NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Graph

LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

2. 1. Percobaan 1

Class Node :

PrakASD_1F_13 > src > P15 > Percobaan1 > Node13.java > ...

```java
package P15.Percobaan1;

Codeium: Refactor | Explain
public class Node13 {
    int data;
    Node13 prev, next;
    int jarak;

    public Node13(Node13 prev, int data, int jarak, Node13 next) {
        this.prev = prev;
        this.data = data;
        this.next = next;
        this.jarak = jarak;
    }
}
```

Class DoubleLinked List :

PrakASD_1F_13 > src > P15 > Percobaan1 > DoubleLinkedList13.java > {} P15.Percobaan1

```java
package P15.Percobaan1;

Codeium: Refactor | Explain
public class DoubleLinkedList13 {
    Node13 head;
    int size;

    public DoubleLinkedList13() {
        head = null;
        size = 0;
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public boolean isEmpty() {
        return head == null;
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public int size() {
        return size;
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public void clear() {
        head = null;
        size = 0;
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public int get(int index) throws Exception {
        if (isEmpty() || index >= size) {
            throw new Exception(message:"Nilai indeks di luar batas.");
        }
        Node13 current = head;
        for (int i = 0; i < index; i++) {
            current = current.next;
        }
        return current.data;
    }
```

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Graph

```
36      public void addFirst(int item, int jarak) {
37          if (isEmpty()) {
38              head = new Node13(prev:null, item, jarak, next:null);
39          } else {
40              Node13 newNode = new Node13(prev:null, item, jarak, head);
41              head.prev = newNode;
42              head = newNode;
43          }
44          size++;
45      }
46
        Codeium: Refactor | Explain | Generate Javadoc | ×
47      public int getJarak(int index) throws Exception {
48          if (isEmpty() || index >= size) {
49              throw new Exception(message:"Nilai ideks di luar batas");
50          }
51          Node13 tmp = head;
52          for (int i = 0; i < index; i++) {
53              tmp = tmp.next;
54          }
55          return tmp.jarak;
56      }
57
        Codeium: Refactor | Explain | Generate Javadoc | ×
58      public void remove(int index) {
59          Node13 current = head;
60          while (current != null) {
61              if (current.data == index) {
62                  if (current.prev != null) {
63                      current.prev.next = current.next;
64                  } else {
65                      head = current.next;
66                  }
67                  if (current.next != null) {
68                      current.next.prev = current.prev;
69                  }
70                  break;
71              }
72              current = current.next;
73          }
74      }
75  }
76
```

NAMA	: Gilang Purnomo
NIM	: 2341720042
NO ABSEN	: 13
KELAS	: 1F
MATERI	: Graph

Class Graph :

PrakASD_1F_13 > src > P15 > Percobaan1 > Graph13.java > ...

```java
package P15.Percobaan1;

Codeium: Refactor | Explain
public class Graph13 {
    int vertex;
    DoubleLinkedList13 list[];

    public Graph13(int v) {
        vertex = v;
        list = new DoubleLinkedList13[v];
        for (int i = 0; i < v; i++) {
            list[i] = new DoubleLinkedList13();
        }
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public void addEdge(int asal, int tujuan, int jarak) {
        list[asal].addFirst(tujuan, jarak);
        // apabila undirected
        // list[tujuan].addFirst(tujuan, jarak);
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public 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));
        // apabila undirected
        // System.out.println("Degree dari Gedung " + (char) ('A' + asal) + ": " + list[asal].size());
    }

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

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public void removeAllEdges() {
        for (int i = 0; i < vertex; i++) {
            list[i].clear();
        }
        System.out.println(x:"Graf berhasil dikosongkan");
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    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(x:"");
            }
        }
        System.out.println(x:"");
    }
}
```

NAMA	: Gilang Purnomo
NIM	: 2341720042
NO ABSEN	: 13
KELAS	: 1F
MATERI	: Graph

Class Main :

PrakASD_1F_13 > src > P15 > Percobaan1 > GraphMain13.java > ...

```java
package P15.Percobaan1;

Codeium: Refactor | Explain
public class GraphMain13 {
    Run | Debug | Codeium: Refactor | Explain | Generate Javadoc | ×
    public static void main(String[] args) throws Exception {
        Graph13 gedung = new Graph13(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();
    }
}
```

Output :

```
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 B terhubung dengan
Exception in thread "main" java.lang.Exception: Nilai indeks di luar batas.
        at P15.Percobaan1.DoubleLinkedList13.get(DoubleLinkedList13.java:27)
        at P15.Percobaan1.Graph13.printGraph(Graph13.java:63)
        at P15.Percobaan1.GraphMain13.main(GraphMain13.java:15)
PS E:\KULIAH 2\Pratikum Algoritma dan Struktur Data\PrakASD_1F_13>
```

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Graph

Question :

1. Perbaiki kode program Anda apabila terdapat error atau hasil kompilasi kode tidak sesuai!
   Jawab : Kode yang diperbaiki pada Class DoubleLinked List di method remove :

```
58      public void remove(int index) {
59          Node13 current = head;
60          while (current != null) {
61              if (current.data == index) {
62                  if (current.prev != null) {
63                      current.prev.next = current.next;
64                  } else {
65                      head = current.next;
66                  }
67                  if (current.next != null) {
68                      current.next.prev = current.prev;
69                  }
70                  size--;
71                  break;
72              }
73              current = current.next;
74          }
75      }
76  }
77
```

   dan di Class Graph pada method removeEdge :

```
43      public void removeEdge(int asal, int tujuan) throws Exception {
44              list[asal].remove(tujuan);
45      }
46
```

   Outputnya menjadi :

```
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),

PS E:\KULIAH 2\Pratikum Algoritma dan Struktur Data\PrakASD_1F_13>
```

2. Pada class Graph, terdapat atribut **list[]** bertipe DoubleLinkedList. Sebutkan tujuan pembuatan variabel tersebut!
   Jawab : Untuk menyimpan bentuk adjececy list dari graph, jadi pada setiap elemen array **List[]** mewakili satu vertex pada graph.

3. Jelaskan alur kerja dari method **removeEdge**!
   Jawab : Dengan mengakses linked list yang sesuai dengan vertex “asal” dan menghapus node vertex “tujuan” dari linked list, proses penghapusan dengan menggunakan method remove pada class DoubleLinkedList yang melakukan iterasi melalui linked list, lalu menemukan node yang sesuai, dan mengupdate pointer ‘prev’ dan ;next’ untuk menghapus node dari linked list.

NAMA	: Gilang Purnomo
NIM	: 2341720042
NO ABSEN	: 13
KELAS	: 1F
MATERI	: Graph

4. Apakah alasan pemanggilan method **addFirst()** untuk menambahkan data, bukan method add jenis lain saat digunakan pada method **addEdge** pada class Graph?
   Jawab : Karena setiap vertex pasti memiliki daftar adjacency yang terurut, maka dengan penggunaan method addFirst ini dapat lebih mudah dalam melakukan traversal graf, mencari jalur graf, dan memudahkan untuk mengelola pointer.

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 :

```
67    public void adjacent(int asal, int tujuan) throws Exception {
68        for (int i = 0; i < list[asal].size(); i++) {
69            if (list[asal].get(i) == tujuan) {
70                System.out.println("Gedung " + (char) ('A' + asal) + " dan " + (char) ('A' + tujuan) + " bertetangga");
71                return;
72            }
73        }
74        System.out.println("Gedung " + (char) ('A' + asal) + " dan " + (char) ('A' + tujuan) + " tidak bertetangga");
75    }
76 }
77
```

Class Main :

```
PrakASD_1F_13 > src > P15 > Percobaan1 > J GraphMain13.java > ...
1   package P15.Percobaan1;
2
3   import java.util.Scanner;
4
    Codeium: Refactor | Explain
5   public class GraphMain13 {
        Run | Debug | Codeium: Refactor | Explain | Generate Javadoc | ×
6       public static void main(String[] args) throws Exception {
7           Scanner in = new Scanner(System.in);
8
9           Graph13 gedung = new Graph13(v:6);
10          gedung.addEdge(asal:0, tujuan:1, jarak:50);
11          gedung.addEdge(asal:0, tujuan:2, jarak:100);
12          gedung.addEdge(asal:1, tujuan:3, jarak:70);
13          gedung.addEdge(asal:2, tujuan:3, jarak:40);
14          gedung.addEdge(asal:3, tujuan:4, jarak:60);
15          gedung.addEdge(asal:4, tujuan:5, jarak:80);
16          gedung.degree(asal:0);
17          gedung.printGraph();
18          gedung.removeEdge(asal:1, tujuan:3);
19          gedung.printGraph();
20
21          System.out.print(s:"Masukkan gedung asal : ");
22          int asal = in.nextInt();
23          System.out.print(s:"Masukkan gedung tujuan : ");
24          int tujuan = in.nextInt();
25          gedung.adjacent(asal, tujuan);
26      }
27  }
28
```

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Graph

Output :

```
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),

Masukkan gedung asal : 2
Masukkan gedung tujuan : 3
Gedung C dan D bertetangga
PS E:\KULIAH 2\Pratikum Algoritma
```

```
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),

Masukkan gedung asal : 2
Masukkan gedung tujuan : 5
Gedung C dan F tidak bertetangga
PS E:\KULIAH 2\Pratikum Algoritma
```

2.2 Percobaan 2

Class Node :

PrakASD_1F_13 > src > P15 > Percobaan1 > J Node13.java > ...

```java
package P15.Percobaan1;

Codeium: Refactor | Explain
public class Node13 {
    int data;
    Node13 prev, next;
    int jarak;

    public Node13(Node13 prev, int data, int jarak, Node13 next) {
        this.prev = prev;
        this.data = data;
        this.next = next;
        this.jarak = jarak;
    }
}
```

NAMA	: Gilang Purnomo
NIM	: 2341720042
NO ABSEN	: 13
KELAS	: 1F
MATERI	: Graph

Class DoubleLinkedList :

PrakASD_1F_13 > src > P15 > Percobaan1 > DoubleLinkedList13.java > { } P15.Percobaan1

```java
package P15.Percobaan1;

Codeium: Refactor | Explain
public class DoubleLinkedList13 {
    Node13 head;
    int size;

    public DoubleLinkedList13() {
        head = null;
        size = 0;
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public boolean isEmpty() {
        return head == null;
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public int size() {
        return size;
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public void clear() {
        head = null;
        size = 0;
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public int get(int index) throws Exception {
        if (isEmpty() || index >= size) {
            throw new Exception(message:"Nilai indeks di luar batas.");
        }
        Node13 current = head;
        for (int i = 0; i < index; i++) {
            current = current.next;
        }
        return current.data;
    }

    public void addFirst(int item, int jarak) {
        if (isEmpty()) {
            head = new Node13(prev:null, item, jarak, next:null);
        } else {
            Node13 newNode = new Node13(prev:null, item, jarak, head);
            head.prev = newNode;
            head = newNode;
        }
        size++;
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public int getJarak(int index) throws Exception {
        if (isEmpty() || index >= size) {
            throw new Exception(message:"Nilai ideks di luar batas");
        }
        Node13 tmp = head;
        for (int i = 0; i < index; i++) {
            tmp = tmp.next;
        }
        return tmp.jarak;
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    public void remove(int index) {
        Node13 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;
        }
    }
}
```

NAMA	: Gilang Purnomo
NIM	: 2341720042
NO ABSEN	: 13
KELAS	: 1F
MATERI	: Graph

Class GraphMatriks :

```java
package P15.Percobaan2;

public class GraphMatriks13 {
    int vertex;
    int [][] matriks;

    public GraphMatriks13(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] = -1;
    }

    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();
        }
    }
}
```

Output :

```
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 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 E:\KULIAH 2\Pratikum Algoritma dan Struktur Data\PrakASD_1F_13>
```

Question :

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

```java
public void removeEdge(int asal, int tujuan) {
    matriks[asal][tujuan] = -1;
}
```

Mengganti matriks[asal][tujuan] = 0

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

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Graph

Output :

```
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 E:\KULIAH 2\Pratikum Algoritma dan Struktur Data\PrakASD_1F_13>
```

2. Apa jenis graph yang digunakan pada Percobaan 2?
   Jawab : Pada percobaan 2 menggunakan Graph Matriks

3. Apa maksud dari dua baris kode berikut?

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

   Jawab : 1. Untuk menambahkan lintasan/edge dari elemen 1 ke elemen 2 dengan jarak 70.
   2. Ini sebaliknya yaitu menambahkan lintasan/edge dari elemen 2 ke elemen 1 dengan jarak 80.

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

```
31
32      public void degree(int asal) throws Exception {
33          int totalIn = 0, totalOut = 0;
34          // inDegree
35          for (int i = 0; i < vertex; i++) {
36              if (matriks[asal][i] != 0) {
37                  totalIn++;
38              }
39          }
40          // outDegree
41          for (int j = 0; j < vertex; j++) {
42              if (matriks[j][asal] != 0) {
43                  totalOut++;
44              }
45          }
46          System.out.println("InDegree dari Gedung " + (char) ('A' + asal) + ": " + totalIn);
47          System.out.println("OutDegree dari Gedung " + (char) ('A' + asal) + ": " + totalOut);
48          System.out.println("Degree dari Gedung " + (char) ('A' + asal) + ": " + (totalIn + totalOut));
49          // apabila undirected
50          // System.out.println("Degree dari Gedung " + (char) ('A' + asal) + ": " + list[asal].size());
51      }
52  }
53
```

   Class Main :

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Graph

PrakASD_1F_13 > src > P15 > Percobaan2 > GraphMain13.java > ...

```java
package P15.Percobaan2;

public class GraphMain13 {
    Run | Debug
    public static void main(String[] args) throws Exception {
        GraphMatriks13 gdg = new GraphMatriks13(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();
        gdg.degree(asal:0);
    }
}

```

Output :

```
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),
InDegree dari Gedung A: 1
OutDegree dari Gedung A: 2
Degree dari Gedung A: 3
PS E:\KULIAH 2\Pratikum Algoritma dan Struktur Data\PrakASD_1F_13>
```

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Graph

3. Latihan Praktikum

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

a) Add Edge

b) Remove Edge

c) Degree

d) Print Graph

e) Cek Edge

Pengguna dapat memilih menu program melalui input Scanner

Jawab :

Penambahan method cekEdge

```
76
77      public void cekEdge(int asal, int tujuan) throws Exception {
78          for (int i = 0; i < list[asal].size(); i++) {
79              if (list[asal].get(i) == tujuan) {
80                  System.out.println("Jarak antar gedung " + (char) ('A' + asal) + " dan gedung " + (char) ('A' + tujuan) + " : " + list[asal].getJarak(i) + " m");
81                  return;
82              }
83          }
84          System.out.println("Tidak ada jarak antar gedung " + (char) ('A' + asal) + " dan gedung " + (char) ('A' + tujuan));
85      }
86  }
87
```

Class Main :

```
1   package P15.Percobaan1;
2
3   import java.util.Scanner;
4
5   public class GraphMain13 {
6       public static void menu() {
7           System.out.println(x:"---------------------");
8           System.out.println(x:"Menu");
9           System.out.println(x:"---------------------");
10          System.out.println(x:"1. Add Edge");
11          System.out.println(x:"2. Remove Edge");
12          System.out.println(x:"3. Degree");
13          System.out.println(x:"4. Print Graph");
14          System.out.println(x:"5. Cek Edge");
15          System.out.println(x:"6. Exit");
16          System.out.println(x:"---------------------");
17      }
        Run | Debug
18      public static void main(String[] args) throws Exception {
19          Scanner in = new Scanner(System.in);
20          Graph13 gedung = new Graph13(v:7);
21          gedung.addEdge(asal:0, tujuan:1, jarak:50);
22          gedung.addEdge(asal:0, tujuan:2, jarak:100);
23          gedung.addEdge(asal:1, tujuan:3, jarak:70);
24          gedung.addEdge(asal:2, tujuan:3, jarak:40);
25
```

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Graph

```
        while (true) {
            menu();
            int pilih = in.nextInt();

            switch (pilih) {
                case 1:
                    System.out.println(x:"Masukkan Gedung asal");
                    int asal = in.nextInt();
                    System.out.println(x:"Masukkan Gedung tujuan");
                    int tujuan = in.nextInt();
                    System.out.println(x:"Masukkan jumlah jarak");
                    int jarak = in.nextInt();
                    gedung.addEdge(asal, tujuan, jarak);
                    break;
                case 2:
                    System.out.println(x:"Masukkan Gedung asal");
                    asal = in.nextInt();
                    System.out.println(x:"Masukkan Gedung tujuan");
                    tujuan = in.nextInt();
                    gedung.removeEdge(asal, tujuan);
                    break;
                case 3:
                    System.out.println(x:"Masukkan gedung untuk melihat sisi yang bersebelahan : ");
                    asal = in.nextInt();
                    gedung.degree(asal);
                    break;
                case 4:
                    gedung.printGraph();
                    break;
                case 5:
                    System.out.println(x:"Masukkan Gedung asal");
                    asal = in.nextInt();
                    System.out.println(x:"Masukkan Gedung tujuan");
                    tujuan = in.nextInt();
                    gedung.cekEdge(asal, tujuan);
                    break;
                case 6:
                    System.exit(status:0);
                default:
                    System.out.println(x:"Pilihan tidak tersedia");
                    break;
            }
        }
    }
}
```

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Graph

Output :

Add

```
---------------------
Menu
---------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Exit
---------------------
1
Masukkan Gedung asal
4
Masukkan Gedung tujuan
2
Masukkan jumlah jarak
90
---------------------
Menu
---------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Exit
---------------------
4
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 E terhubung dengan
C (90 m),
```

Remove

```
---------------------
Menu
---------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Exit
---------------------
2
Masukkan Gedung asal
0
Masukkan Gedung tujuan
2
---------------------
Menu
---------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Exit
---------------------
4
Gedung A terhubung dengan
B (50 m),
Gedung B terhubung dengan
D (70 m),
Gedung C terhubung dengan
D (40 m),
Gedung E terhubung dengan
C (90 m),
```

Print

```
----------------------
Menu
----------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Exit
----------------------
4
Gedung A terhubung dengan
B (50 m),
Gedung B terhubung dengan
D (70 m),
Gedung C terhubung dengan
D (40 m),
Gedung E terhubung dengan
C (90 m),
```

Degree

```
---------------------
Menu
---------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Exit
---------------------
3
Masukkan gedung untuk melihat sisi yang bersebelahan :
4
InDegree dari Gedung E: 0
OutDegree dari Gedung E: 1
Degree dari Gedung E: 1
---------------------
Menu
---------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Exit
---------------------
3
Masukkan gedung untuk melihat sisi yang bersebelahan :
2
InDegree dari Gedung C: 1
OutDegree dari Gedung C: 1
Degree dari Gedung C: 2
```

CekEdge

```
---------------------
Menu
---------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Exit
---------------------
5
Masukkan Gedung asal
1
Masukkan Gedung tujuan
3
Jarak antar gedung B dan gedung D : 70 m
---------------------
Menu
---------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Exit
---------------------
5
Masukkan Gedung asal
1
Masukkan Gedung tujuan
0
Tidak ada jarak antar gedung B dan gedung A
```

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Graph

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

Jawab :

Penambahan method di DoubleLinkedList :

```
77    public void jarak(int tujuan, int jarakBaru) throws Exception {
78        Node13 current = head;
79        while (current != null) {
80            if (current.data == tujuan) {
81                current.jarak = jarakBaru;
82                return;
83            }
84            current= current.next;
85        }
86    }
87 }
88
```

Penambahan method di Graph :

```
87    public void updateJarak(int asal, int tujuan, int jarakBaru) throws Exception {
88        list[asal].jarak(tujuan, jarakBaru);
89    }
```

Sebelum update

```
----------------------
Menu
----------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Update Jarak
7. Exit
----------------------
4
Gedung A terhubung dengan
C (100 m), B (50 m),
Gedung B terhubung dengan
D (70 m),
Gedung C terhubung dengan
D (40 m),
```

Melakukan update

```
----------------------
Menu
----------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Update Jarak
7. Exit
----------------------
6
Masukkan Gedung asal
1
Masukkan Gedung tujuan
3
Masukkan jumlah jarak
13
```

Sesudah update

```
----------------------
Menu
----------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Update Jarak
7. Exit
----------------------
4
Gedung A terhubung dengan
C (100 m), B (50 m),
Gedung B terhubung dengan
D (13 m),
Gedung C terhubung dengan
D (40 m),
```

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Graph

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

Jawab :

Method :

```
91      public int hitungEdge() {
92          int jmlEdge = 0;
93          for (int i = 0; i < vertex; i++) {
94              jmlEdge += list[i].size();
95          }
96          return jmlEdge;
97      }
98  }
```

Class main :

```
74                  case 7:
75                      int total = gedung.hitungEdge();
76                      System.out.println("Total Edge : " + total);
77                      break;
```

Output :

```
----------------------
Menu
----------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Update Jarak
7. Hitung Edge
8. Exit
----------------------
7
Total Edge : 4
```

```
----------------------
Menu
----------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Update Jarak
7. Hitung Edge
8. Exit
----------------------
1
Masukkan Gedung asal
1
Masukkan Gedung tujuan
2
Masukkan jumlah jarak
13
----------------------
Menu
----------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Update Jarak
7. Hitung Edge
8. Exit
----------------------
4
Gedung A terhubung dengan
C (100 m), B (50 m),
Gedung B terhubung dengan
C (13 m), D (70 m),
Gedung C terhubung dengan
D (40 m),
```

```
----------------------
Menu
----------------------
1. Add Edge
2. Remove Edge
3. Degree
4. Print Graph
5. Cek Edge
6. Update Jarak
7. Hitung Edge
8. Exit
----------------------
7
Total Edge : 5
```