JOBSHEET 15
GRAPH

NAMA : DWI NUR OKTAVIANI
NIM (ABSEN) : 1941720239 (09)
KELAS : TI – 1F
TANGGAL PRAKTIKUM : 14 MEI 2020

PRAKTIKUM JOBSHEET 15

1. PRAKTIKUM 1

- **Source Code**

No	Program Node.java
1	
2	`package JOBSHEET14;`
3	
4	`public class Node {`
5	`    int data;`
6	`    Node prev, next;`
7	
8	`    Node(Node prev, int data, Node next){`
9	`        this.prev = prev;`
10	`        this.data = data;`
11	`        this.next = next;`
12	`    }`
13	`}`

No	Program LinkedList.java
1	
2	`package JOBSHEET14;`
3	
4	`public class LinkedList {`
5	`        Node head;`
6	`    int size;`
7	
8	`    public LinkedList(){`
9	`        head = null;`
10	`        size = 0;`
11	`    }`
12	
13	`    public boolean isEmpty(){`
14	`        return head == null;`
15	`    }`
16	
17	`    public void addFirst(int item){`
18	`        if(isEmpty())`
19	`            head = new Node(null, item, null);`
20	`        else{`
21	`            Node newNode = new Node(null, item, head);`
22	`            head.prev = newNode;`
23	`            head = newNode;`
24	`        }`
25	`        size++;`
26	`    }`
27	
28	`    public void addLast(int item){`

JOBSHEET 15
GRAPH

NAMA : DWI NUR OKTAVIANI
NIM (ABSEN) : 1941720239 (09)
KELAS : TI – 1F
TANGGAL PRAKTIKUM : 14 MEI 2020

```
29          if(isEmpty())
30              addFirst(item);
31          else{
32              Node current = head;
33              while(current.next != null){
34                  current = current.next;
35              }
36              Node newNode = new Node(current, item, null);
37              current.next = newNode;
38              size++;
39          }
40      }
41
42      public void add(int item, int index) throws Exception{
43          if(isEmpty()) addFirst(item);
44          if(index < 0 || index > size) throw new
45  Exception("Nilai index di luar batas");
46          else{
47              Node current = head;
48              int i = 0;
49              while(i < index){
50                  current = current.next;
51                  i++;
52              }
53              if(current.prev == null){
54                  Node newNode = new Node(null, item,
55  current);
56                  current.prev = newNode;
57                  head = newNode;
58              }
59              else{
60                  Node newNode = new Node(current.prev, item,
61  current);
62                  newNode.prev = current.prev;
63                  newNode.next = current;
64                  current.prev.next = newNode;
65                  current.prev = newNode;
66              }
67          }
68          size++;
69      }
70
71      public int size(){
72          return size;
73      }
74
75      public void clear(){
76          head = null;
77          size = 0;
78      }
79
```

JOBSHEET 15
GRAPH

NAMA : DWI NUR OKTAVIANI
NIM (ABSEN) : 1941720239 (09)
KELAS : TI – 1F
TANGGAL PRAKTIKUM : 14 MEI 2020

```
80      public void print(){
81          if(!isEmpty()){
82              Node tmp = head;
83              while(tmp != null){
84                  System.out.println(tmp.data + "\t");
85                  tmp = tmp.next;
86              }
87              System.out.println("Berhasil diisi");
88          }
89          else{
90              System.out.println("Linked list kosong");
91          }
92      }
93
94      public void removeFirst() throws Exception{
95          if(isEmpty()) throw new Exception("Linked List
96  masih Kosong, tidak dapat dihapus");
97          else if(size == 1){
98              removeLast();
99          }
100         else{
101             head = head.next;
102             head.prev = null;
103             size--;
104         }
105     }
106
107     public void removeLast() throws Exception{
108         if(isEmpty()) throw new Exception("Linked List
109 masih Kosong, tidak dapat dihapus");
110         if(head.next == null){
111             head = null;
112             size--;
113             return;
114         }
115         Node current = head;
116         while(current.next.next != null){
117             current = current.next;
118         }
119         current.next = null;
120         size--;
121     }
122
123     public void remove(int index) throws Exception{
124         if(isEmpty() || index >= size) throw new
125 Exception("Nilai indeks di luar batas");
126         if(index == 0){
127             removeFirst();
128         }
129         else{
130             Node current = head;
```

JOBSHEET 15
GRAPH

NAMA : DWI NUR OKTAVIANI
NIM (ABSEN) : 1941720239 (09)
KELAS : TI – 1F
TANGGAL PRAKTIKUM : 14 MEI 2020

```
131             int i = 0;
132             while(i < index){
133                 current = current.next;
134                 i++;
135             }
136             if(current.next == null){
137                 current.prev.next = null;
138             }
139             else if(current.prev.next == null){
140                 current = current.next;
141                 current.prev = null;
142                 head = current;
143             }
144             else{
145                 current.prev.next = current.next;
146                 current.next.prev = current.prev;
147             }
148             size--;
149         }
150     }
151
152     public int getFirst() throws Exception{
153         if(isEmpty()) throw new Exception("Linkes List
154 Kosong");
155         return head.data;
156     }
157
158     public int getLast() throws Exception{
159         if(isEmpty()) throw new Exception("Linked List
160 Kosong");
161         Node tmp = head;
162         while(tmp.next != null){
163             tmp = tmp.next;
164         }
165         return tmp.data;
166     }
167
168     public int get(int index) throws Exception{
169         if(isEmpty() || index>=size) throw new
170 Exception("Nilai index di luar batas");
171         Node tmp = head;
172         for(int i = 0; i < index; i++){
173             tmp = tmp.next;
174         }
175         return tmp.data;
176     }
177
178 }
```

JOBSHEET 15
GRAPH

NAMA : DWI NUR OKTAVIANI
NIM (ABSEN) : 1941720239 (09)
KELAS : TI – 1F
TANGGAL PRAKTIKUM : 14 MEI 2020

No	Program Graph.java
1	
2	`package JOBSHEET14;`
3	
4	`public class Graph {`
5	`    int vertex;`
6	`    LinkedList list[];`
7	
8	`    public Graph(int vertex){`
9	`        this.vertex = vertex;`
10	`        list = new LinkedList[vertex];`
11	`        for(int i = 0; i < vertex; i++){`
12	`            list[i] = new LinkedList();`
13	`        }`
14	`    }`
15	
16	`    public void addEdge(int source, int destination){`
17	`        list[source].addFirst(destination);`
18	`        list[destination].addFirst(source);`
19	`    }`
20	
21	`    public void degree(int source) throws Exception{`
22	`        System.out.println("degree vertex " + source + " : "`
23	`+ list[source].size());`
24	
25	`        int k, totalIn = 0, totalOut = 0;`
26	`        for(int i = 0; i < vertex; i++){`
27	`            for(int j = 0; j < list[i].size(); j++){`
28	`                if(list[i].get(j) == source)`
29	`                    ++totalIn;`
30	`            }`
31	
32	`        for(k = 0; k < list[source].size(); k++){`
33	`            list[source].get(k);`
34	`        }`
35	
36	`        totalOut = k;`
37	`        }`
38	
39	`        System.out.println("Indegree dari vertex " + source`
40	`+ " : " + totalIn);`
41	`        System.out.println("Outdegree dari vertex " + source`
42	`+ " : " + totalOut);`
43	`        System.out.println("degree vertex " + source + " : "`
44	`+ (totalIn + totalOut));`
45	`    }`
46	
47	`    public void removeEdge(int source, int destination)`
48	`throws Exception {`
49	`        for(int i = 0; i< vertex; i++){`
50	`            if(i == destination){`

JOBSHEET 15
GRAPH

NAMA : DWI NUR OKTAVIANI
NIM (ABSEN) : 1941720239 (09)
KELAS : TI – 1F
TANGGAL PRAKTIKUM : 14 MEI 2020

```
51                  list[source].remove(destination);
52              }
53          }
54      }
55
56      public void removeAllEdges(){
57          for(int i = 0; i < vertex; i++){
58              list[i].clear();
59          }
60          System.out.println("Graph berhasil dikosongkan");
61      }
62
63      public void printGraph() throws Exception{
64          for(int i = 0; i < vertex; i++){
65              if(list[i].size() > 0){
66                  System.out.println("Vertex " + i + "
67  terhubung dengan : ");
68                  for(int j = 0; j < list[i].size(); j++){
69                      System.out.print(list[i].get(j) + " ");
70                  }
71                  System.out.println(" ");
72              }
73          }
74          System.out.println(" ");
75      }
76
77      public static void main(String[] args) throws Exception{
78          Graph graph = new Graph(6);
79          graph.addEdge(0, 1);
80          graph.addEdge(0, 4);
81          graph.addEdge(1, 2);
82          graph.addEdge(1, 3);
83          graph.addEdge(1, 4);
84          graph.addEdge(2, 3);
85          graph.addEdge(3, 4);
86          graph.addEdge(3, 0);
87          graph.printGraph();
88          graph.degree(2);
89          graph.removeEdge(1, 2);
90          graph.printGraph();
91      }
92
93  }
```

JOBSHEET 15
GRAPH

NAMA : DWI NUR OKTAVIANI
NIM (ABSEN) : 1941720239 (09)
KELAS : TI – 1F
TANGGAL PRAKTIKUM : 14 MEI 2020

- **Output**

Tanpa remove

```
run:
Vertex 0 terhubung dengan :
3 4 1
Vertex 1 terhubung dengan :
4 3 2 0
Vertex 2 terhubung dengan :
3 1
Vertex 3 terhubung dengan :
0 4 2 1
Vertex 4 terhubung dengan :
3 1 0

degree vertex 2 : 2
Indegree dari vertex 2 : 2
Outdegree dari vertex 2 : 2
degree vertex 2 : 4
BUILD SUCCESSFUL (total time: 4 seconds)
```

Dengan remove

```
run:
Vertex 0 terhubung dengan :
3 4 1
Vertex 1 terhubung dengan :
4 3 2 0
Vertex 2 terhubung dengan :
3 1
Vertex 3 terhubung dengan :
0 4 2 1
Vertex 4 terhubung dengan :
3 1 0

degree vertex 2 : 2
Indegree dari vertex 2 : 2
Outdegree dari vertex 2 : 2
degree vertex 2 : 4
Vertex 0 terhubung dengan :
3 4 1
Vertex 1 terhubung dengan :
4 3 0
Vertex 2 terhubung dengan :
3 1
Vertex 3 terhubung dengan :
0 4 2 1
Vertex 4 terhubung dengan :
3 1 0

BUILD SUCCESSFUL (total time: 7 seconds)
```

2. PRAKTIKUM 2

- **Source Code**

No	Program graphArray.java
1	
2	`package JOBSHEET14;`
3	
4	`public class graphArray {`
5	`    private final int vertices;`
6	`    private int[][] twoD_array;`
7	
8	`    public graphArray(int v){`
9	`        vertices = v;`
10	`        twoD_array = new int[vertices + 1][vertices + 1];`
11	`    }`
12	

JOBSHEET 15
GRAPH

NAMA : DWI NUR OKTAVIANI
NIM (ABSEN) : 1941720239 (09)
KELAS : TI – 1F
TANGGAL PRAKTIKUM : 14 MEI 2020

```
13      public void makeEdge(int to, int from, int edge){
14          try{
15              twoD_array[to][from] = edge;
16          }
17          catch (ArrayIndexOutOfBoundsException index)
18          {
19              System.out.println("Vartex tidak ada");
20          }
21      }
22
23      public int getEdge(int to, int from){
24          try{
25              return twoD_array[to][from];
26          }
27          catch (ArrayIndexOutOfBoundsException index)
28          {
29              System.out.println("Vartex tidak ada");
30          }
31          return -1;
32      }
33
34  }
```

No	Program main1.java

```
1
2   package JOBSHEET14;
3   import java.util.Scanner;
4
5   public class main1 {
6       public static void main(String[] args){
7           int v, e, count = 1, to = 0, from = 0;
8           Scanner sc = new Scanner(System.in);
9           graphArray graph;
10
11          try
12          {
13              System.out.println("Masukkan jumlah vertices :
14  ");
15              v = sc.nextInt();
16              System.out.println("Masukkan jumlah edges : ");
17              e = sc.nextInt();
18
19              graph = new graphArray(v);
20
21              System.out.println("Masukkan edges : <to>
22  <from> ");
23              while(count <= e)
24              {
25                  to = sc.nextInt();
26                  from = sc.nextInt();
27
```

JOBSHEET 15
GRAPH

NAMA : DWI NUR OKTAVIANI
NIM (ABSEN) : 1941720239 (09)
KELAS : TI – 1F
TANGGAL PRAKTIKUM : 14 MEI 2020

```
28                  graph.makeEdge(to, from, 1);
29                  count++;
30              }
31
32              System.out.println("Array 2D sebagai
33  representasi graph sbb : ");
34              System.out.print(" ");
35
36              for(int i = 1; i <= v; i++)
37                  System.out.print(i + " ");
38              System.out.println();
39
40              for(int i = 1; i <= v; i++){
41                  System.out.print(i + " ");
42                  for(int j = 1; j <= v; j++)
43                      System.out.print(graph.getEdge(i, j) +
44  " ");
45                  System.out.println();
46              }
47          }
48
49          catch(Exception E){
50              System.out.println("Error. Silahkan cek
51  kembali");
52          }
53          sc.close();
54      }
55  }
```

- **Outpu**

```
run:
Masukkan jumlah vertices :
5
Masukkan jumlah edges :
6
Masukkan edges : <to> <from>
1 2
1 5
2 3
2 4
2 5
3 4
Array 2D sebagai representasi graph sbb :
 1 2 3 4 5
1 0 1 0 0 1
2 0 0 1 1 1
3 0 0 0 1 0
4 0 0 0 0 0
5 0 0 0 0 0
BUILD SUCCESSFUL (total time: 55 seconds)
```

JOBSHEET 15
GRAPH

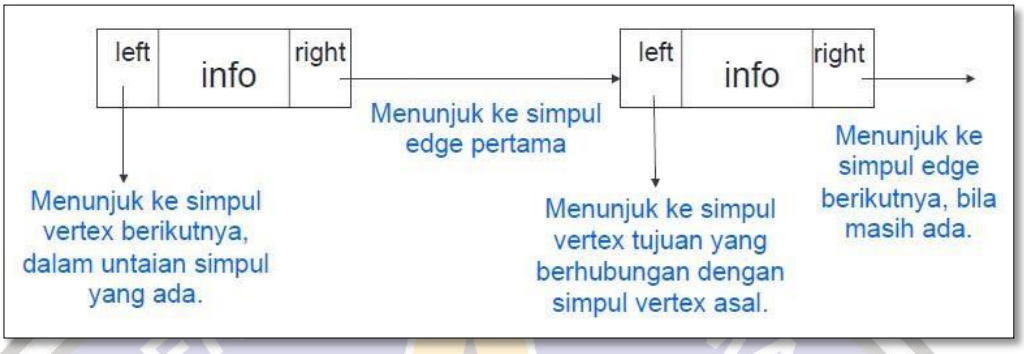
NAMA	: DWI NUR OKTAVIANI
NIM (ABSEN)	: 1941720239 (09)
KELAS	: TI – 1F
TANGGAL PRAKTIKUM	: 14 MEI 2020

PERTANYAAN

1. Mengapa graph diimplementasikan dengan double linked list, bukan single linked list?

   JAWAB :

   Graph diimplementasikan dengan double linked list karena graph memiliki simpul awal (vertex awal) dan simpul tujuan (vertex akhir).



Representasi Graph *dalam bentuk Linked List*

2. Jelaskan masing-masingpada dua jenis looping yang terdapat dalam metod **printGraph()**!

   JAWAB :

   - Pada looping for pertama untuk mengeluarkan atau menampilkan vertex awal dalam method printGraph().
   - Pada looping for kedua atau nested loop nya adalah untuk menampilkan vertex yang terhubung dengan vertex destination.

3. Apakah perbedaan Graph dengan Binary Tree pada implementasi nya menggunakan linked list?

   JAWAB :

   - Pada binary tree digunakan node kiri pada linked list untuk membandingkan nilai yang lebih kecil atau nilai yang lebih besar pada data sebelumnya
   - Pada graph digunakan node linked list yang berfungsi untuk menghubungkan vertex sesuai dengan edge nya

4. Jelaskan dengan contoh perbedaan antara Edge dan Path pada graph!

   JAWAB

JOBSHEET 15
GRAPH

NAMA : DWI NUR OKTAVIANI
NIM (ABSEN) : 1941720239 (09)
KELAS : TI – 1F
TANGGAL PRAKTIKUM : 14 MEI 2020

- Lintasan (*Edge*)
  Garis-garis penghubung antar simpul dalam *graph* disebut dengan lintasan (*edge*).

- Lintasan (*Path*)
  Lintasan (*path*) adalah urutan dari lintasan (*edge*). Contohnya lintasan J ke P, yang dapat kita sebut sebagai lintasan (*path*) JBCP adalah lintasan (*path*) dari simpul J ke P.

5. Sebutkan beberapa contoh (minimal 3) implementasi graph dalam permasalahan yang membutuhkan representasi internal dalam memori komputer untuk suatu struktur data!
   JAWAB :
   - Graf berarah (Directed Graph)
   - Graf Tak Berarah (Undirected Graph)
   - Graf Berbobot (Weighted Graph)

6. Sebutkan beberapa jenis (minimal 3) algoritma yang menggunakan dasar Graph, dan apakah kegunaan algoritma-algoritma tersebut?
   JAWAB :
   - Algoritma Dijkstra
     Algoritma yang digunakan untuk menentukan lintasan terpendek pada graf berbobot (weighted graph)
   - Algoritma Greedy
     Algoritma yang diterapkan dalam mencari lintasan terpendek dengan hasil optimum. Didasarkan pada pemindahan edge per edge.
   - Algoritma Kruskal
     Algoritma yang digunakan untuk menentukan pohon perentang (spaning tree) terbobot minimum.

7. Pada **class Graph** terdapat array bertipe LinkedList, yaitu **LinkedList list[].** Apakah tujuan pembuatan variabel tersebut?
   JAWAB :
   Tujuan pembuatan array bertipe linkedlist() pada class Graph adalah untuk membuat array bertipe data node.

JOBSHEET 15
GRAPH

NAMA : DWI NUR OKTAVIANI
NIM (ABSEN) : 1941720239 (09)
KELAS : TI – 1F
TANGGAL PRAKTIKUM : 14 MEI 2020

8. Apakah alasan pemanggilan method **addFirst()** untuk menambahkan data, bukan method add jenis lain pada linked list ketika digunakan pada method addEdge pada **class Graph?**

   JAWAB :

   Pemanggilan method addFirst() adalah untuk menggunakan fungsi addFirst() pada addEdge(). Sehingga pada graph saat menggunkaan method add() harus memasukkan index, namun jka menggunakan addFirst() hanya memasukkan item nya saja.

9. Bagaimana cara mendeteksi prev pointer pada saat akan melakukan penghapusan suatu edge pada graph?

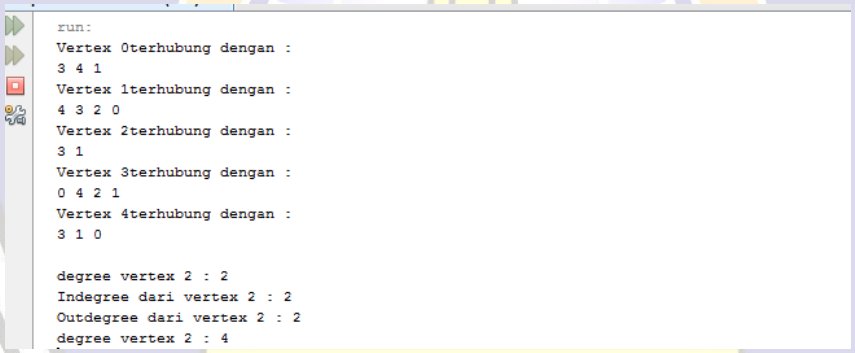
   JAWAB :

   Mendeteksi prev pointer pada penghapusan edge dalam graph dengan dilakukan proses looping, setelah itu baru dilakukan proses remove.

10. Kenapa pada praktikum 15.3 langkah ke-12 untuk menghapus path yang bukan merupakan lintasan pertama kali menghasilkan output yang salah ? Bagaimana solusinya ?

```
89          graph.removeEdge(1, 3);
90          graph.printGraph();
```

JAWAB :



JOBSHEET 15
GRAPH

NAMA : DWI NUR OKTAVIANI
NIM (ABSEN) : 1941720239 (09)
KELAS : TI – 1F
TANGGAL PRAKTIKUM : 14 MEI 2020

PERTANYAAN

1. Apakah perbedaan degree/derajat pada *directed* dan *undirected graph*?
   JAWAB :

   - *Directed graph*
     Indegree pada *graph* adalah jumlah busur yang kepalanya incident dengan simpul tersebut, atau jumlah busur yang "masuk" atau menuju simpul tersebut.
     Outdegree pada graph adalah jumlah busur yang ekornya incident dengan simpul tersebut, atau jumlah busur yang "keluar" atau berasal dari simpul tersebut.
   - *Undirected graph*
     Simpul yang dimiliki simpul satu dengan lainnya mempunyai indegree dan outdegree

2. Pada implementasi graph menggunakan adjecency matriks. Kenapa jumlah vertices harus ditambahkan dengan 1 pada indeks array berikut?

```
8       public graphArray(int v)
9       {
10          vertices = v;
11          twoD_array = new int[vertices + 1][vertices + 1];
12      }
```

   JAWAB :
   Karena input yang dimasukkan dimulai dari 1.

3. Apakah kegunaan method getEdge()
   JAWAB :
   Method getEdge() berfungsi untuk menampilkan nilai dari edge ada atau tidak

4. **Termasuk jenis graph apakah uji coba pada praktikum 15.4 ?**
   JAWAB :
   Pada praktikum 2 termasuk dalam representasi graph *adjecency matrik*

JOBSHEET 15
GRAPH

NAMA	: DWI NUR OKTAVIANI
NIM (ABSEN)	: 1941720239 (09)
KELAS	: TI – 1F
TANGGAL PRAKTIKUM	: 14 MEI 2020