# LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

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

# Jobsheet-8 QUEUE



Nama : Surya Rahmat Fatahillah

NIM : 2341760020

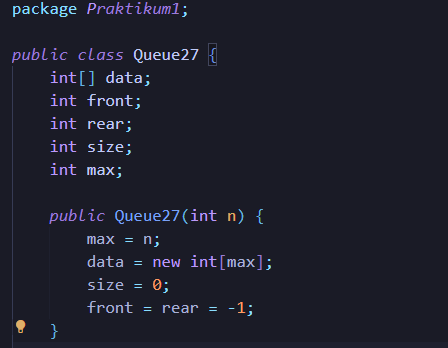
Prodi : Sistem Informasi Bisnis

# JURUSAN TEKNOLOGI INFORMASI POLITEKNIK NEGERI MALANG 2023/2024

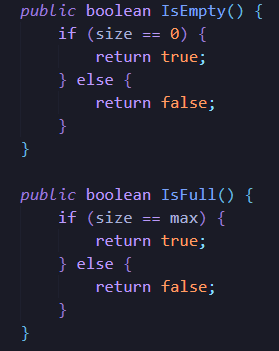
**8.2 Praktikum 1**

**8.2.1 Langkah-langkah Percobaan**

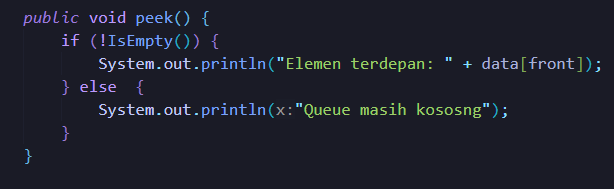
Buat package dengan nama Praktikum1, kemudian buat class baru dengan nama Queue. Tambahkan atribut-atribut Queue sesuai diagram class, kemudian tambahkan pula konstruktornya seperti gambar berikut ini.



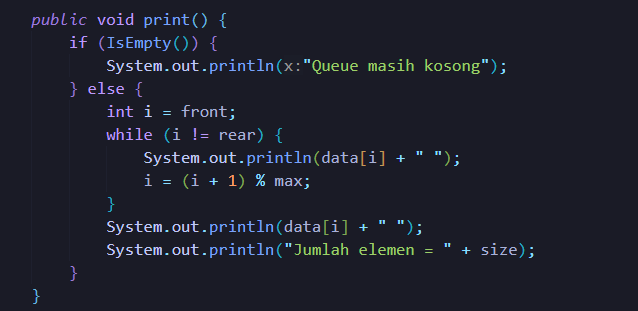
Buat method IsEmpty bertipe boolean yang digunakan untuk mengecek apakah queue kosong dan method IsFull bertipe boolean yang digunakan untuk mengecek apakah queue sudah penuh.



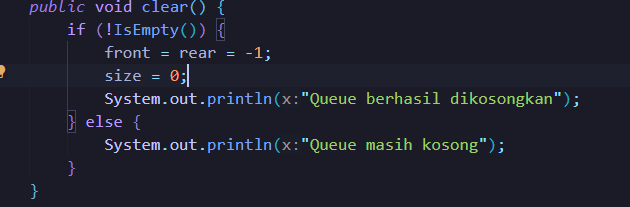
Buat method peek bertipe void untuk menampilkan elemen queue pada posisi paling depan.



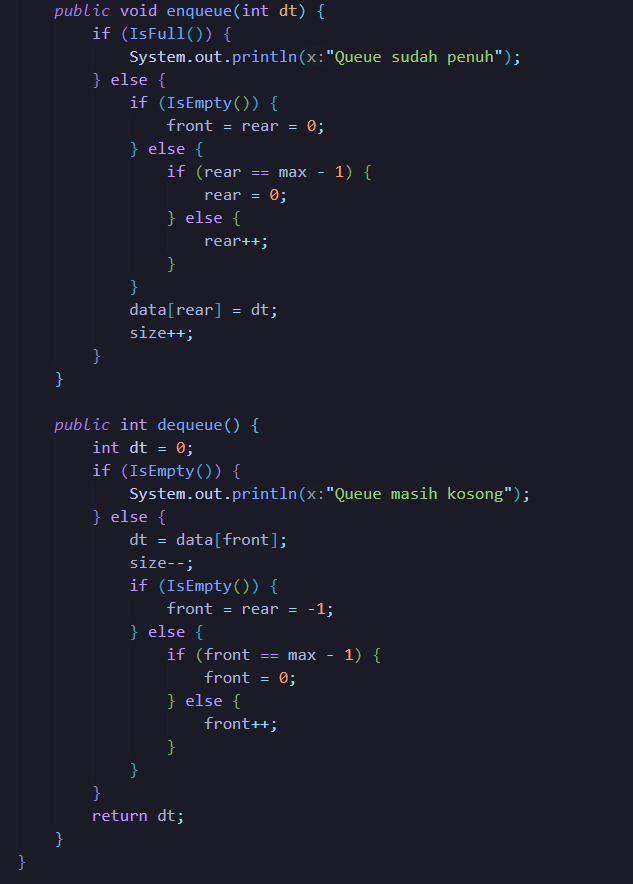
Buat method print bertipe void untuk menampilkan seluruh elemen pada queue mulai dari posisi front sampai dengan posisi rear.



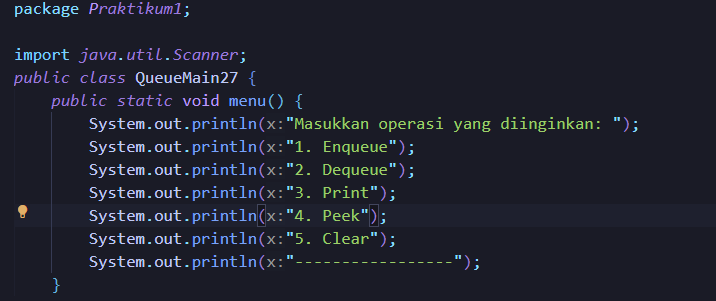
Buat method clear bertipe void untuk menghapus semua elemen pada queue.



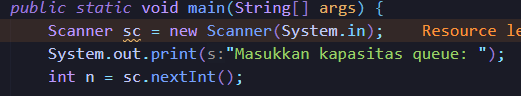
Buat method Enqueue bertipe void untuk menambahkan isi queue dengan parameter dt yang bertipe integer dan method Dequeue bertipe int untuk mengeluarkan data pada queue di posisi paling depan.



Selanjutnya, buat class baru dengan nama QueueMain tetap pada package Praktikum1. Buat method menu bertipe void untuk memilih menu program pada saat dijalankan.



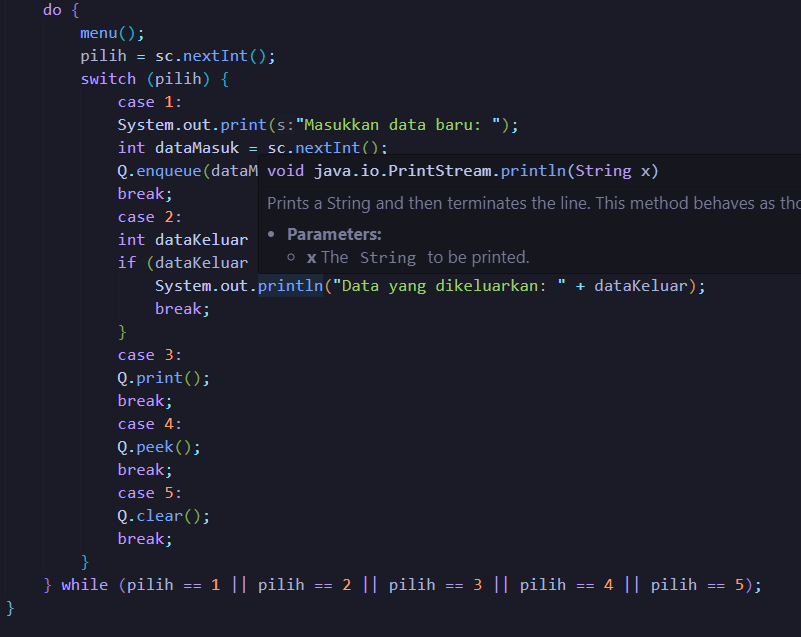
Buat fungsi main, kemudian deklarasikan Scanner dengan nama sc. Kemudian buat variabel n untuk menampung masukan berupa jumlah maksimal elemen yang dapat disimpan pada queue.



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



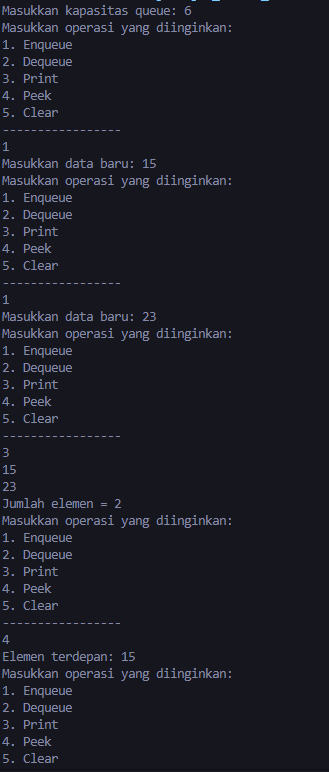
Deklarasikan variabel dengan nama pilih bertipe integer untuk menampung pilih menu dari pengguna. Lakukan perulangan menggunakan do-while untuk menjalankan program secara terus menerus sesuai masukan yang diberikan. Di dalam perulangan tersebut, terdapat pemilihan kondisi menggunakan switch-case untuk menjalankan operasi queue sesuai dengan masukan pengguna.

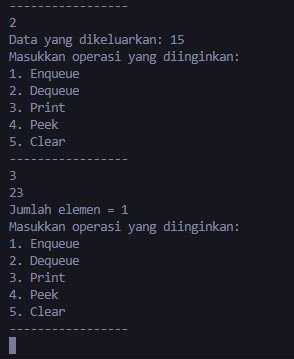


Compile dan jalankan class QueueMain, kemudian amati hasilnya.

**8.2.2 Verifikasi Hasil Percobaan**

Samakan hasil compile kode program Anda dengan gambar berikut ini.

****

****

**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!



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



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!

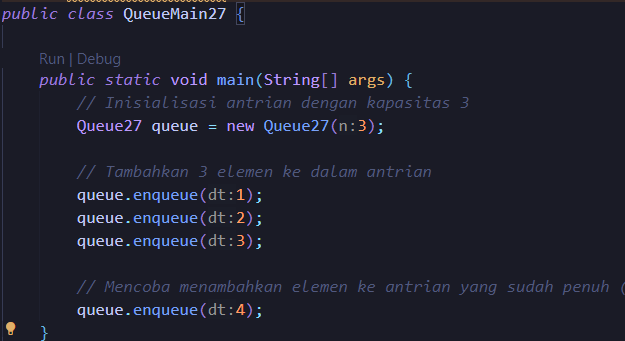


6. Tunjukkan potongan kode program yang merupakan queue overflow!

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!

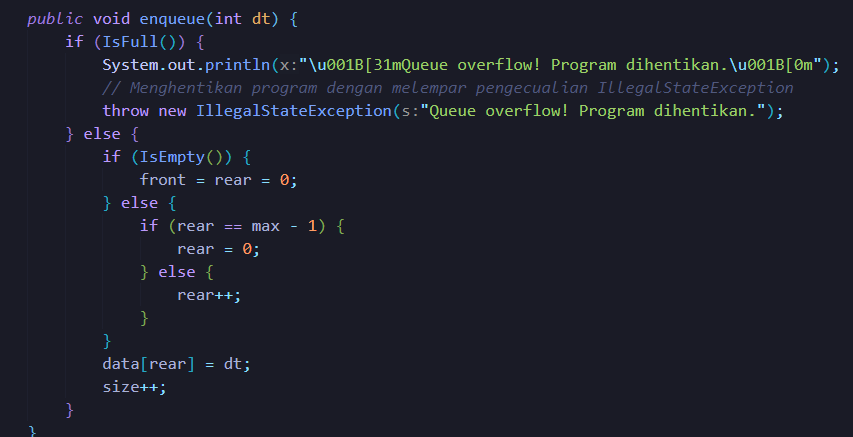
**Jawaban!**

1. Dalam konstruktor kelas **Queue27**, atribut **front** dan **rear** diinisialisasi dengan nilai -1 untuk menunjukkan bahwa pada awalnya antrian kosong. Atribut **size** diinisialisasi dengan nilai 0 karena pada awalnya tidak ada elemen yang ada di dalam antrian.Nilai -1 menunjukkan bahwa tidak ada elemen yang ada di dalam antrian, sedangkan nilai size yang sama dengan 0 mengartikan bahwa tidak ada elemen yang ada di dalam antrian pada awalnya. Ketika elemen pertama dimasukkan ke dalam antrian, **front** dan **rear** akan diperbarui menjadi 0, dan **size** akan bertambah menjadi 1.
2. Potongan kode **if (rear == max - 1) { rear = 0; }** dalam metode **enqueue** bertujuan untuk mengatur ulang posisi **rear** ke indeks awal array jika telah mencapai batas maksimum array (yaitu **max - 1**). Kode **rear == max - 1** memeriksa apakah **rear** sudah mencapai indeks terakhir dari array. Jika sudah, ini menandakan bahwa antrian telah mencapai kapasitas maksimum dan tidak ada lagi ruang untuk menambahkan elemen baru. ika kondisi **rear == max - 1** terpenuhi, artinya **rear** telah mencapai batas maksimum array. Dengan mengatur **rear** kembali ke indeks awal array (**rear = 0**), kita menciptakan efek lingkaran pada array. Dengan kata lain, setelah **rear** mencapai akhir array, langkah selanjutnya adalah memulai lagi dari awal array.
3. Potongan kode **if (front == max - 1) { front = 0; }** dalam metode **dequeue** bertujuan untuk mengatur ulang posisi **front** ke indeks awal array jika telah mencapai batas maksimum array (yaitu **max - 1**). Kode **front == max - 1** memeriksa apakah **front** telah mencapai indeks terakhir dari array. Jika ya, ini menandakan bahwa antrian telah mencapai kapasitas maksimum dan elemen pertama harus dihapus dari antrian. Jika kondisi **front == max - 1** terpenuhi, artinya **front** telah mencapai batas maksimum array. Dengan mengatur **front** kembali ke indeks awal array (**front = 0**).
4. Pada method **print**, variabel **i** tidak dimulai dari 0 (**int i=0**), melainkan diinisialisasi dengan nilai **front**, karena kita ingin mencetak elemen-elemen antrian sesuai dengan urutan sebenarnya dari elemen-elemen tersebut.
5. Potongan kode **i = (i + 1) % max;** dalam method **print** adalah langkah yang digunakan untuk memastikan bahwa perulangan berlangsung secara melingkar (circular) melalui array yang mewakili antrian. Hal ini bertujuan untuk memastikan bahwa semua elemen dalam antrian akan dicetak secara berurutan, bahkan jika **rear** telah melingkari batas akhir array dan berpindah ke awal array lagi. **i + 1** digunakan untuk memindahkan indeks i ke elemen berikutnya dalam array. **% max**: Operasi modulo digunakan di sini untuk memastikan bahwa indeks **i** akan tetap berada dalam rentang yang valid untuk array yang mewakili antrian.
6. Berikut contoh potongan kode program yang merupakan queue overflow

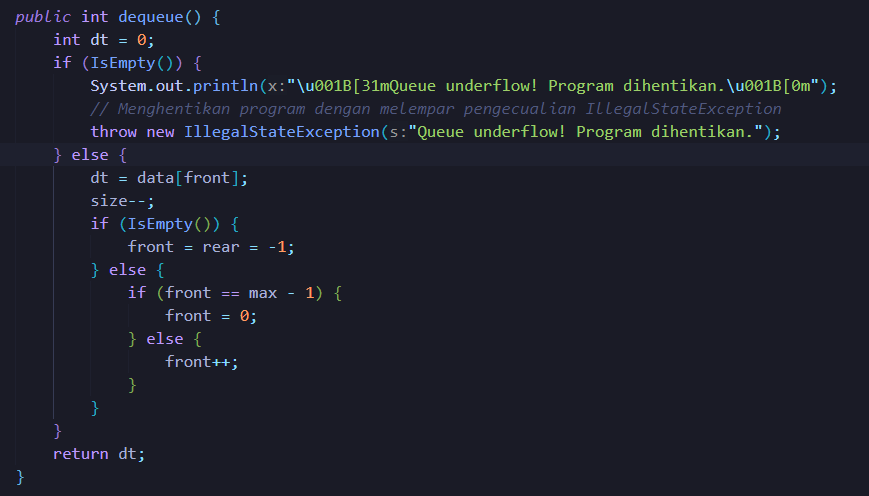


1. Berikut modifikasi program sehingga pada saat terjadi queue overflow dan queue underflow, program dihentikan!

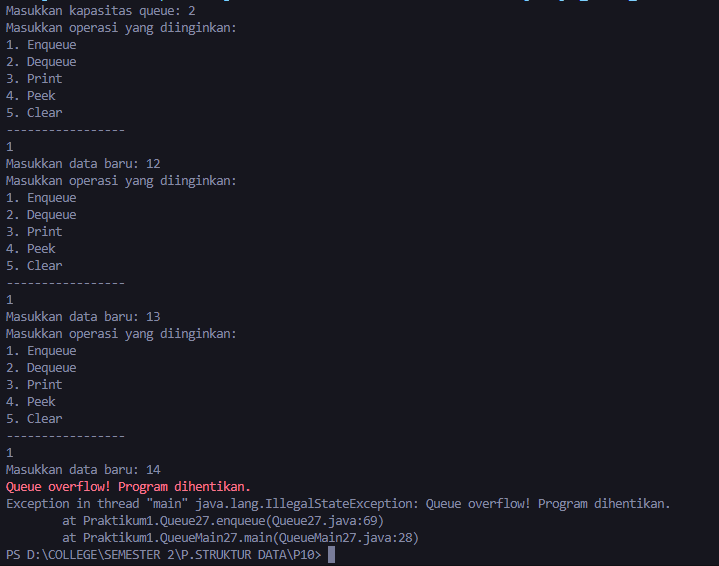
* Overflow

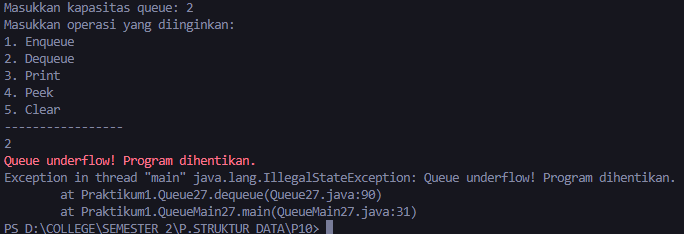


* Underflow



Berikut hasil programnya:





# 8.3 Praktikum 2

# 8.3.1 Langkah-langkah Percobaan

# Buat package dengan nama Praktikum2, kemudian buat class baru dengan nama Nasabah. Tambahkan atribut-atribut Nasabah seperti pada Class Diagram, kemudian tambahkan pula konstruktornya seperti gambar berikut ini.

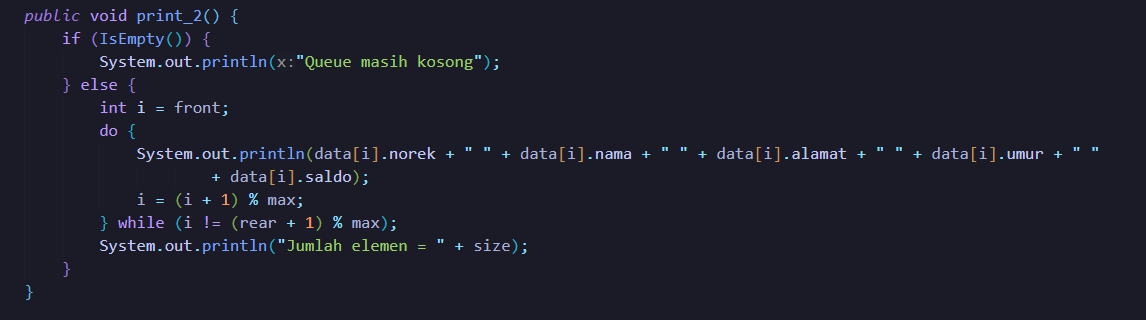
# 

# 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.

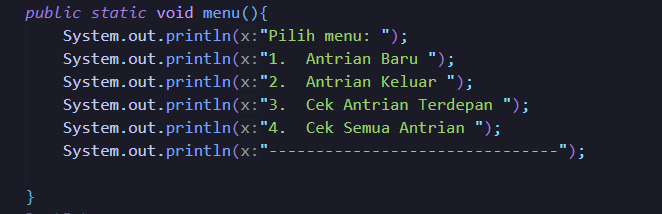
# 

# 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.

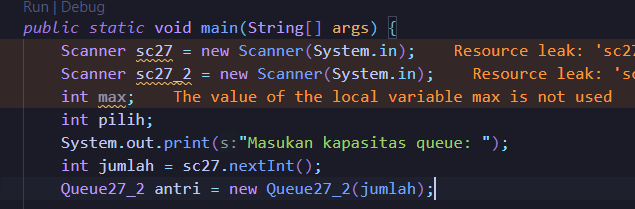
# 



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



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.



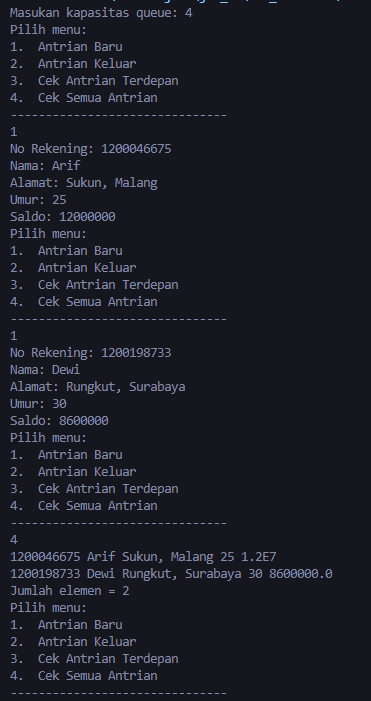
Deklarasikan variabel dengan nama pilih bertipe integer untuk menampung pilih menu dari pengguna. Tambahkan kode berikut untuk melakukan perulangan menu sesuai dengan masukan yang diberikan oleh pengguna

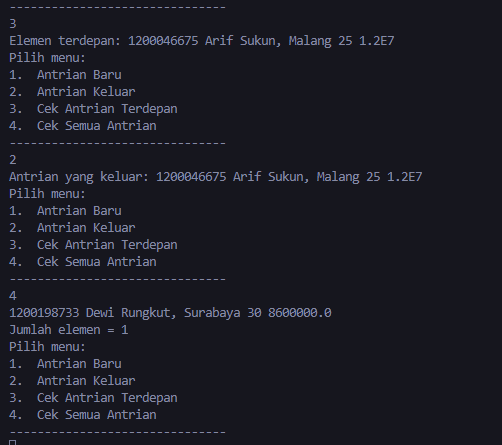


Compile dan jalankan class QueueMain, kemudian amati hasilnya.

**8.3.2 Verifikasi Hasil Percobaan**

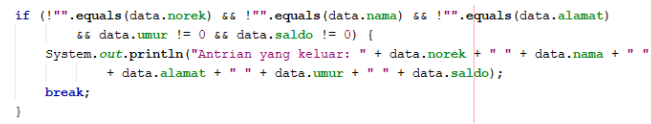
Samakan hasil compile kode program Anda dengan gambar berikut ini





**8.3.3 Pertanyaan**

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



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!

**Jawaban!**

1. Pada potongan kode tersebut, fungsi **if** digunakan untuk memeriksa apakah data yang dikeluarkan dari antrian (**data**) memiliki nilai yang valid sebelum mencetaknya.

**! " ".equals(data.norek)**:digunakan untuk memeriksa apakah nomor rekening (**norek**) dari data antrian tidak sama dengan string kosong. Jika nomor rekening tidak kosong, berarti data tersebut valid.

**! "".equals(data.nama)**: Ini memeriksa apakah nama dari data antrian tidak sama dengan string kosong. Jika nama tidak kosong, berarti data tersebut valid.

**! " ".equals(data.alamat)**: Ini memeriksa apakah alamat dari data antrian tidak sama dengan string kosong. Jika alamat tidak kosong, berarti data tersebut valid.

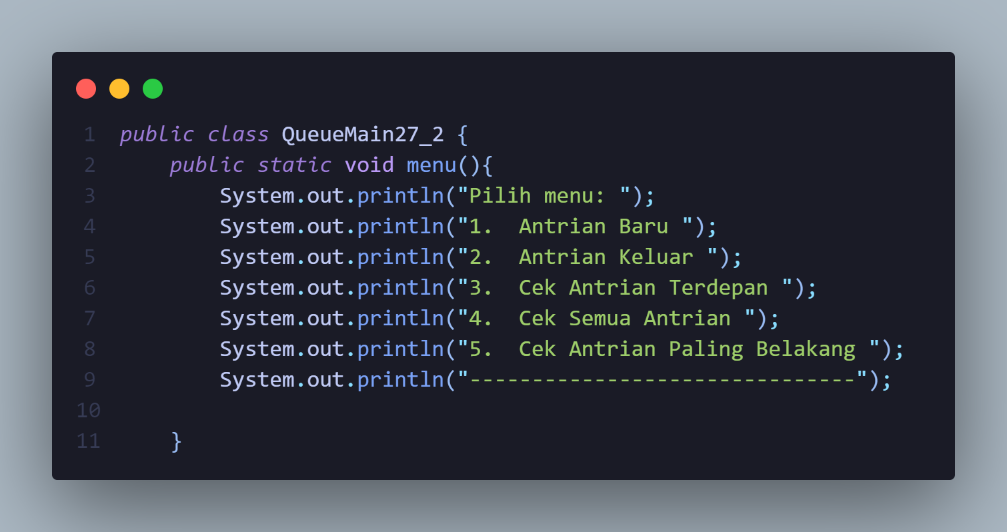
**data.umur != 0**: Ini memeriksa apakah umur dari data antrian tidak sama dengan 0. Umur yang tidak sama dengan 0 menunjukkan bahwa data tersebut valid.

**data.saldo != 0**: Ini memeriksa apakah saldo dari data antrian tidak sama dengan 0. Saldo yang tidak sama dengan 0 menunjukkan bahwa data tersebut valid.

Jadi, jika semua kondisi di atas terpenuhi, maka data dianggap valid dan akan dicetak ke layar dengan informasi nomor rekening, nama, alamat, umur, dan saldo. Jika salah satu kondisi tidak terpenuhi, data tersebut dianggap tidak valid dan tidak akan dicetak.

1. Berikut modifikasi program dengan menambahkan method baru bernama peekRear pada class Queue yang digunakan untuk mengecek antrian yang berada di posisi belakang.

Kemudian menambahkan daftar menu 5. Cek Antrian paling belakang pada class QueueMain sehingga method peekRear dapat dipanggil



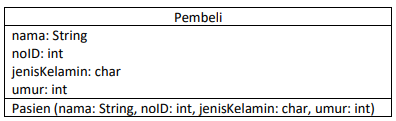


Berikut hasil ketika program dijalankan:

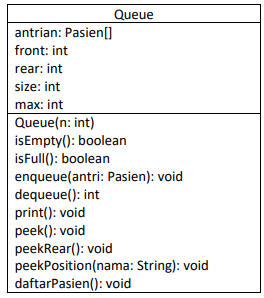


**8.4 Tugas**

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

**Jawaban!**

1. Berikut program antrian untuk mengilustrasikan antrian pasien di sebuah klinik

* Class Pasien27



* Class Queue27





* Class QueueMain27



Berikut hasil dari program ketika dijalankan:



