**LAPORAN PRAKTIKUM**

**MATA KULIAH ALGORITMA DAN STRUKTUR DATA**

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

**PERTEMUAN - 14 - Graph**

****

**Nama : M. Zidna Billah Faza**

**NIM : 2341760030**

**Prodi : D-IV Sistem Informasi Bisnis**

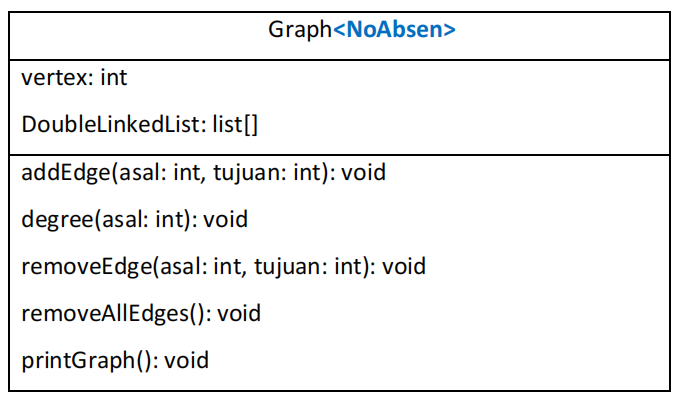
**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI MALANG**

**2024**

**Percobaan 1**

Sebuah universitas membuat program untuk memodelkan graf berarah berbobot yang mewakili gedung-gedung dan jarak antar gedung tersebut menggunakan Linked List. Setiap gedung dihubungkan dengan jalan yang memiliki jarak tertentu (dalam meter). Perhatikan class diagram Graph berikut ini.

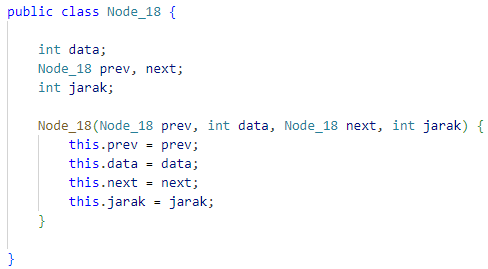


1. Buka text editor. Buat class Node<NoAbsen>.java dan class DoubleLinkedList<NoAbsen>.java sesuai dengan praktikum Double Linked List.

* Class Node\_18



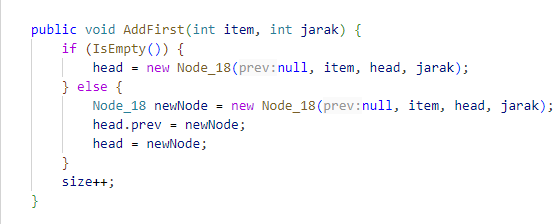
Kode program yang terdapat pada class Node belum dapat mengakomodasi kebutuhan pembuatan graf berbobot, sehingga diperlukan sedikit modifikasi. Setelah Anda menyalin kode program dari class Node pada praktikum Double Linked List, tambahkan atribut jarak bertipe int untuk menyimpan bobot graf



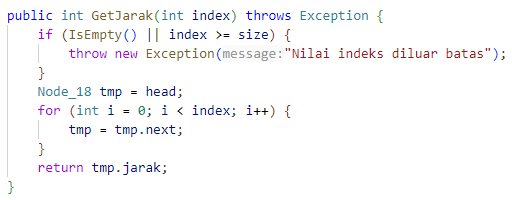
* Class DoubleLinkedList\_18



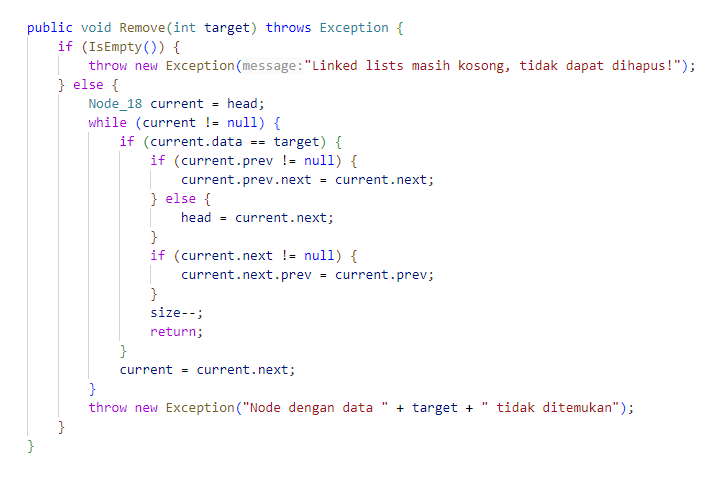
Setelah Anda menyalin kode program dari class DoubleLinkedList pada praktikum Double Linked List, lakukan modifikasi pada method addFirst agar dapat menerima parameter jarak dan digunakan saat instansiasi Node.



Selanjutnya buat method getJarak (hampir sama seperti method get) yang digunakan untuk mendapatkan nilai jarak edge antara dua node.



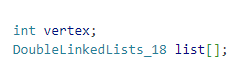
Modifikasi method remove agar dapat melakukan penghapusan edge sesuai dengan node asal dan tujuan pada graf. Pada praktikum Double Linked List, parameter index digunakan untuk menghapus data sesuai posisi pada indeks tertentu, sedangkan pada Graf ini, penghapusan didasarkan pada data node tujuan, sehingga modifikasi kode diperlukan untuk menghindari index out of bound.



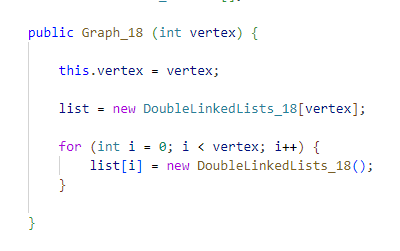
1. Buat file baru, beri nama Graph<NoAbsen>.java



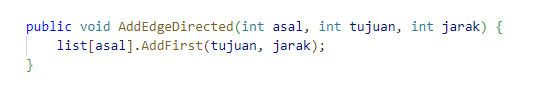
1. Lengkapi class Graph dengan atribut yang telah digambarkan di dalam pada class diagram, yang terdiri dari atribut vertex dan DoubleLinkedList.



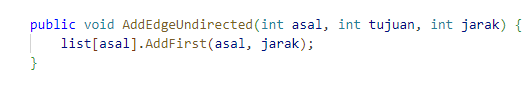
1. Tambahkan konstruktor default untuk menginisialisasi variabel vertex dan menambahkan perulangan jumlah vertex sesuai dengan panjang array yang telah ditentukan.



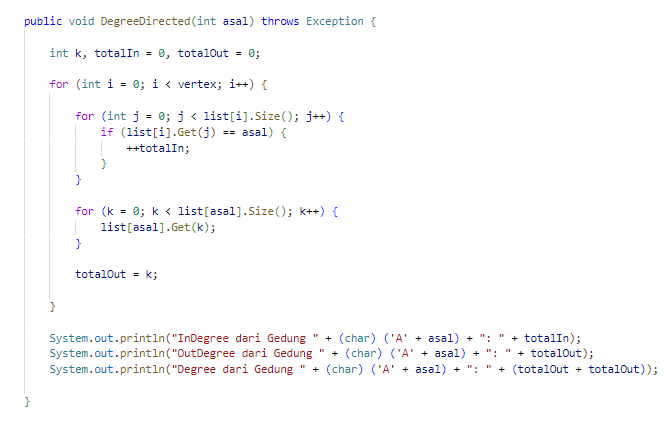
1. Tambahkan method addEdge() untuk menghubungkan dua node. Baris kode program berikut digunakan untuk graf berarah (directed).



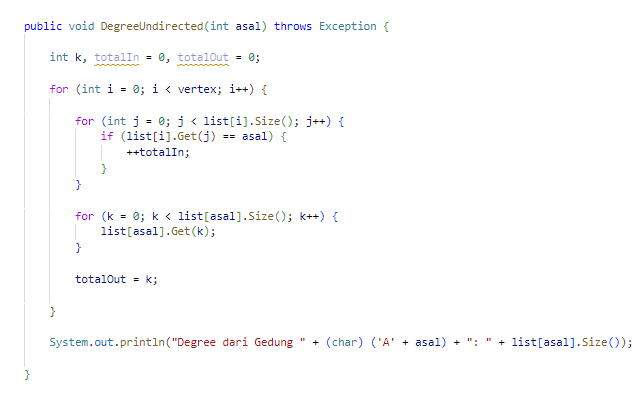
Apabila graf yang dibuat adalah undirected graph, maka tambahkan kode berikut



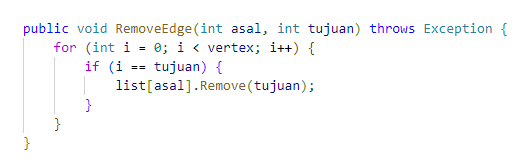
1. Tambahkan method degree() untuk menampilkan jumlah derajat lintasan pada setiap vertex. Kode program berikut digunakan untuk menghitung degree pada graf berarah



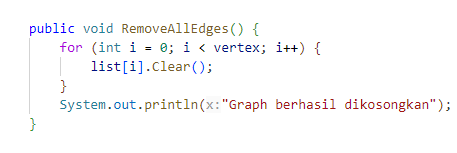
Apabila graf yang dibuat adalah undirected graph, maka cukup gunakan kode berikut.



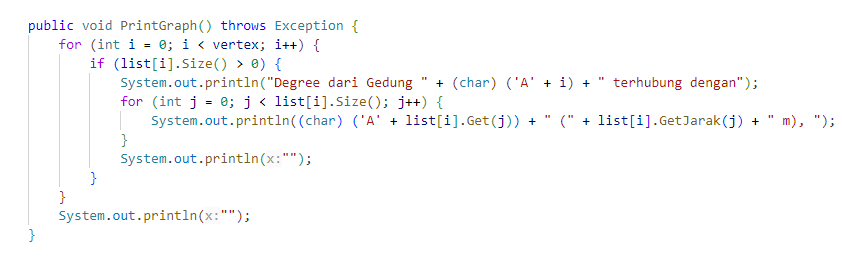
1. Tambahkan method removeEdge() untuk menghapus lintasan pada suatu graph. Penghapusan membutuhkan 2 parameter yaitu node asal dan tujuan.



1. Tambahkan method removeAllEdges() untuk menghapus semua vertex yang ada di dalam graf.



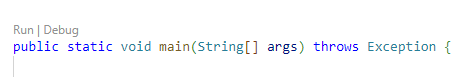
1. Tambahkan method printGraph() untuk mencetak graf



1. Buat file baru, beri nama GraphMain<NoAbsen>.java



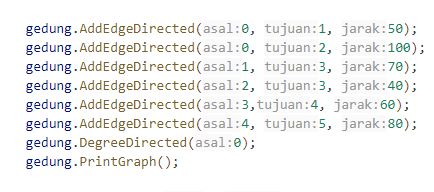
1. Tuliskan struktur dasar bahasa pemrograman Java yang terdiri dari fungsi main



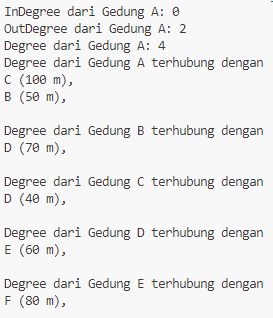
1. Di dalam fungsi main, lakukan instansiasi object Graph bernama gedung dengan nilai parameternya adalah 6.



1. Tambahkan beberapa edge pada graf, tampilkan degree salah satu node, kemudian tampilkan grafnya.

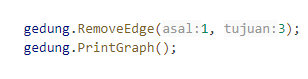


1. Compile dan run program.



Catatan: Degree harus disesuaikan dengan jenis graf yang digunakan. Pada kasus ini, digunakan directed weighted graph

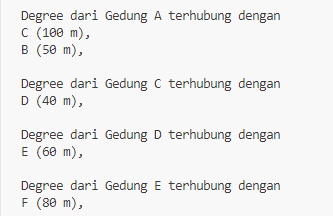
1. Tambahkan pemanggilan method removeEdge(), kemudian tampilkan kembali graf tersebut.



1. Commit dan push kode program ke Github.

<https://github.com/zidnafaz/Praktikum-Algoritma-Struktur-Data>

1. Compile dan run program serta verifikasi hasil percobaan.



**Pertanyaan Percobaan 1**

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!

Variable tersebut dibuat untuk mengimplementasikan struktur adjacency list yang efisien dan fleksibel dalam menyimpan dan mengelola graph beserta edge dan bobot (jarak) yang terkait.

1. Jelaskan alur kerja dari method removeEdge!

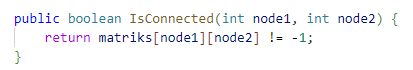
* Lakukan iterasi melalui semua vertex.
* Periksa apakah indeks iterasi (i) sama dengan vertex tujuan.
* Jika sama, panggil method Remove pada adjacency list yang sesuai (list[asal]) untuk menghapus edge ke tujuan.

Method removeEdge menghapus edge yang menghubungkan dua vertex dalam graph dengan mencari dan menghapusnya dari adjacency list yang sesuai.

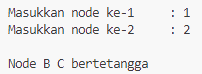
1. Apakah alasan pemanggilan method addFirst() untuk menambahkan data, bukan method add jenis lain saat digunakan pada method addEdge pada class Graph?

Penggunaan AddFirst dalam method AddEdgeDirected dan AddEdgeUndirected adalah pilihan yang logis untuk memastikan bahwa penambahan edge dilakukan dengan efisien dan sederhana, serta sesuai dengan kebutuhan representasi adjacency list pada graph.

1. Modifikasi kode program sehingga dapat dilakukan pengecekan apakah terdapat jalur antara suatu node dengan node lainnya, seperti contoh berikut (Anda dapat memanfaatkan Scanner).



Dengan program tersebut kita dapat mengecek apakah node terhubung

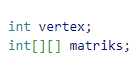


**Percobaan 2**

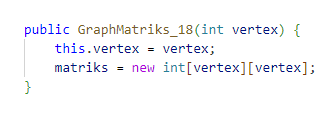
1. Buat file baru, beri nama GraphMatriks<NoAbsen>.java



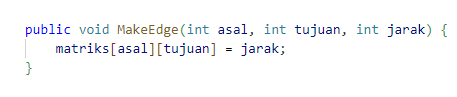
1. Lengkapi class GraphMatriks dengan atribut vertex dan matriks



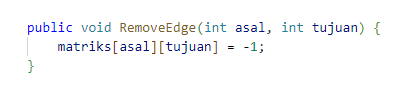
1. Tambahkan konstruktor default untuk menginisialisasi variabel vertex dan menginstansiasi panjang array dua dimensi yang telah ditentukan.



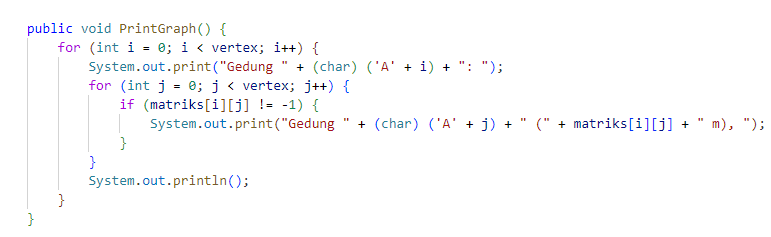
1. Untuk membuat suatu lintasan yang menghubungkan dua node, maka dibuat method makeEdge() sebagai berikut.



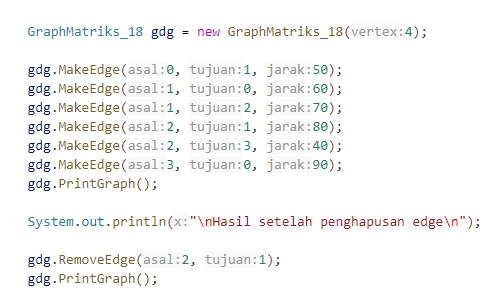
1. Tambahkan method removeEdge() untuk menghapus lintasan pada suatu graf.



1. Tambahkan method printGraph() untuk mencetak graf.



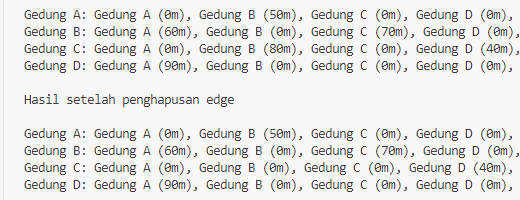
1. Tambahkan kode berikut pada file GraphMain<NoAbsen>.java yang sudah dibuat pada Percobaan 1.



1. Commit dan push kode program ke Github

<https://github.com/zidnafaz/Praktikum-Algoritma-Struktur-Data>

1. Compile dan run program serta verifikasi hasil percobaan.

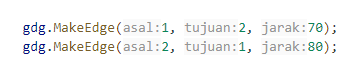


**Pertanyaan Percobaan 2**

1. Perbaiki kode program Anda apabila terdapat error atau hasil kompilasi kode tidak sesuai!
2. Apa jenis graph yang digunakan pada Percobaan 2?

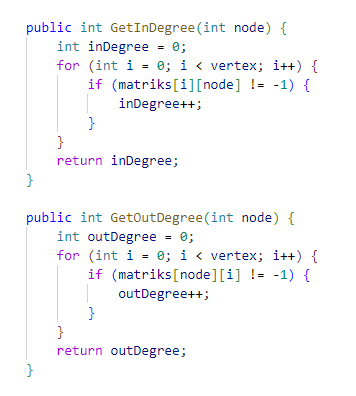
Graph tersebut adalah graph directed

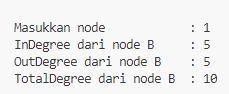
1. Apa maksud dari dua baris kode berikut?



Kode tersebut adalah penambahan edge dimana diinputkan node asal dan tujuan serta jarak antara kedua node tersebut

1. Modifikasi kode program sehingga terdapat method untuk menghitung degree, termasuk inDegree dan outDegree!



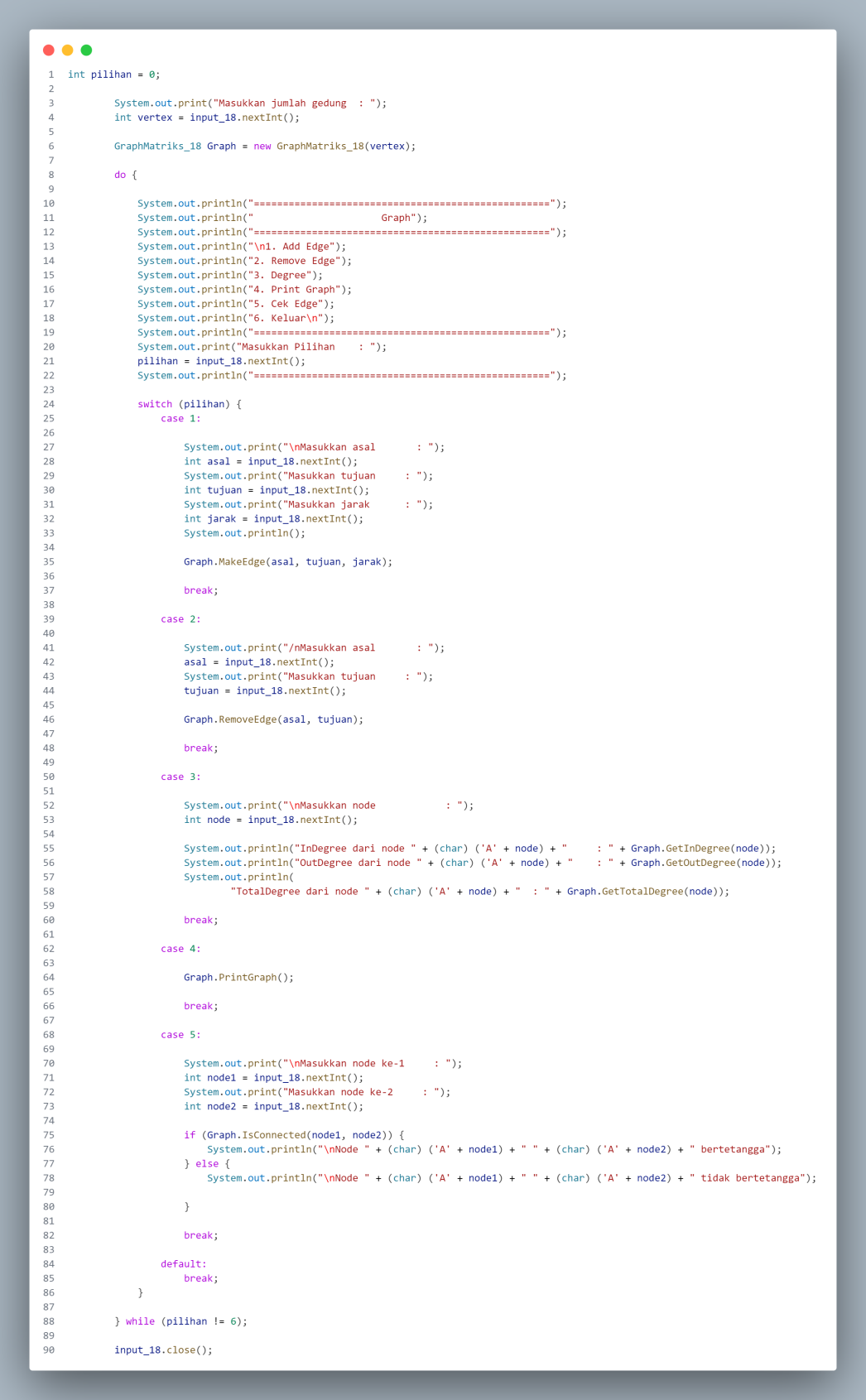


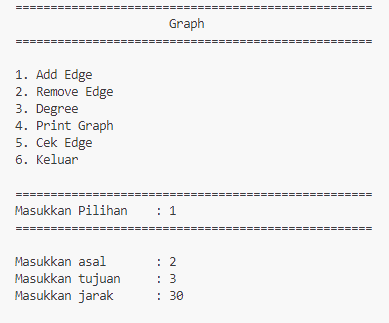
**Tugas Praktikum 1**

1. Modifikasi kode program pada class GraphMain sehingga terdapat menu program yang bersifat dinamis, setidaknya terdiri dari:

* Add Edge
* Remove Edge
* Degree
* Print Graph
* Cek Edge

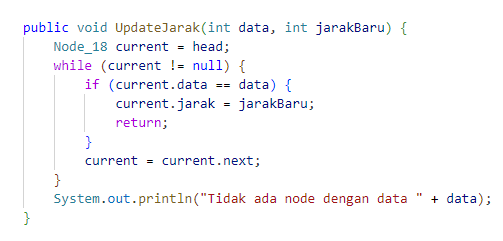
Pengguna dapat memilih menu program melalui input Scanner



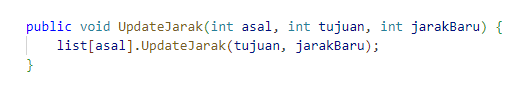


1. Tambahkan method updateJarak pada Percobaan 1 yang digunakan untuk mengubah jarak antara dua node asal dan tujuan!

Pada class DoubleLinkedList\_18

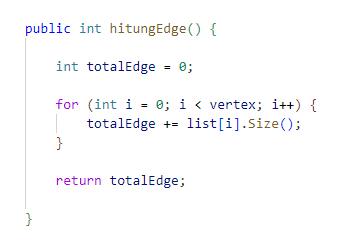


Pada class Graph\_18



1. Tambahkan method hitungEdge untuk menghitung banyaknya edge yang terdapat di dalam graf!

Pada class Graph\_18



Repository : <https://github.com/zidnafaz/Praktikum-Algoritma-Struktur-Data>