**LAPORAN HASIL PRAKTIKUM**

**ALGORITMA DAN STRUKTUR DATA**

****

**Oleh:**

**DZULFIKAR MUHAMMAD AL GHIFARI**

**NIM. 2341760071**

**SIB-1F / 08**

**D-IV SISTEM INFORMASI BISNIS**

**JURUSAN TEKNOLOGI INFORMASI**

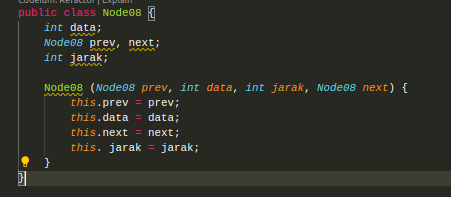
**POLITEKNIK NEGERI MALANG**

**PRAKTIKUM 14**

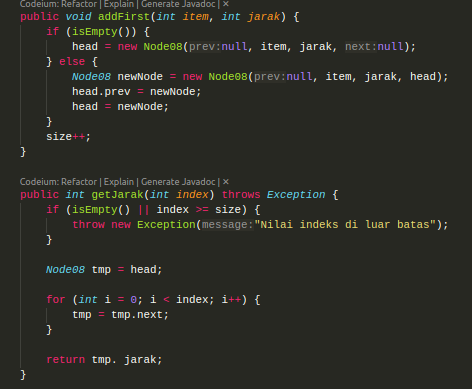
**14.2 Percobaan 1**

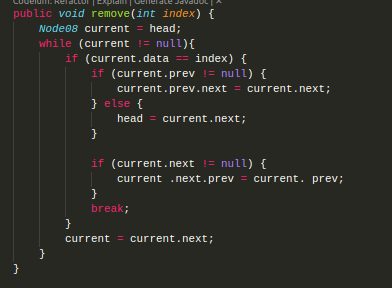
**14.2.1 Langkah Langkah percobaan**

1. Membuat Class Node

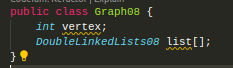


1. Menyalin class double linkedlist dan menyesuaikan beberapa method

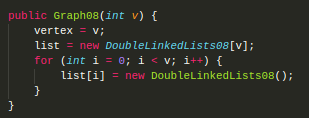




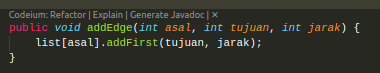
1. Membuat class graph



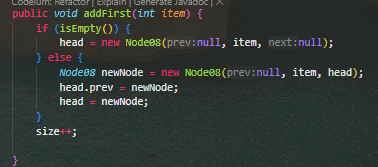
1. Membuat konnstruktor class graph



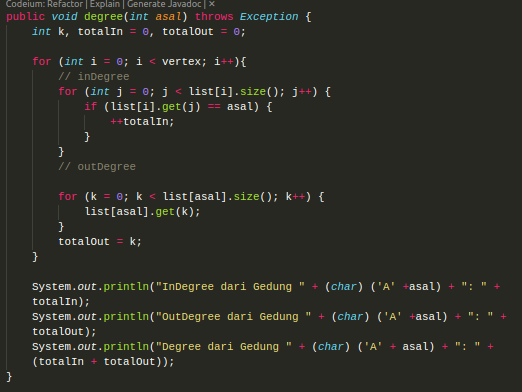
1. Menambahkan method addedge



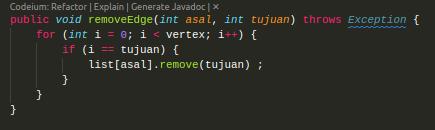
1. Menambahkan method addFirst



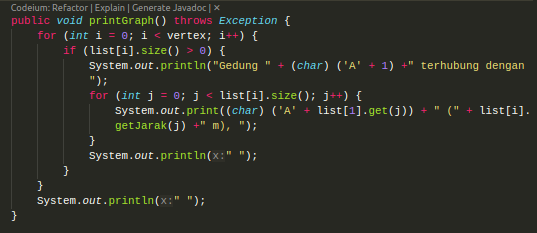
1. Menambahkan method degree



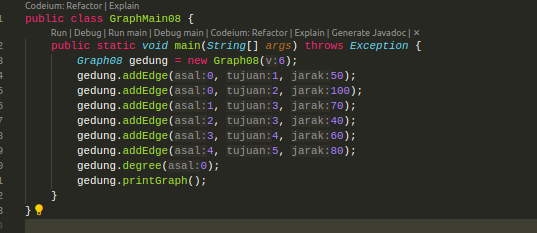
1. Menambahkan method remove edge



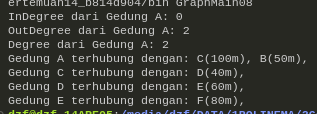
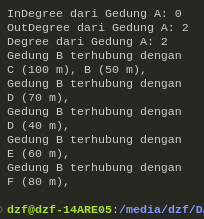
9. Membuat method print graph



10. Membuat class main



**VERIFIKASI HASIL PERCOBAAN 14.2.2**



**PERTANYAAN 14.2.3**

1. **Perbaiki kode program Anda apabila terdapat error atau hasil kompilasi kode tidak sesuai!**
2. **Pada class Graph, terdapat atribut list[] bertipe DoubleLinkedList. Sebutkan tujuan pembuatan variabel tersebut!**

Untuk menyimpan daftar tetangga (adjacency list) dari

setiap simpul (vertex) dalam graf

1. **Jelaskan alur kerja dari method removeEdge!**

**4. Apakah alasan pemanggilan method addFirst() untuk menambahkan data, bukan method add**

**jenis lain saat digunakan pada method addEdge pada class Graph?**

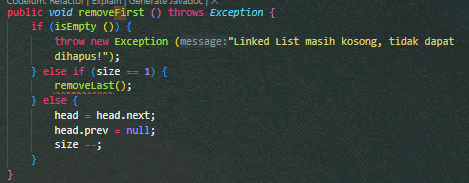
**5. Modifikasi kode program sehingga dapat dilakukan pengecekan apakah terdapat jalur antara**

**suatu node dengan node lainnya, seperti contoh berikut (Anda dapat memanfaatkan Scanner).**

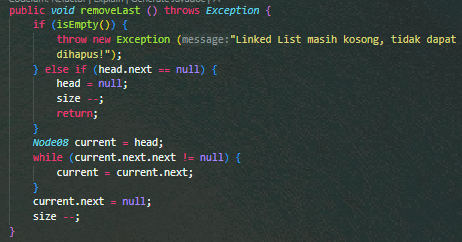
**12.3 Percobaan 2**

**12.3.1 Langkah-langkah Percobaan**

1. Menambahkan method removeFirst



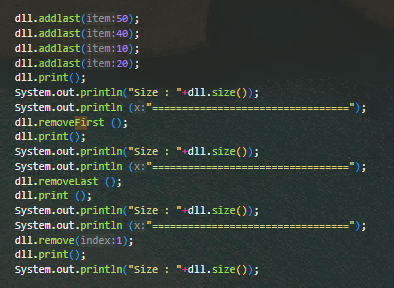
1. Menambahkan method removeLast



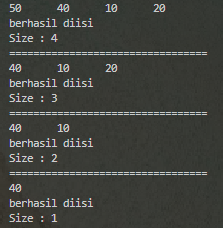
1. Menambahkan method remove



1. Menambahkan kode pada main



**VERIFIKASI HASIL PERCOBAAN 12.3.2**



**PERTANYAAN 12.3.3**

1. **Apakah maksud statement berikut pada method *removeFirst()*?**

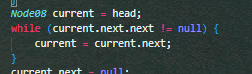
**head = head.next;**

**head.prev = null;**

head=head.next berfungsi untuk memperbarui pointer head yang menunjukan ke node berikutnya setelah node pertama yang akan dihapus.

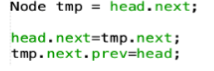
head.prev=null berfungsi untuk menghapus ponter prev karena head berasa pada daftar pertama

1. **Bagaimana cara mendeteksi posisi data ada pada bagian akhir pada method *removeLast()*?**



Dengan menggunakan iterasi loop while, mencari current.next hingga ke daftar yang terakhir. Karena daftar terakhir tidak bisa melakukan .next, jadi data tersebut menunjukan data terakhir, dan ponter akan di simpan di variable current

1. **Jelaskan alasan potongan kode program di bawah ini tidak cocok untuk perintah *remove*!**



Karena kode program di atas tidak memeriksa kondisi spesifik dari daftar

1. **Jelaskan fungsi kode program berikut ini pada fungsi *remove*!**



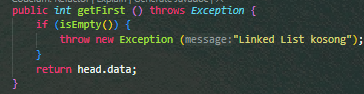
Kode program pertama berfungsi memperbarui pointer next dari node sebelumnya untuk menunjuk ke node berikutnya

Kode program kedua berfungsi memperbarui ponter prev dari node berikutnya untuk menunjuk ke node sebelumnya

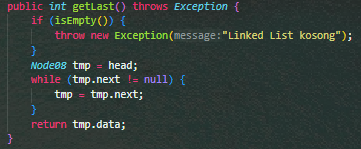
**12.4 Percobaan 2**

**12.4.1 Langkah-langkah Percobaan**

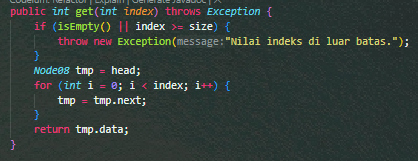
1. Menambahkan method getFIrst



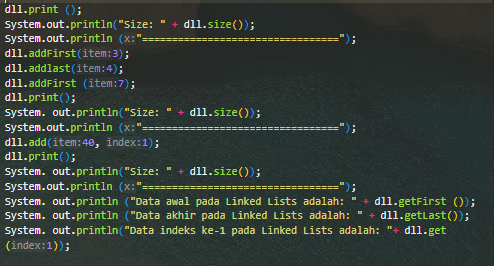
1. Menambahkan method getLast



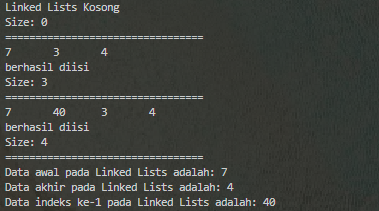
1. Menambahkan method get



1. Menambahkan kode pada main



**VERIFIKASI HASIL PERCOBAAN 12.4.2**



**PERTANYAAN 12.4.3**

1. **Jelaskan method size() pada class DoubleLinkedLists!**

Size() mengembalikan nilai size yang telah di atur pada awal menjalankan program

1. **Jelaskan cara mengatur indeks pada double linked lists supaya dapat dimulai dari indeks ke-1**

Secara default index umumnya dimulai dari 0. Namun dengan dengan sedikit mengubah pada method add dengan memberikan validasi untuk index seperti dibawah ini

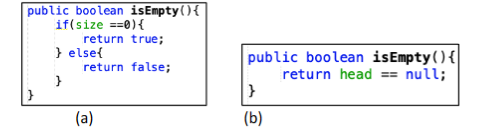


1. **Jelaskan perbedaan karakteristik fungsi Add pada Double Linked Lists dan Single Linked Lists!**

Penambahan pada single linked hanya satu arah dari depan ke belakang, tidak bisa menambahkan elemen di tengah daftar

Penambahan pada double linked memiliki 2 arah, bisa dari depan dan belakang, bisa menyisipkan kode di mana saja

1. **Jelaskan perbedaan logika dari kedua kode program di bawah ini**



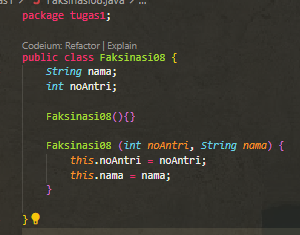
pada kode A akan memeriksa jumlah elemen, jika jumlah elemen tersebut bernilai 0 maka akan mengembalikan true, namun jika tidak sama dengan 0 maka akan mengembalikan false

Pada kode B akan memeriksa ponter head apakah berisikan null, jika iya maka akan mengembalikan true, jika tidak null maka kan mengembalikan false

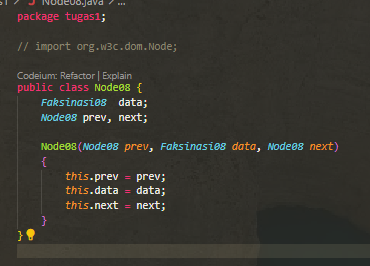
**8.4 TUGAS**

1. Buat program antrian vaksinasi menggunakan queue berbasis double linked list sesuai ilustrasi dan menu di bawah ini! (counter jumlah antrian tersisa di menu cetak(3) dan data orang yang telah divaksinasi di menu Hapus Data(2) harus ada

Class faksinasi



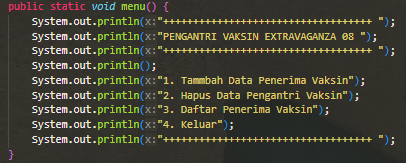
Class Node

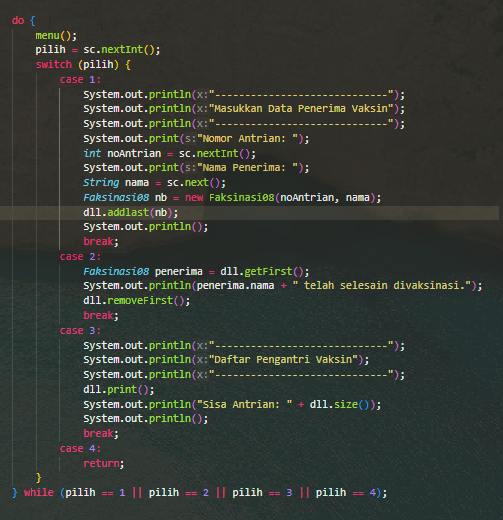


Class double berisikan hasil copy dari percobaan 1-3

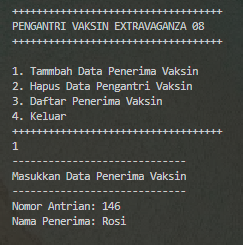
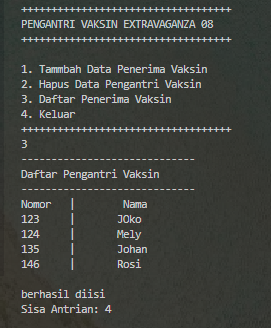


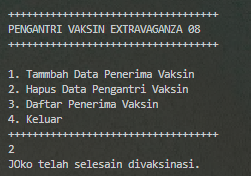
Class Main





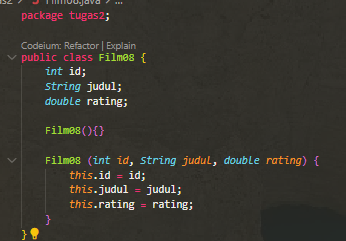
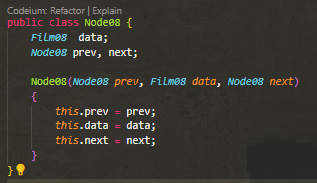
Hasil Compile



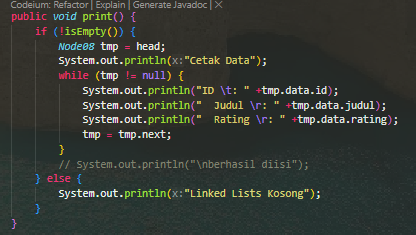


1. Buatlah program daftar film yang terdiri dari id, judul dan rating menggunakan double linked lists, bentuk program memiliki fitur pencarian melalui ID Film dan pengurutan Rating secara descending. Class Film wajib diimplementasikan dalam soal ini.

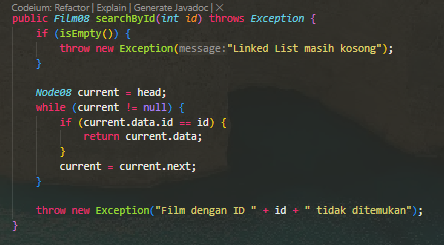
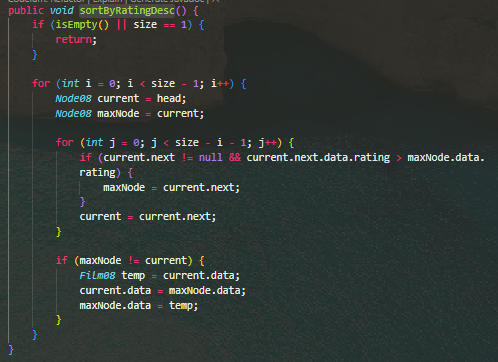
Class film Class node

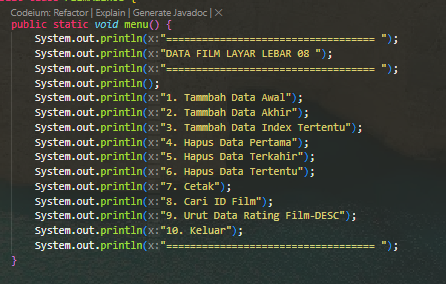
Perubahan pada class doublelinkedlist

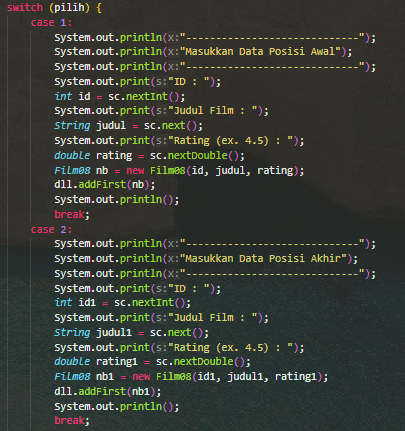
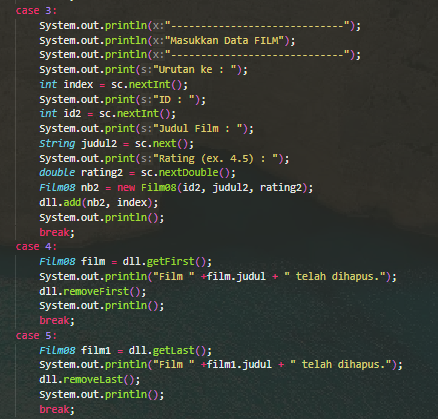


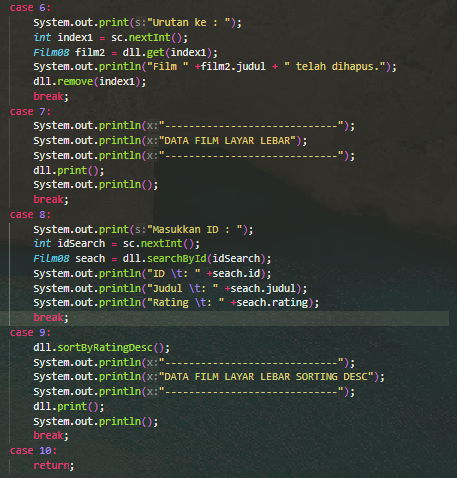
Penambahan pada class double linkedlist

Class main





Hasil compile

