LAPORAN PRAKTIKUM

QUEUE



Oleh :

MUHAMMAD GALID AVERO

NIM 2311532008

MATA KULIAH STRUKTUR DATA

DOSEN PENGAMPU : DR. WAHYUDI, S.T, M.T

ASISTEN LABORATORIUM : ZAKY ADIL HAKIM

FAKULTAS TEKNOLOGI INFORMASI

DEPARTEMEN INFORMATIKA

UNIVERSITAS ANDALAS

PADANG, MEI 2024

1. PENDAHULUAN

Queue atau dalam bahasa Indonesia yang berarti antrean adalah struktur data yang menyusun elemen-elemen data dalam urutan linier1. Prinsip dasar dari struktur data ini adalah “First In, First Out” (FIFO) yang berarti elemen data yang pertama dimasukkan ke dalam antrean akan menjadi yang pertama pula untuk dikeluarkan.

Caranya bekerja adalah seperti jejeran orang yang sedang menunggu antrean di supermarket di mana orang pertama yang datang adalah yang pertama dilayani (First In, First Out). Pada struktur data ini, urutan pertama (data yang akan dikeluarkan) disebut Front atau Head. Sebaliknya, data pada urutan terakhir (data yang baru saja ditambahkan) disebut Back, Rear, atau Tail.

Operasi yang terdapat pada queue:

1. Menambahkan Elemen (Enqueue):

* add(E e): Menambahkan elemen ke akhir queue.

1. Melempar IllegalStateException jika tidak ada ruang tersedia.

* offer(E e): Menambahkan elemen ke akhir queue. Mengembalikan false jika tidak ada ruang tersedia, tanpa melempar pengecualian.

1. Menghapus Elemen (Dequeue):

* remove(): Menghapus dan mengembalikan elemen pertama dalam queue. Melempar NoSuchElementException jika queue kosong.
* poll(): Menghapus dan mengembalikan elemen pertama dalam queue. Mengembalikan null jika queue kosong.

1. Mengakses Elemen Pertama:

* element(): Mengembalikan elemen pertama tanpa menghapusnya. Melempar NoSuchElementException jika queue kosong.
* peek(): Mengembalikan elemen pertama tanpa menghapusnya. Mengembalikan null jika queue kosong.

Queue memiliki peran yang penting dalam berbagai aplikasi dan algoritma. Salah satu fungsi utamanya adalah mengatur dan mengelola antrean tugas atau operasi secara efisien. Dalam sistem komputasi, ia digunakan untuk menangani tugas-tugas seperti penjadwalan proses, antrean pesan, dan manajemen sumber daya.

Meskipun struktur data queue memiliki banyak kegunaan, kamu juga harus mengetahui beberapa keuntungan dan keterbatasan yang perlu diperhatikan sebelum menggunakannya:

Keuntungan

1. Data berjumlah besar dapat dikelola dengan mudah dan efisien.
2. Proses insert dan delete data dapat dilakukan dengan mudah karena mengikuti aturan first in first out (FIFO).
3. Efisien dalam menangani tugas berdasarkan urutan kedatangan.

Keterbatasan

1. Tidak efisien untuk pencarian elemen tertentu dalam antrean.
2. Memerlukan alokasi memori yang cukup untuk menyimpan antrean.

Queue dapat diimplementasikan menggunakan berbagai cara, di antaranya adalah:

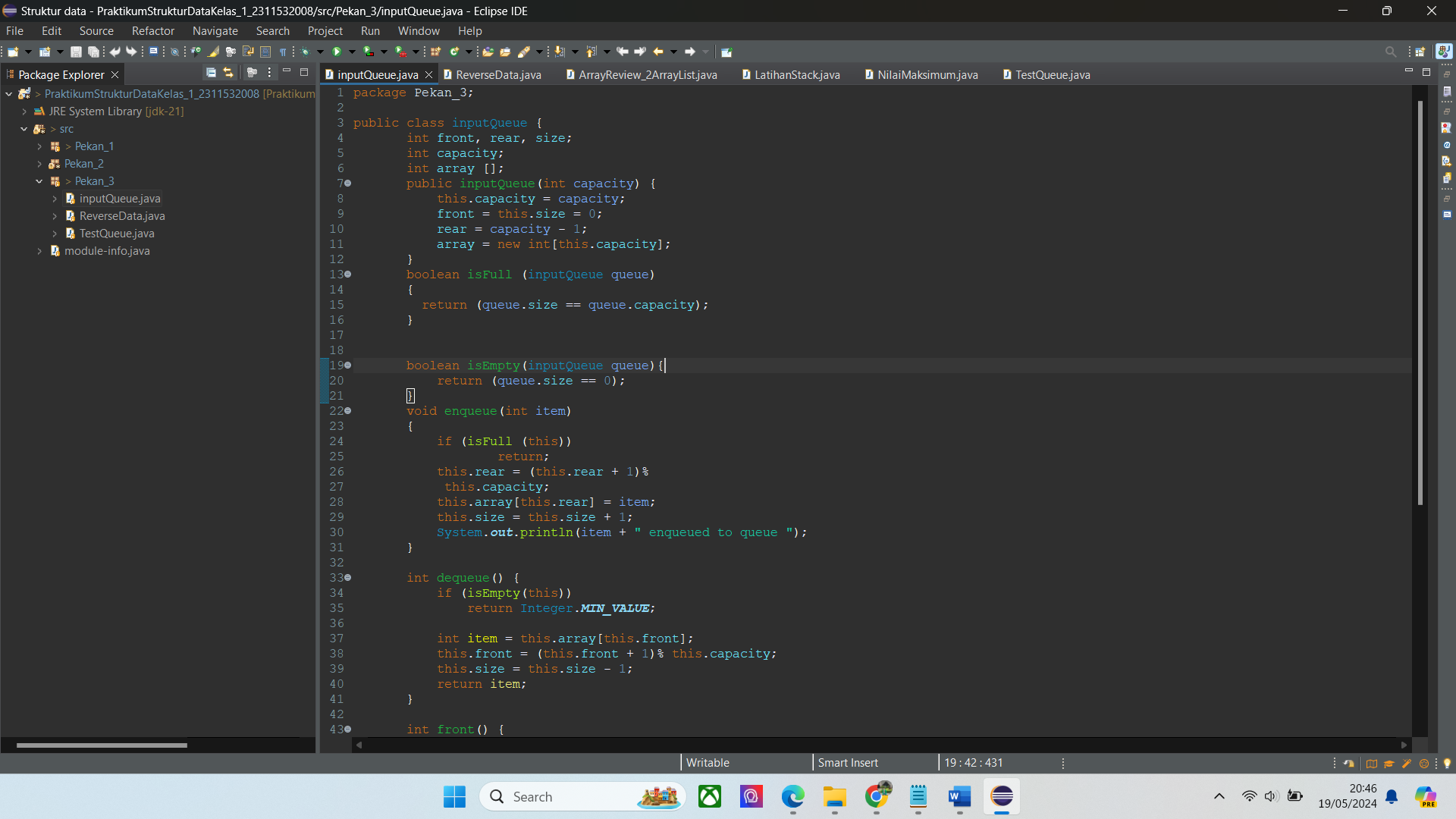
1. Array: Implementasi queue dengan array memerlukan ukuran tetap yang ditentukan sebelumnya. Implementasi ini dapat memanfaatkan dua indeks untuk melacak posisi depan (front) dan belakang (rear) dari queue.
2. Linked List: Implementasi queue dengan linked list lebih fleksibel karena tidak memerlukan ukuran tetap dan dapat dengan mudah menyesuaikan dengan jumlah elemen yang berubah-ubah.
3. TUJUAN PRAKTIKUM

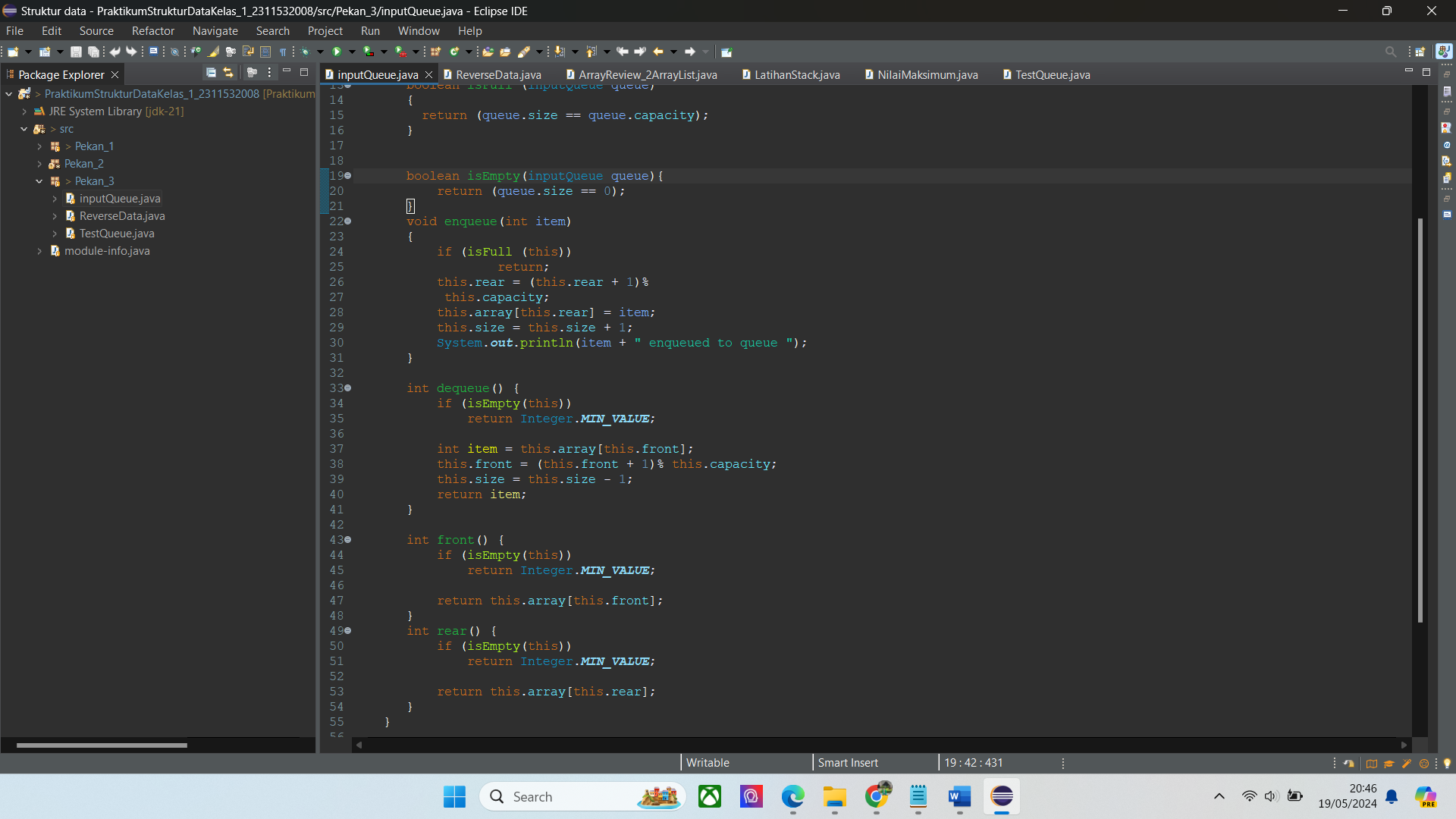
1. Membuat program queue.

2. Membuat program input untuk queue yang dibuat.

3. Membuat program reverse data pada queue.

1. LANGKAH PRAKTIKUM
2. Membuat program Queue menggunakan Class





Pada program diatas adalah program untuk mendefenisikan metod “inputQueue” yang mana method tersebut akan digunakan pada suatu program di class yang lain. Untuk penjelasannya sebagai berikut :

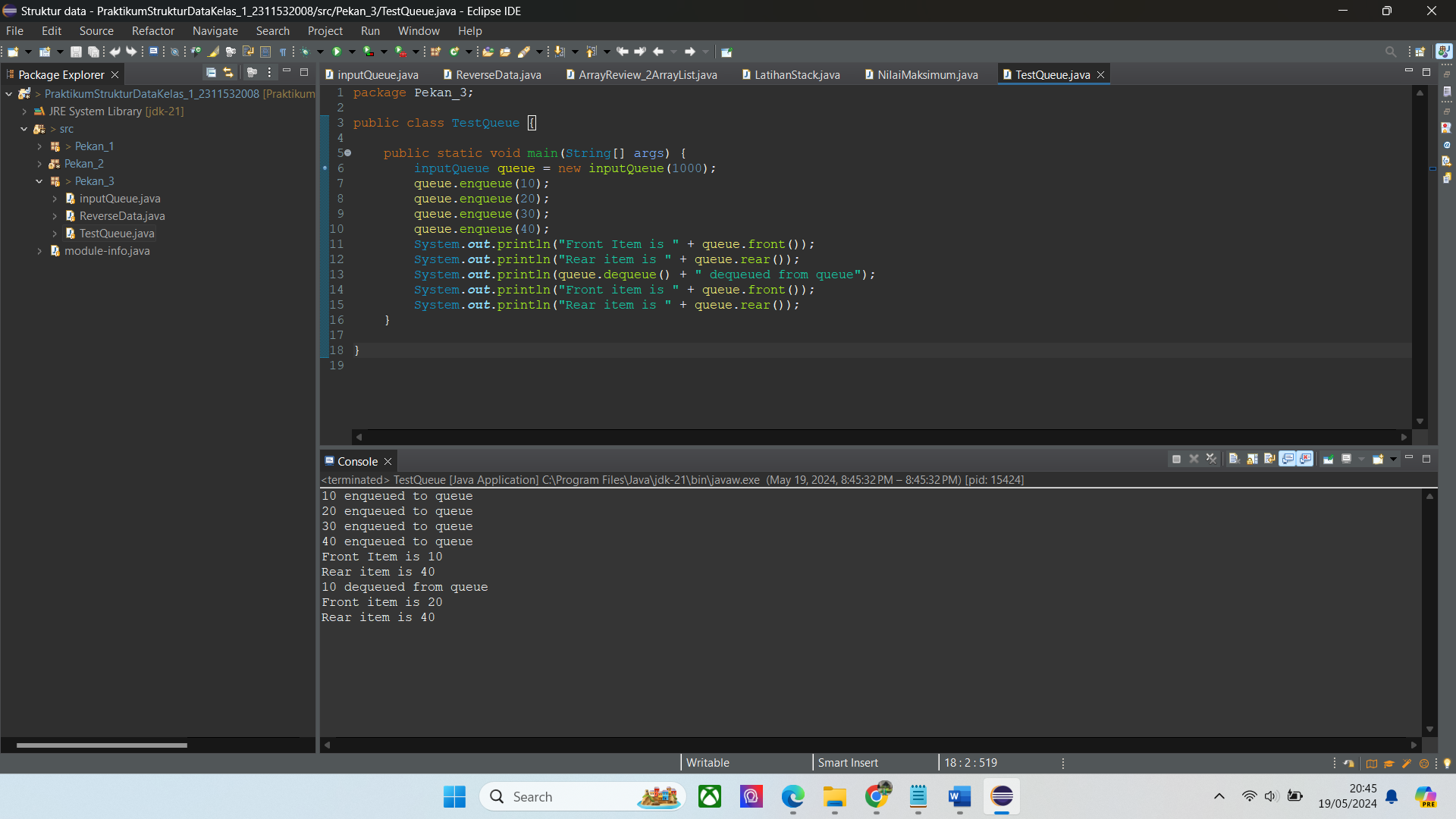
Kelas inputQueue merepresentasikan antrian dengan metode queue circular.

Konstruktor inputQueue(int capacity), Konstruktor ini menginisialisasi antrian dengan kapasitas yang diberikan. Ia menetapkan variabel front dan size menjadi 0, dan variabel rear menjadi capacity - 1. Ia juga membuat array dengan kapasitas yang diberikan.

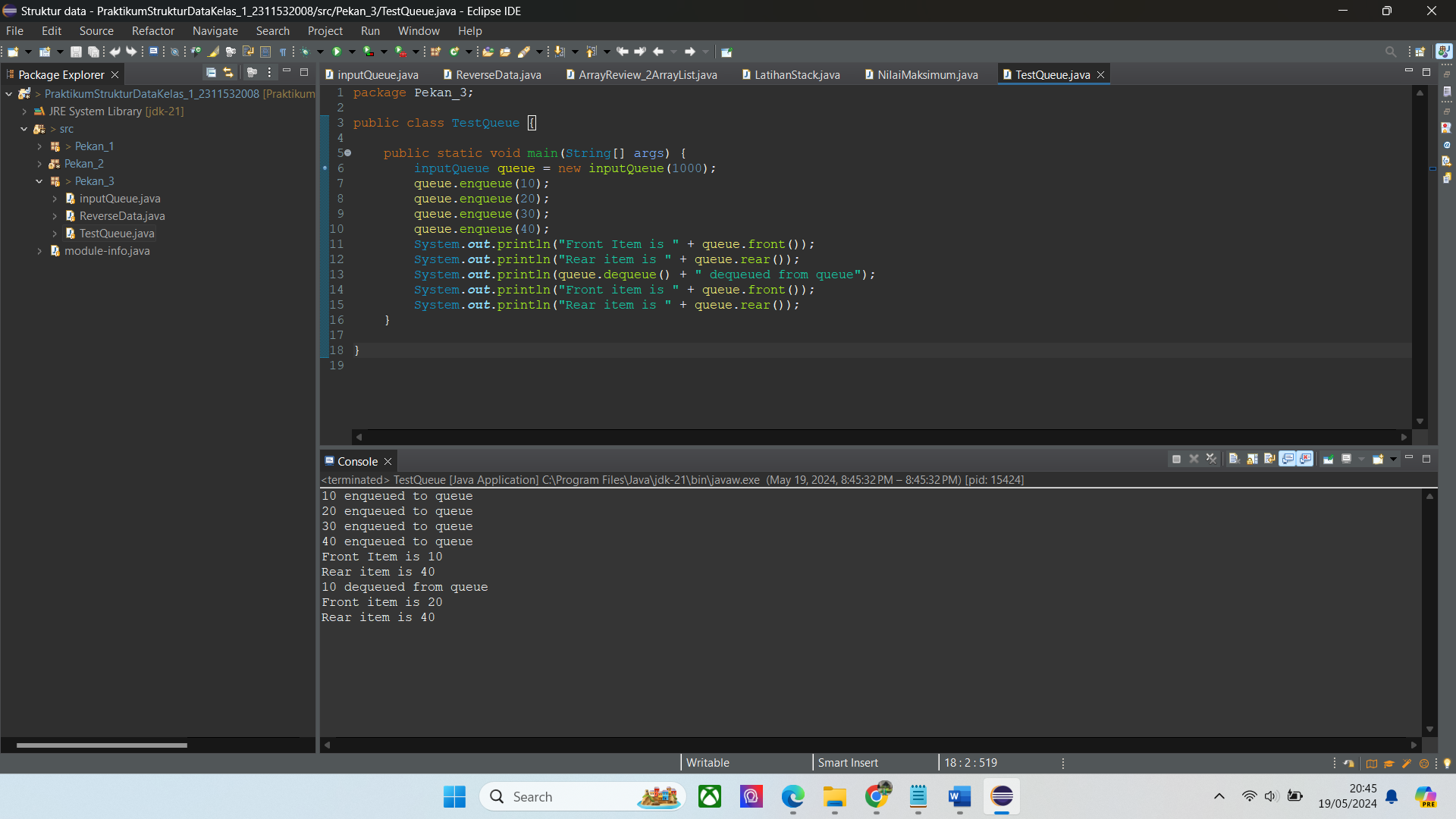
Kelas ini memiliki beberapa metode:

1. isFull(inputQueue queue): Metode ini mengembalikan nilai true jika antrian penuh, yaitu jika variabel size sama dengan capacity.
2. isEmpty(inputQueue queue): Metode ini mengembalikan nilai true jika antrian kosong, yaitu jika variabel size nol.
3. enqueue(int item): Metode ini menambahkan item ke akhir antrian. Jika antrian penuh, maka tidak melakukan apa-apa. Selain itu, ia meningkatkan variabel rear, mengembalikannya ke awal array jika diperlukan, dan menambahkan item ke array. Ia juga meningkatkan variabel size.
4. dequeue(): Metode ini menghapus item dari depan antrian. Jika antrian kosong, maka ia mengembalikan Integer.MIN\_VALUE. Selain itu, ia mengembalikan item di depan antrian, meningkatkan variabel front, dan mengurangi variabel size.
5. front(): Metode ini mengembalikan item di depan antrian. Jika antrian kosong, ia mengembalikan Integer.MIN\_VALUE.
6. rear(): Metode ini mengembalikan item di belakang antrian. Jika antrian kosong, ia mengembalikan Integer.MIN\_VALUE.

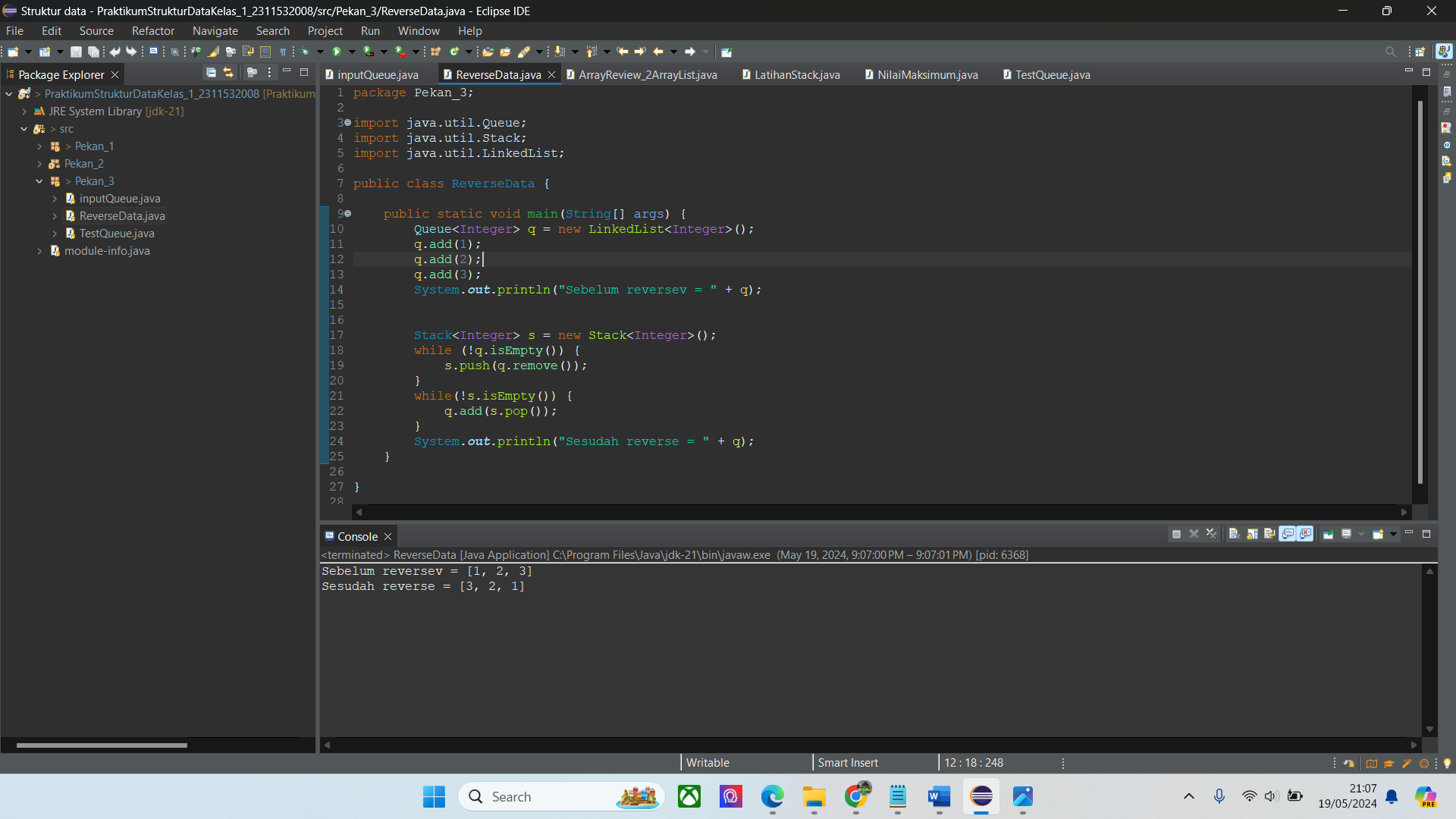
Contoh program yang menggunakan kode di atas adalah sebagai berikut :



Pada kode di atas memerlukan definisi dari “inputQueue” yang telah dijelaskan pada program sebelumnya. Menggunakan syntax “.enqueue()” untuk menambahkan isi dari queue, lalu untuk menampilkan bagian depan dari queue menggunakan syntax “.front()”, untuk menampilkan bagian belakang menggunakan “.rear()”. Lalu juga menggunakan syntax “.dequeue()” untuk menghapus isi paling depan dari queue. Untuk hasil dari program tersebut adalah sebagai berikut :



1. Menggunakan Queue, Stack, dan LinkedList untuk memutarkan isi Queue



Pada program tersebut terdapat beberapa penggunaan struktur data dan beberapa urutan yang bisa dijelaskan sebagai berikut :

1. Inisialisasi Antrian

Pertama, kita membuat objek antrian (queue) dengan tipe data Queue<Integer> dan menggunakan implementasi LinkedList<Integer>. Kemudian, kita menambahkan tiga angka ke dalam antrian dengan menggunakan metode add().

1. Menampilkan Antrian Sebelum Reverse

Kita menampilkan isi antrian sebelum di-reverse dengan menggunakan metode System.out.println().

1. Menggunakan Stack

Kita membuat objek stack dengan tipe data Stack<Integer>. Kemudian, kita menggunakan perulangan while untuk mengambil semua elemen dari antrian dan menaruhnya ke dalam stack menggunakan metode push().

1. Menghapus Elemen dari Antrian

Kita menghapus elemen dari antrian menggunakan metode remove().

1. Menambahkan Elemen ke Antrian

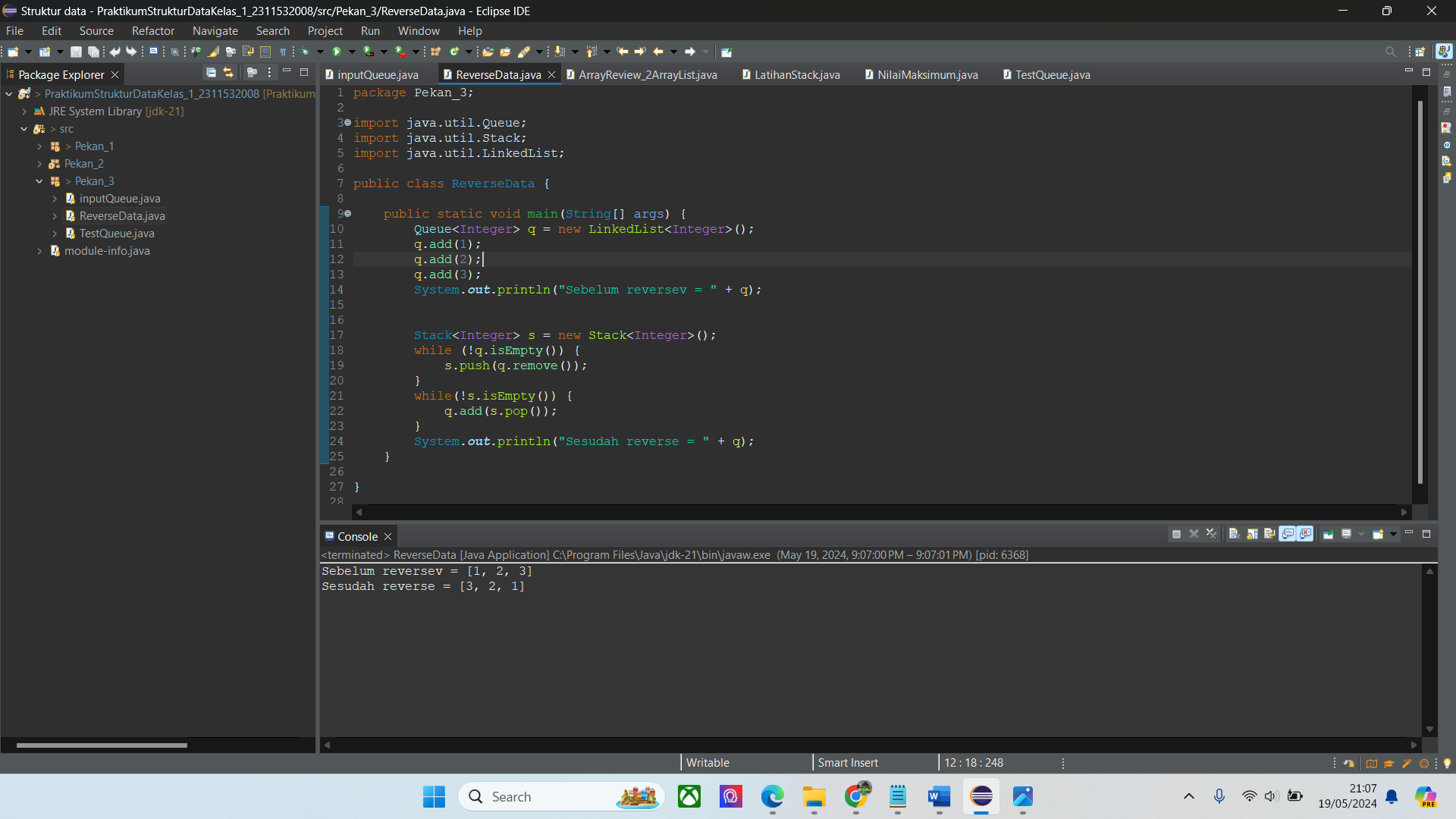
Kita menambahkan elemen dari stack ke dalam antrian menggunakan metode add().

1. Menampilkan Antrian Setelah Reverse

Kita menampilkan isi antrian setelah di-reverse dengan menggunakan metode System.out.println().

Hasil dari program ini adalah antrian yang di-reverse. Misalnya, jika kita memasukkan angka 1, 2, dan 3 ke dalam antrian, maka hasilnya adalah antrian yang berisi angka 3, 2, dan 1.

Berikut hasil dari program tersebut :



1. KESIMPULAN

Setelah praktikum, kita dapat mengetahui bahwa "queue" adalah kumpulan elemen yang disusun secara linear menurut prinsip FIFO (First In, First Out), yaitu elemen.

Elemen pertama yang keluar akan menjadi yang pertama kali dimasukkan.   
Kita akan membuat program queue dengan operasi enqueue dan dequeue pada praktikum ini. Untuk menjalankan program tersebut, diperlukan kelas yang berisi input untuk queue. Untuk melakukannya, dibuat kelas program yang berisi input untuk queue yang telah dibuat sebelumnya. Selanjutnya, diciptakan program yang dapat me-reverse urutan data dalam queue dengan memindahkan elemen anggota queue ke stack, kemudian kembali ke queue dengan urutan yang berkebalikan. Pada queue juga bisa dikombinasikan dengan Stack dan juga LinkedList yang mana akan bisa membuat program menjadi lebih efektik.