NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Graph

2. 1 Percobaan 1

```java
package P15;

public class Node24 {
    int data;
    Node24 prev, next;
    int jarak;

    Node24(Node24 prev, int data, int jarak, Node24 next) {
        this.prev = prev;
        this.data = data;
        this.next = next;
        this.jarak = jarak;
    }
}
```

```java
package P15;

public class Graph24 {
    int vertex;
    DoubleLinkedLists list[];

    Graph24(int v) {
        vertex = v;
        list = new DoubleLinkedLists[v];
        for (int i = 0; i < v; i++) {
            list[i] = new DoubleLinkedLists();
        }
    }

    void addEdge(int asal, int tujuan, int jarak) {
        list[asal].addFirst(tujuan, jarak);
    }

    void degree(int asal) throws Exception {
        int k, totalIn = 0, totalOut = 0;
        for (int i = 0; i < vertex; i++) {
            // inDegree
            for (int j = 0; j < list[i].size(); j++) {
                if (list[i].get(j) == asal) {
                    ++totalIn;
                }
            }
            // outDegree
            for (k = 0; k < list[asal].size(); k++) {
                list[asal].get(k);
            }
            totalOut = k;
        }
        System.out.println("InDegree dari Gedung " + (char) ('A' + asal) + " : " + totalIn);
        System.out.println("OutDegree dari Gedung " + (char) ('A' + asal) + " : " + totalOut);
        System.out.println("Degree dari Gedung " + (char) ('A' + asal) + " : " + (totalIn + totalOut));
    }

    void removeEdge(int asal, int tujuan) throws Exception {
        for (int i = 0; i < vertex; i++) {
            if (i == tujuan) {
                list[asal].remove(tujuan);
            }
        }
    }
```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Graph

```
void removeAllEdges() {
    for (int i = 0; i < vertex; i++) {
        list[i].clear();
    }
    System.out.println(x:"Graf berhasil dikosongkan");
}

void printGraph() throws Exception {
    for (int i = 0; i < vertex; i++) {
        if (list[i].size() > 0) {
            System.out.println("Gedung " + (char) ('A' + i) + " terhubung dengan ");
            for (int j = 0; j < list[i].size(); j++) {
                System.out.print((char) ('A' + list[i].get(j)) + " (" + list[i].getJarak(j) + " m), ");
            }
            System.out.println(x:"");
        }
    }
    System.out.println(x:"");
}
```

```
package P15;

public class GraphMain24 {
    Run | Debug
    public static void main(String[] args) throws Exception {
        Graph24 gedung = new Graph24(v:6);
        gedung.addEdge(asal:0, tujuan:1, jarak:50);
        gedung.addEdge(asal:0, tujuan:2, jarak:100);
        gedung.addEdge(asal:1, tujuan:3, jarak:70);
        gedung.addEdge(asal:2,tujuan:3, jarak:40);
        gedung.addEdge(asal:3, tujuan:4, jarak:60);
        gedung.addEdge(asal:4, tujuan:5, jarak:80);
        gedung.degree(asal:0);
        gedung.printGraph();
    }
}
```

```
InDegree dari Gedung A : 0
OutDegree dari Gedung A : 2
Degree dari Gedung A : 2
Gedung A terhubung dengan
C (100 m), B (50 m),
Gedung B terhubung dengan
D (70 m),
Gedung C terhubung dengan
D (40 m),
Gedung D terhubung dengan
E (60 m),
Gedung E terhubung dengan
F (80 m),
```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Graph

Pertanyaan :

1. *Perbaiki kode program Anda apabila terdapat error atau hasil kompilasi kode tidak sesuai!*

```
void remove(int index) throws Exception {
    Node24 current = head;
    while (current != null) {
        if (current.data == index) {
            if (current.prev != null) {
                current.prev.next = current.next;
            } else {
                head = current.next;
            }
            if (current.next != null) {
                current.next.prev = current.prev;
            }
            break;
        }
        current = current.next;
    }
    size--;      You, 1 minute ago • Uncommitted c
}
```

```
InDegree dari Gedung A : 0
OutDegree dari Gedung A : 2
Degree dari Gedung A : 2
Gedung A terhubung dengan
C (100 m), B (50 m),
Gedung B terhubung dengan
D (70 m),
Gedung C terhubung dengan
D (40 m),
Gedung D terhubung dengan
E (60 m),
Gedung E terhubung dengan
F (80 m),

Gedung A terhubung dengan
C (100 m), B (50 m),
Gedung C terhubung dengan
D (40 m),
Gedung D terhubung dengan
E (60 m),
Gedung E terhubung dengan
F (80 m),
```

2. *Pada class Graph, terdapat atribut list[] bertipe DoubleLinkedList. Sebutkan tujuan pembuatan variabel tersebut!*
   Jawab : Variabel tersebut digunakan sebagai wadah bagi vertex sehingga tiap vertex memiliki satu doublelinked list

3. *Jelaskan alur kerja dari method removeEdge!*
   Jawab : Langkah pertama yang dilakukan adalah mencari vertex dari vertex tujuan yang akan dihapus, jika vertex ditemukan barulah kita melakukan penghapusan pada vertex asal yang memiliki adjacency dengan vertex tujuan.

4. *Apakah alasan pemanggilan method addFirst() untuk menambahkan data, bukan method add jenis lain saat digunakan pada method addEdge pada class Graph?*
   Jawab : Hal ini dilakukan untuk memudahkan penambahan data tanpa alasan lain, karna pada dasarnya urutan pada adjacency itu tidak penting

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

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Graph

```
Masukkan gedung asal: 2
Masukkan gedung tujuan: 3
Gedung C dan D bertetangga

Masukkan gedung asal: 2
Masukkan gedung tujuan: 5
Gedung C dan F tidak bertetangga
```

Jawab :

```
boolean checkAdjacency (int asal, int tujuan) throws Exception{
    for (int i = 0; i < list[asal].size(); i++) {
        if (list[asal].get(i) == tujuan) {
            return true;
        }
    }
    return false;
}
```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Graph

```
public static void main(String[] args) throws Exception {
    Graph24 gedung = new Graph24(v:6);
    gedung.addEdge(asal:0, tujuan:1, jarak:50);
    gedung.addEdge(asal:0, tujuan:2, jarak:100);
    gedung.addEdge(asal:1, tujuan:3, jarak:70);
    gedung.addEdge(asal:2,tujuan:3, jarak:40);
    gedung.addEdge(asal:3, tujuan:4, jarak:60);
    gedung.addEdge(asal:4, tujuan:5, jarak:80);
    gedung.degree(asal:0);
    gedung.printGraph();
    gedung.removeEdge(asal:1, tujuan:3);
    gedung.printGraph();
    checkAdjacency(gedung);
    checkAdjacency(gedung);
}

static void checkAdjacency(Graph24 check) throws Exception{
    Scanner sc = new Scanner(System.in);
    System.out.print(s:"Masukkan gedung asal : ");
    int asal = sc.nextInt();
    System.out.print(s:"Masukkan gedung tujuan : ");
    int tujuan = sc.nextInt();      You, 41 seconds ago • Uncommitted changes
    if (check.checkAdjacency(asal, tujuan)) {
        System.out.println(adjacencyMessage(asal, tujuan) + " bertetangga");
    } else {
        System.out.println(adjacencyMessage(asal, tujuan) + " tidak bertetangga");
    }
}

static String adjacencyMessage(int asal, int tujuan) {
    return "Gedung " + (char) ('A' + asal) + " dan " + "Gedung " + (char) ('A' + tujuan);
}
```

```
Masukkan gedung asal : 2
Masukkan gedung tujuan : 3
Gedung C dan Gedung D bertetangga

Masukkan gedung asal : 2
Masukkan gedung tujuan : 5
Gedung C dan Gedung F tidak bertetangga
```

2.2 Percobaan 2

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Graph

```java
package P15;

public class GraphMatriks24 {
    int vertex;
    int[][] matriks;

    GraphMatriks24(int v) {
        vertex = v;
        matriks = new int[v][v];
    }

    void makeEdge(int asal, int tujuan, int jarak) {
        matriks[asal][tujuan] = jarak;
    }

    void removeEdge(int asal, int tujuan) {
        matriks[asal][tujuan] = 0;
    }

    void printGraph() {
        for (int i = 0; i < vertex; i++) {
            System.out.print("Gedung " + (char) ('A' + i) + ": ");
            for (int j = 0; j < vertex; j++) {
                if (matriks[i][j] != -1) {
                    System.out.print("Gedung " + (char) ('A' + j) + " (" + matriks[i][j] + " m), ");
                }
            }
            System.out.println();
        }
    }
}
```

```java
GraphMatriks24 gdg = new GraphMatriks24(v:4);
gdg.makeEdge(asal:0, tujuan:1, jarak:50);
gdg.makeEdge(asal:1, tujuan:0, jarak:60);
gdg.makeEdge(asal:1, tujuan:2, jarak:70);
gdg.makeEdge(asal:2, tujuan:1, jarak:80);
gdg.makeEdge(asal:2, tujuan:3, jarak:40);
gdg.makeEdge(asal:3, tujuan:0, jarak:90);
gdg.printGraph();
System.out.println(x:"Hasil setelah penghapusan edge");
gdg.removeEdge(asal:2, tujuan:1);       You, 6 minutes a
gdg.printGraph();
```

```
Gedung A: Gedung A (0 m), Gedung B (50 m), Gedung C (0 m), Gedung D (0 m),
Gedung B: Gedung A (60 m), Gedung B (0 m), Gedung C (70 m), Gedung D (0 m),
Gedung C: Gedung A (0 m), Gedung B (80 m), Gedung C (0 m), Gedung D (40 m),
Gedung D: Gedung A (90 m), Gedung B (0 m), Gedung C (0 m), Gedung D (0 m),
Hasil setelah penghapusan edge
Gedung A: Gedung A (0 m), Gedung B (50 m), Gedung C (0 m), Gedung D (0 m),
Gedung B: Gedung A (60 m), Gedung B (0 m), Gedung C (70 m), Gedung D (0 m),
Gedung C: Gedung A (0 m), Gedung B (0 m), Gedung C (0 m), Gedung D (40 m),
Gedung D: Gedung A (90 m), Gedung B (0 m), Gedung C (0 m), Gedung D (0 m),
PS D:\Vian\kuliah\sem-2\AlgoritmaStrukturData>
```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Graph

Pertanyaan :

1. *Perbaiki kode program Anda apabila terdapat error atau hasil kompilasi kode tidak sesuai!*

2. *Apa jenis graph yang digunakan pada Percobaan 2?*
   Jawab : Graph Matriks

3. *Apa maksud dari dua baris kode berikut?*

```
gdg.makeEdge(1, 2, 70);
gdg.makeEdge(2, 1, 80);
```

   Jawab : Memberikan value atau nilai jarak pada matriks yang berada pada indeks baris 1 kolom 2 dengan value 70 dan indeks baris 2 kolom 1 dengan value 80

4. *Modifikasi kode program sehingga terdapat method untuk menghitung degree, termasuk inDegree dan outDegree!*
   Jawab :

```
void degree(int asal) {
    int totalIn = 0, totalOut = 0;
    // inDegree
    for (int i = 0; i < matriks[asal].length; i++) {
        if (matriks[asal][i] != 0) {
            totalIn++;
        }
    }
    // outDegree
    for (int i = 0; i < vertex; i++) {
        if (matriks[i][asal] != 0) {
            totalOut++;      You, 3 minutes ago • Uncommitted changes
        }
    }
    System.out.println("InDegree dari Gedung " + (char) ('A' + asal) + " : " + totalIn);
    System.out.println("OutDegree dari Gedung " + (char) ('A' + asal) + " : " + totalOut);
    System.out.println("Degree dari Gedung " + (char) ('A' + asal) + " : " + (totalIn + totalOut));
}
```

```
Gedung A: Gedung A (0 m), Gedung B (50 m), Gedung C (0 m), Gedung D (0 m),
Gedung B: Gedung A (60 m), Gedung B (0 m), Gedung C (70 m), Gedung D (0 m),
Gedung C: Gedung A (0 m), Gedung B (0 m), Gedung C (0 m), Gedung D (40 m),
Gedung D: Gedung A (90 m), Gedung B (0 m), Gedung C (0 m), Gedung D (0 m),
InDegree dari Gedung B : 2
OutDegree dari Gedung B : 1
Degree dari Gedung B : 3
```