NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Graph (Jobsheet 12)

PERCOBAAN 1

```
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),
```

```java
package P15;

public class GraphMain27 {
    public static void main(String[] args) throws Exception {
        Graph27 gedung = new Graph27(6);
        gedung.addEdge(0, 1, 50);
        gedung.addEdge(0, 2, 100);
        gedung.addEdge(1, 3, 70);
        gedung.addEdge(2, 3, 40);
        gedung.addEdge(3, 4, 60);
        gedung.addEdge(4, 5, 80);
        gedung.degree(0);
        gedung.printGraph();

    }
}
```

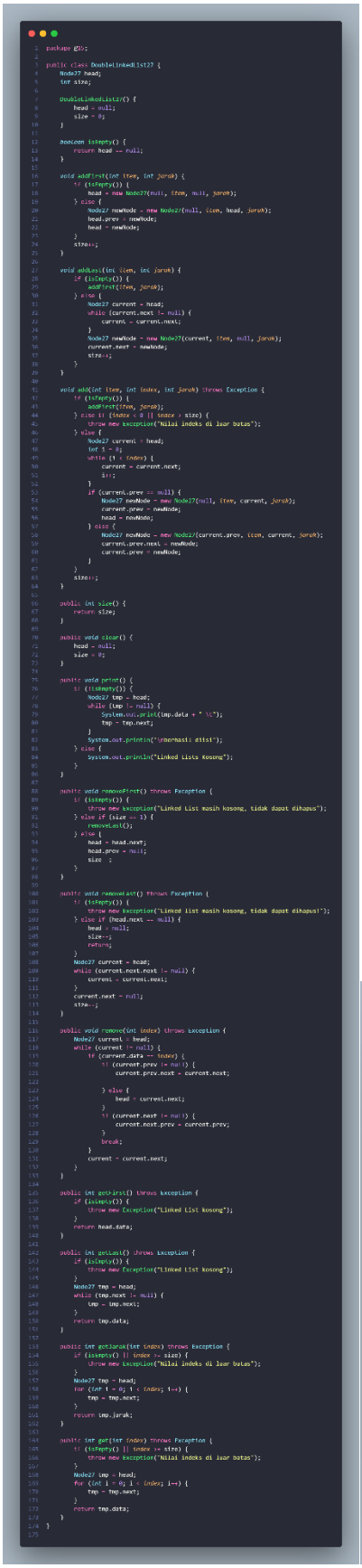
RUNNING Langkah 17

```
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),
```

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Graph (Jobsheet 12)

```java
package P15;

public class Graph27 {
    int vertex;
    DoubleLinkedList27 list[];

    public Graph27(int v) {
        vertex = v;
        list = new DoubleLinkedList27[v];
        for (int i = 0; i < v; i++) {
            list[i] = new DoubleLinkedList27();
        }
    }

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

    public void degree(int asal) throws Exception {
        int k, totalIn = 0, totalOut = 0;
        for (int i = 0; i < vertex; i++) {
            for (int j = 0; j < list[i].size(); j++) {
                if (list[i].get(j) == asal) {
                    ++totalIn;
                }
            }
            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));
    }

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

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

    public 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("");
            }
        }
        System.out.println("");
    }
}

```

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Graph (Jobsheet 12)



```
package P15;

public class Node27 {
    int data;
    Node27 prev;
    Node27 next;
    int jarak;

    Node27(Node27 prev, int data, Node27 next, int jarak) {
        this.prev = prev;
        this.data = data;
        this.next = next;
        this.jarak = jarak;

    }
}
```

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Graph (Jobsheet 12)

Pertanyaan

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

   Jawab:

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

```
public void remove(int index) throws Exception {
    if (isEmpty() || index >= size) {
        throw new Exception("Nilai indeks di luar batas");
    }

    Node27 current = head;
    for (int i = 0; i < index; i++) {
        current = current.next;
    }
    if (current.prev != null) {
        current.prev.next = current.next;
    } else {
        head = current.next;
    }

    if (current.next != null) {
        current.next.prev = current.prev;
    }
    size--;
}
```

2. Pada class Graph, terdapat atribut list[] bertipe DoubleLinkedList. Sebutkan tujuan pembuatan variabel tersebut!

   Jawab:

   Tujuan pembuatan variabel list[] adalah untuk menyimpan daftar adjacency dari setiap vertex

3. Jelaskan alur kerja dari method removeEdge!

   Jawab:

   Mengiterasi daftar adjacency , mencari edge yang menuju ke tujuan dan menghapusnya jika ditemukan.

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Graph (Jobsheet 12)

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

   Jawab:

   Penggunaan method addFirst() pada class graph ini karena efisiensi dan kesederhanaan implementasi, memanfaatkan karakteristik dari linked list untuk operasi penambahan yang cepat dan mudah dikelola.

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

```
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 adjacency(int start, int end) throws Exception {
    return list[start].size() > 0 && list[start].get(index:0) == end;
}
```

```
Menu:
1. Cek Gedung
0. Exit
Pilih Menu:
1
Masukkan gedung asal: 2
Masukkan gedung tujuan: 3
Gedung C dan D Bertetangga
Menu:
1. Cek Gedung
0. Exit
Pilih Menu:
1
Masukkan gedung asal: 2
Masukkan gedung tujuan: 5
Gedung C dan F tidak bertetangga
Menu:
1. Cek Gedung
0. Exit
Pilih Menu:
```

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Graph (Jobsheet 12)

```java
package P15;

import java.util.Scanner;

public class GraphMain27 {
    public static void main(String[] args) throws Exception {
        Scanner roy = new Scanner(System.in);
        Graph27 gedung = new Graph27(6);
        gedung.addEdge(0, 1, 50);
        gedung.addEdge(0, 2, 100);
        gedung.addEdge(1, 3, 70);
        gedung.addEdge(2, 3, 40);
        gedung.addEdge(3, 4, 60);
        gedung.addEdge(4, 5, 80);
        // gedung.degree(0);
        // gedung.printGraph();

        gedung.removeEdge(1, 3);
        gedung.printGraph();

        boolean isBreak = false;
        while (!isBreak) {
            System.out.println("Menu: ");
            System.out.println("1. Cek Gedung");
            System.out.println("0. Exit");
            System.out.println("Pilih Menu: ");
            int choice = roy.nextInt();
            roy.nextLine();

            switch (choice) {
                case 1:
                    System.out.print("Masukkan gedung asal: ");
                    int asal = roy.nextInt();
                    System.out.print("Masukkan gedung tujuan: ");
                    int tujuan = roy.nextInt();

                    if (asal < 0 || asal > 5 || tujuan < 0 || tujuan > 5) {
                        System.out.println("Gedung yang dimasukkan tidak valid. Masukkan gedung dalam rentang 0-5");

                    } else {
                        if (gedung.adjacency(asal, tujuan)) {
                            System.out.println(
                                    "Gedung " + (char) ('A' + asal) + " dan " + (char) ('A' + tujuan) + " Bertetangga");
                        } else {
                            System.out.println("Gedung " + (char) ('A' + asal) + " dan " + (char) ('A' + tujuan)
                                    + " tidak bertetangga");
                        }
                    }
                    break;

                case 0:
                    isBreak = true;
                    System.out.println("Terimakasih telah menggunakan program");
                    break;
                default:
                    System.out.println("Pilihan salah");
                    break;
            }
        }

    }
}

```

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Graph (Jobsheet 12)

Percobaan 2

```
Gedung A: Gedung A(0m), Gedung B(50m), Gedung C(0m), Gedung D(0m),
Gedung B: Gedung A(60m), Gedung B(0m), Gedung C(70m), Gedung D(0m),
Gedung C: Gedung A(0m), Gedung B(80m), Gedung C(0m), Gedung D(40m),
Gedung D: Gedung A(90m), Gedung B(0m), Gedung C(0m), Gedung D(0m),
Hasil setelah penghapusan edge
Gedung A: Gedung A(0m), Gedung B(50m), Gedung C(0m), Gedung D(0m),
Gedung B: Gedung A(60m), Gedung B(0m), Gedung C(70m), Gedung D(0m),
Gedung C: Gedung A(0m), Gedung B(0m), Gedung C(0m), Gedung D(40m),
Gedung D: Gedung A(90m), Gedung B(0m), Gedung C(0m), Gedung D(0m),
```

```java
package P15;

public class GraphMatriks27 {
    int vertex;
    int[][] matriks;

    public GraphMatriks27(int v) {
        vertex = v;
        matriks = new int[v][v];
    }

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

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

    public 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();
        }
    }

}

```

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Graph (Jobsheet 12)

```
public static void main(String[] args) throws Exception {

    GraphMatriks27 gdg = new GraphMatriks27(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);
    gdg.printGraph();
```

Pertanyaan

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

   Jawab:

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

2. Apa jenis graph yang digunakan pada Percobaan 2?

   Jawab:

   Graph yang digunakan adalah direct weight graph

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Graph (Jobsheet 12)

3. Apa maksud dari dua baris kode berikut?

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

Jawab:

Dua baris kode diatas sedang menambahkan dua tepi berarah (directed edges) dengan bobot / jarak (weights) ke dalam graph.

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

Jawab:

```
Gedung A: Gedung A(0m), Gedung B(50m), Gedung C(0m), Gedung D(0m),
Gedung B: Gedung A(60m), Gedung B(0m), Gedung C(70m), Gedung D(0m),
Gedung C: Gedung A(0m), Gedung B(80m), Gedung C(0m), Gedung D(40m),
Gedung D: Gedung A(90m), Gedung B(0m), Gedung C(0m), Gedung D(0m),
Hasil setelah penghapusan edge
Gedung A: Gedung A(0m), Gedung B(50m), Gedung C(0m), Gedung D(0m),
Gedung B: Gedung A(60m), Gedung B(0m), Gedung C(70m), Gedung D(0m),
Gedung C: Gedung A(0m), Gedung B(0m), Gedung C(0m), Gedung D(40m),
Gedung D: Gedung A(90m), Gedung B(0m), Gedung C(0m), Gedung D(0m),
Gedung A: InDegree = 2, OutDegree = 1, Total Degree = 3
Gedung B: InDegree = 1, OutDegree = 2, Total Degree = 3
Gedung C: InDegree = 1, OutDegree = 1, Total Degree = 2
Gedung D: InDegree = 1, OutDegree = 1, Total Degree = 2
```

```
GraphMatriks27 gdg = new GraphMatriks27(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);
gdg.printGraph();
for (int i = 0; i < 4; i++) {
    System.out.println("Gedung " + (char) ('A' + i) + ": InDegree = " + gdg.inDegree(i) + ", OutDegree = "
            + gdg.outDegree(i) + ", Total Degree = " + gdg.degree(i));
}
```

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Graph (Jobsheet 12)

```java
public int inDegree(int gedung) {
    int inDegree = 0;
    for (int i = 0; i < vertex; i++) {
        if (matriks[i][gedung] != 0) {
            inDegree++;
        }

    }
    return inDegree;
}

public int outDegree(int gedung) {
    int OutDegree = 0;
    for (int j = 0; j < vertex; j++) {
        if (matriks[gedung][j] != 0) {
            OutDegree++;
        }
    }
    return OutDegree;
}

public int degree(int gedung) {
    return inDegree(gedung) + outDegree(gedung);
}
```

LINK GITHUB:
https://github.com/RoyW12/AlgoritmaStrukturData_1G_28