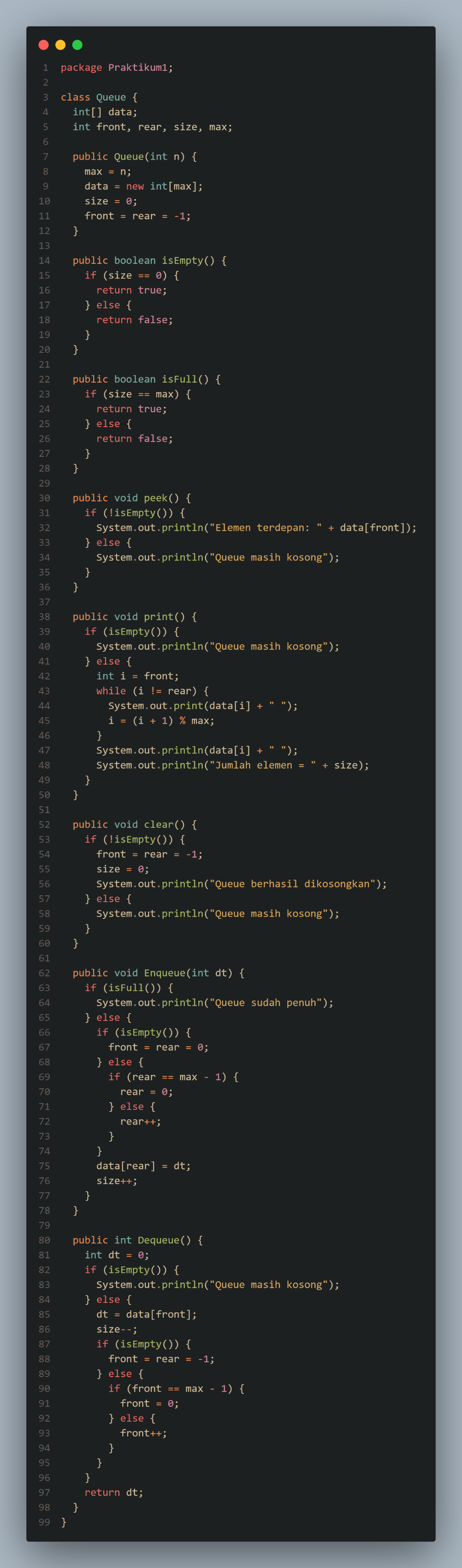
**ALGORITMA STRUKTUR DATA**

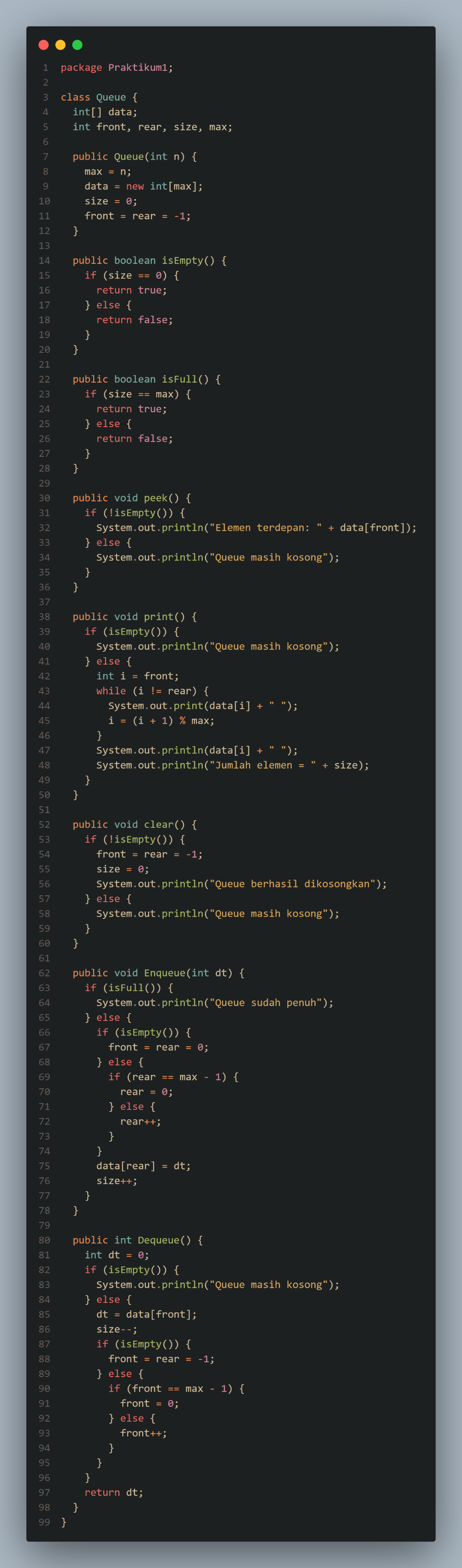
**Praktikum – Queue**

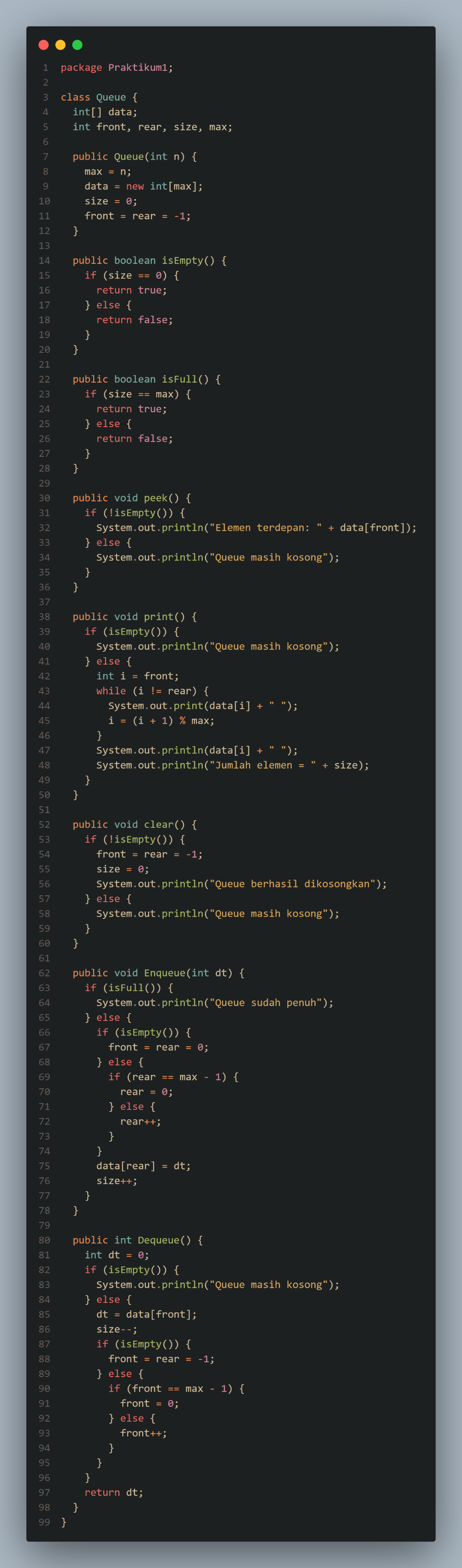
**Lavina 2341760062**

**Praktikum 1**

Queue.java







QueueMain.java





Output :

|  |  |
| --- | --- |
|  |  |
|  |  |

**Pertanyaan**

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

**Jawab :** Atribut size dibuat 0 karena digunakan untuk mengecek jumlah data yang ada didalam queue yang mana jika 0 maka artinya queue kosong, sedangkan pada rear, -1 mewakili index dari data yang ada didalam queue, jika -1 maka queue kosong, jika 0 maka ada 1 data pada index ke-0.

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



**Jawab :**  Maksudnya jika index rear = max - 1 berarti data terakhir(rear) berada diujung queue, jika demikian index rear diubah ke 0 sehingga data yang baru akan masuk ke index 0.

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



**Jawab :** Maksudnya jika index front = max – 1 berarti data front berada diujung queue, jika true maka index front dipindah ke index 0, karena data yang sebelumnya ada difront sudah hilang.

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

**Jawab :** Karena prinsip dari queue adalah FIFO yang mana data mana yang pertama masuk (front) itulah data yang akan keluar(diprint) terlebih dahulu.

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



**Jawab :** Maksud dari kode tersebut adalah untuk memberi nilai variabel i dengan i + 1 yang lalu dimodulo dengan nilai max, tujuannya untuk melakukan loop sampai dimana i = rear, berikut contoh perhitungannya :

**while (i != rear) {**

// Iterasi 1

// 1 != 3

// Iterasi 2

// 2 != 3

// Iterasi 3

// 3 != 3 -> false, loop berhenti

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

// Iterasi 1

// i = (1 + 1) % 6

// i = 2 % 6

// i = 2

// Iterasi 2

// i = (2 + 1) % 6

// i = 3 % 6

// i = 3

**}**

1. Tunjukkan potongan kode program yang merupakan queue overflow!

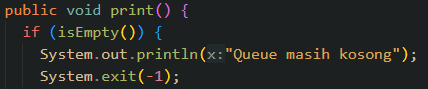
**Jawab :**  Queue overflow ditunjukkan oleh,

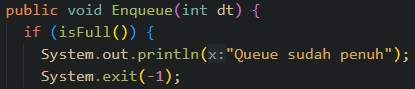
if (isFull()) {

System.out.println("Queue sudah penuh");

}

1. 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!





**Praktikum 2**

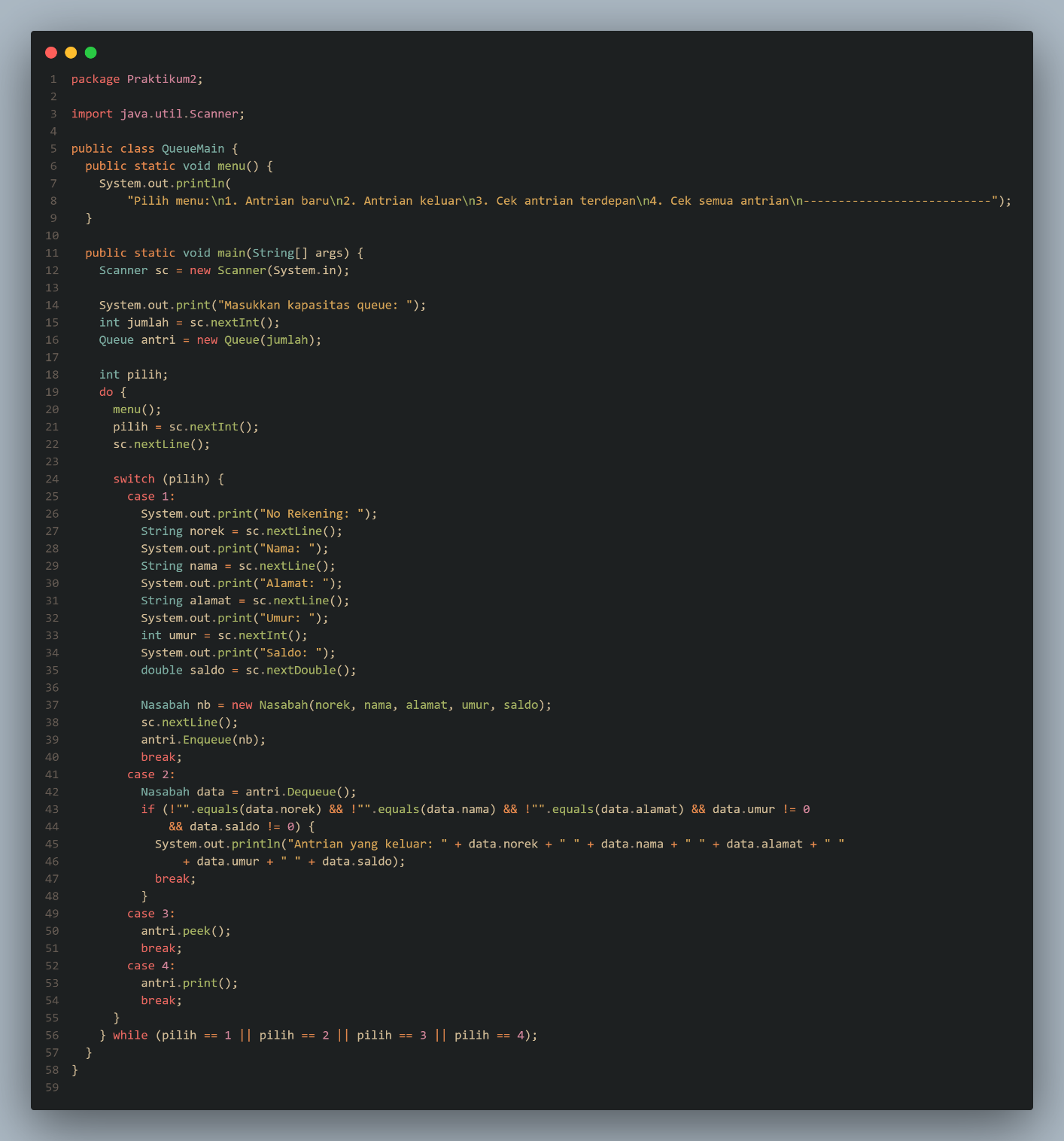
Queue.java





QueueMain.java



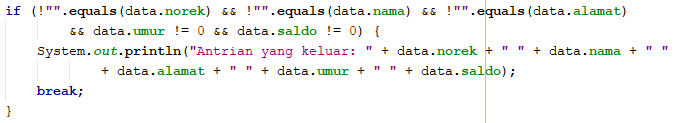


Output :

|  |  |
| --- | --- |
|  |  |

**Pertanyaan**

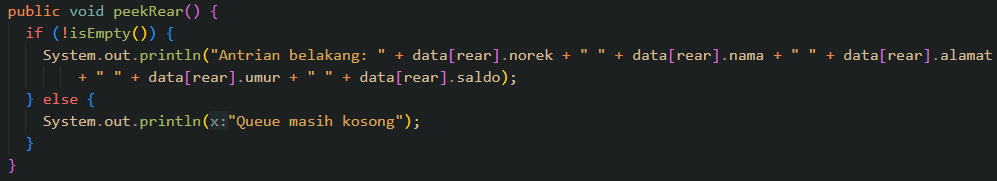
1. Pada class QueueMain, jelaskan fungsi IF pada potongan kode program berikut!



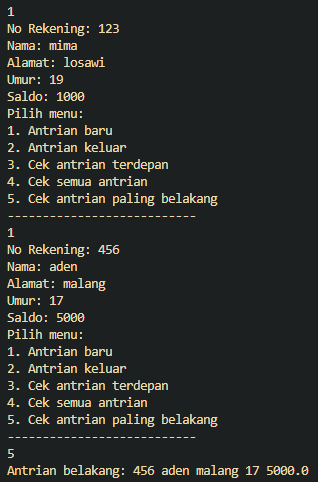
**Jawab :** Pertama kode tersebut mengecek apakah data pada queue kosong dengan menggunakan operator !”” atau tidak sama dengan kosong, jika ture maka data nasabah akan ditampilkan.

1. 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!

Method peekRear() :



Output :



**Tugas**

1. Buatlah program antrian untuk mengilustrasikan antrian pasien di sebuah klinik. Ketika seorang pasien akan mengantri, maka dia harus mendaftarkan nama, nomor identitas, jenis kelamin dan umur seperti yang digambarkan pada Class diagram berikut:

|  |
| --- |
| Pembeli |
| nama: String  noID: int  jenisKelamin: char  umur: int |
| Pasien (nama: String, noID: int, jenisKelamin: char, umur: int) |

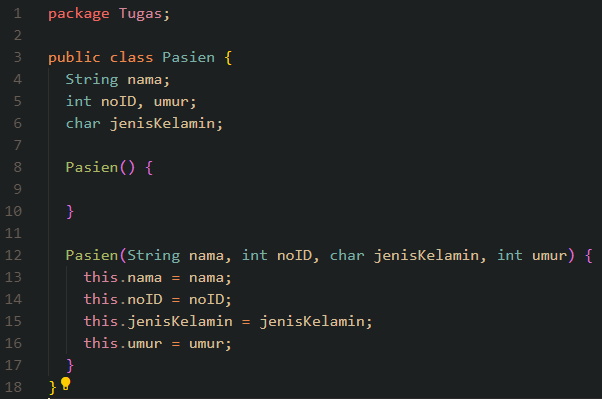
Class diagram Queue digambarkan sebagai berikut:

|  |
| --- |
| Queue |
| antrian: Pasien[]  front: int  rear: int  size: int  max: int |
| Queue(n: int)  isEmpty(): boolean  isFull(): boolean  enqueue(antri: Pasien): void  dequeue(): int  print(): void  peek(): void  peekRear(): void  peekPosition(nama: String): void  daftarPasien(): void |

Keterangan method:

* Method create(), isEmpty(), isFull(), enqueue(), dequeue() dan print(), kegunaannya sama seperti yang telah dibuat pada Praktikum
* Method peek(): digunakan untuk menampilkan data Pasien yang berada di posisi antrian paling depan
* Method peekRear(): digunakan untuk menampilkan data Pasien yang berada di posisi antrian paling belakang
* Method peekPosition(): digunakan untuk menampilkan seorang pasien (berdasarkan nama) posisi antrian ke berapa
* Method daftarPasien(): digunakan untuk menampilkan data seluruh pasien

Pasien.java



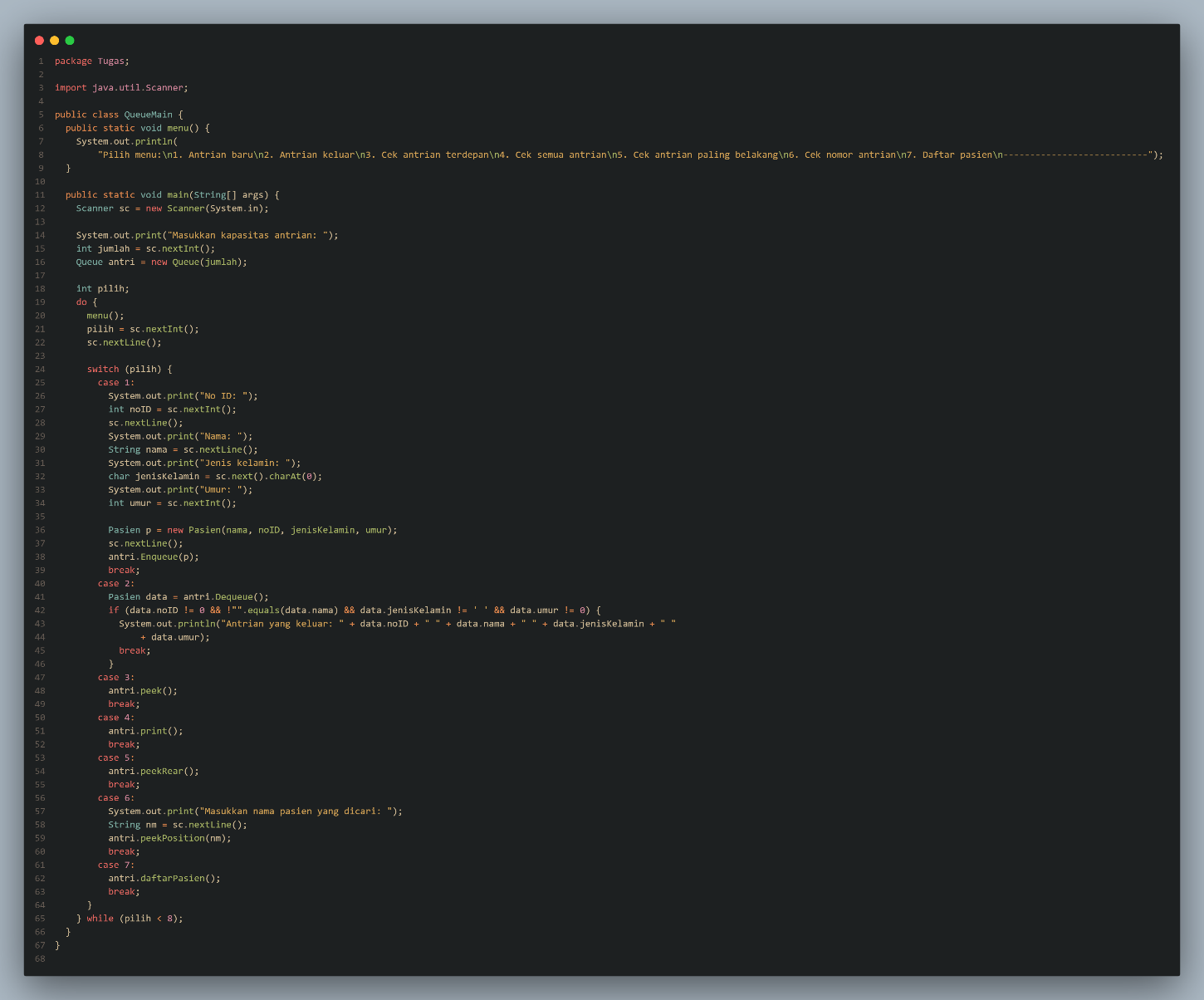
Queue.java







QueueMain.java



Output :

|  |  |
| --- | --- |
|  |  |