

Nama : Frederico Steven Kwok

NPM : 242310037

Kelas : TI-24-PA

Matkul : Lab. Desain dan Analisis Algoritma

Github : https://github.com/RicoSteven120206/Desain_Analisis_Algoritma.git

Tugas : Membuat program C++ dengan konsep Graph

A. Implementasi Graph dengan directed, undirected dan weighted graph

1. Screenshot Source Code

```
1  #include <iostream>
2  #include <vector>
3  #include <cstdlib>
4  using namespace std;
5
6  class Graph {
7  private:
8      int V;
9      vector<vector<int>> > adj; // Adjacency list
10
11  public:
12      Graph (int V) {
13          this->V = V;
14          adj.resize(V); // Manipulasi ukuran pada simpul
15      }
16
17      void addEdgeUndir(int u, int v) {
18          adj[u].push_back(v); // Add edge dari u ke v
19          adj[v].push_back(u); // Add edge dari v ke u (untuk undirected)
20      }
21
22      void addEdgeDir(int u, int v) {
23          adj[u].push_back(v); // Add edge dari u ke v
24      }
25
26      void printGraph() {
27          for (int u = 0; u < V; u++) {
28              cout << u << " -> ";
29              for (vector<int>::iterator it = adj[u].begin(); it != adj[u].end(); ++it) {
30                  cout << *it << " ";
31              }
32              cout << endl;
33          }
34      }
35  };
36
37
```

```

38 class graphWeight {
39     private:
40         int V;
41         vector<vector<pair<int, int > > > adj; //Adjacency List
42     public:
43         graphWeight (int V) {
44             this->V = V;
45             adj.resize(V); // Manipulasi ukuran pada simpul
46         }
47
48         void addEdgeWeiUndir(int u, int v, int weight) {
49             adj[u].push_back(make_pair(v, weight)); // Add edge dari u ke v
50             adj[v].push_back(make_pair(u, weight)); // Add edge dari u ke v (untuk undirected)
51         }
52
53         void addEdgeWeiDir(int u, int v, int weight) {
54             adj[u].push_back(make_pair(v, weight)); // Add edge dari u ke v
55         }
56
57         void printGraphWeight() {
58             for (int u = 0; u < V; u++) {
59                 cout << u << " -> ";
60                 for (vector<pair<int, int> >::iterator it = adj[u].begin(); it != adj[u].end(); ++it) {
61                     cout << "(" << it->first << ", " << it->second << ") ";
62                 }
63                 cout << endl;
64             }
65         }
66     };
67
68 int main() {
69     int n, pilih, v1, v2, w;
70     string data;
71
72     ulang:
73     cout<<endl;
74     cout<<"Masukkan jumlah edge pada vertex: ";
75     cin>>n;
76     Graph graph(n);
77     graphWeight gw(n);
78
79     cout<<"IMPLEMENTAION GRAPH"<<endl;
80     cout<<"1. Undirected Graph"<<endl;
81     cout<<"2. Directed Graph"<<endl;
82     cout<<"3. Undirected Weighted Graph"<<endl;
83     cout<<"4. Directed Weighted Graph"<<endl;
84     cout<<"Pilih salah satu: ";
85     cin>>pilih;
86
87     if(pilih == 1) {
88         for(int i = 0; i < n; i++) {
89             cout<<"Masukkan Simpul Asal : ";
90             cin>>v1;
91             cout<<"Masukkan Simpul Tujuan : ";
92             cin>>v2;
93             graph.addEdgeUndir(v1, v2);
94         }

```

```

95     } else if(pilih == 2) {
96         for(int i = 0; i < n; i++){
97             cout<<"Masukkan Simpul Asal : ";
98             cin>>v1;
99             cout<<"Masukkan Simpul Tujuan : ";
100            cin>>v2;
101            graph.addEdgeDir(v1, v2);
102        }
103    } else if(pilih == 3) {
104        for(int i = 0; i < n; i++) {
105            cout<<"Masukkan Simpul Asal : ";
106            cin>>v1;
107            cout<<"Masukkan Simpul Tujuan : ";
108            cin>>v2;
109            cout<<"Masukkan Beban : ";
110            cin>>w;
111            gw.addEdgeWeiUndir(v1, v2, w);
112        }
113    } else if(pilih == 4) {
114        for(int i = 0; i < n; i++) {
115            cout<<"Masukkan Simpul Asal : ";
116            cin>>v1;
117            cout<<"Masukkan Simpul Tujuan : ";
118            cin>>v2;
119            cout<<"Masukkan Beban : ";
120            cin>>w;
121            gw.addEdgeWeiDir(v1, v2, w);
122        }
123    }
124
125    if (pilih == 1 || pilih == 2) {
126        graph.printGraph();
127    } else if (pilih == 3 || pilih == 4) {
128        gw.printGraphWeight();
129    }
130
131    char yes;
132    cout<<"Apakah ingin melanjutkan program ini? (Y/N)? ";
133    cin>>yes;
134
135    if (yes == 'y' || yes == 'Y') {
136        system("cls");
137        goto ulang;
138    } else {
139        cout<<"Program Selesai..."<<endl;
140    }
141
142    return 0;
143 }

```

2. Screenshot Output

1. Undirected Graph

```
Masukkan jumlah edge pada vertex: 4
IMPLEMENTAION GRAPH
1. Undirected Graph
2. Directed Graph
3. Undirected Weighted Graph
4. Directed Weighted Graph
Pilih salah satu: 1
Masukkan Simpul Asal : 0
Masukkan Simpul Tujuan : 1
Masukkan Simpul Asal : 0
Masukkan Simpul Tujuan : 1
Masukkan Simpul Asal : 1
Masukkan Simpul Tujuan : 2
Masukkan Simpul Asal : 2
Masukkan Simpul Tujuan : 3
0 -> 1 1
1 -> 0 0 2
2 -> 1 3
3 -> 2
Apakah ingin melanjutkan program ini? (Y/N)?
```

2. Directed Graph

```
Masukkan jumlah edge pada vertex: 4
IMPLEMENTAION GRAPH
1. Undirected Graph
2. Directed Graph
3. Undirected Weighted Graph
4. Directed Weighted Graph
Pilih salah satu: 2
Masukkan Simpul Asal : 0
Masukkan Simpul Tujuan : 1
Masukkan Simpul Asal : 1
Masukkan Simpul Tujuan : 2
Masukkan Simpul Asal : 2
Masukkan Simpul Tujuan : 3
Masukkan Simpul Asal : 3
Masukkan Simpul Tujuan : 1
0 -> 1
1 -> 2
2 -> 3
3 -> 1
Apakah ingin melanjutkan program ini? (Y/N)?
```

3. Undirected Weighted Graph

```
Masukkan jumlah edge pada vertex: 4
IMPLEMENTAION GRAPH
1. Undirected Graph
2. Directed Graph
3. Undirected Weighted Graph
4. Directed Weighted Graph
Pilih salah satu: 3
Masukkan Simpul Asal : 0
Masukkan Simpul Tujuan : 1
Masukkan Beban : 3
Masukkan Simpul Asal : 0
Masukkan Simpul Tujuan : 3
Masukkan Beban : 9
Masukkan Simpul Asal : 1
Masukkan Simpul Tujuan : 3
Masukkan Beban : 6
Masukkan Simpul Asal : 1
Masukkan Simpul Tujuan : 2
Masukkan Beban : 4
0 -> (1, 3) (3, 9)
1 -> (0, 3) (3, 6) (2, 4)
2 -> (1, 4)
3 -> (0, 9) (1, 6)
Apakah ingin melanjutkan program ini? (Y/N)?
```

4. Directed Weighted Graph

```
Masukkan jumlah edge pada vertex: 4
IMPLEMENTAION GRAPH
1. Undirected Graph
2. Directed Graph
3. Undirected Weighted Graph
4. Directed Weighted Graph
Pilih salah satu: 4
Masukkan Simpul Asal : 0
Masukkan Simpul Tujuan : 1
Masukkan Beban : 4
Masukkan Simpul Asal : 0
Masukkan Simpul Tujuan : 2
Masukkan Beban : 6
Masukkan Simpul Asal : 1
Masukkan Simpul Tujuan : 2
Masukkan Beban : 5
Masukkan Simpul Asal : 2
Masukkan Simpul Tujuan : 3
Masukkan Beban : 7
0 -> (1, 4) (2, 6)
1 -> (2, 5)
2 -> (3, 7)
3 ->
Apakah ingin melanjutkan program ini? (Y/N)?
```

3. Screenshot jam pengerjaan

01.32
26/06/2025

4

B. Buatlah program C++ yang menghasilkan output seperti di modul hal. 77

1. Screenshot Source Code

```
1  #include <bits/stdc++.h>
2  #define MAX 100005
3  #define INF INT_MAX
4  using namespace std;
5
6  int dist[MAX]; // deklarasi dsitance
7  vector<pair<int, int> > adj[MAX]; // deklarasi adjacency dengan vector
8  bool vis[MAX]; // deklarasi visited dengan boolean
9  int route[MAX]; // deklarasi route
10
11 class Graph{
12 public:
13     // membuat function dijkstra
14     void dijkstra(int start) {
15         memset(vis, false, sizeof vis); // membuat set memori di suatu blok memori
16         for (int i = 0; i < MAX; i++) {
17             dist[i] = INF; // INF akan menyimpan data ke distance dengan array i
18         }
19
20         dist[start] = 0; // distance dengan array start dari 0
21         priority_queue<pair<int, int>, vector<pair<int, int> >, greater<pair<int, int> > > pq; // deklarasi pq dengan vector
22         pq.push({0, start}); // variabel pq akan push suatu value dengan parameter 0 dan start
23
24         // akan melakukan looping jika variabel pq nya tidak kosong
25         while(!pq.empty()) {
26             pair<int, int> p = pq.top(); // deklarasi variabel p dengan vector dengan variabel pq
27             pq.pop(); // akan melakukan pop pada variabel pq
28             int x = p.second; // variabel p pada bagian kedua/second akan menyimpan data di variabel x
29             if (vis[x]) {
30                 continue;
31             }
32             vis[x] = true;
33
34             for (int i = 0; i < adj[x].size(); i++) {
35                 int e = adj[x][i].first;
36                 int w = adj[x][i].second;
37                 if (dist[x] + w < dist[e]) {
38                     dist[e] = dist[x] + w;
39                     route[e] = x;
40                     pq.push({dist[e], e});
41                 }
42             }
43         }
44     }
45
46     // membuat function printRoute untuk mencetak suatu rute dengan parameter start dan end
47     void printRoute(int start, int end) {
48         if (dist[end] == INF) {
49             cout << "Tidak ada rute dari node " << start << " ke node " << end << endl;
50             return;
51         }
52
53         vector<int> path; // membuat path dengan vector
54         int currentNode = end; // deklarasi currentNode dengan isi variabel end;
55
56         while (currentNode != 0) {
57             path.push_back(currentNode);
58             currentNode = route[currentNode];
59         }
60
61         reverse(path.begin(), path.end());
62         cout << "Rute terpendek dari kota " << start << " ke kota " << end << " adalah:" << endl;
63         for (size_t i = 0; i < path.size(); ++i) {
64             cout << path[i] << (i == path.size() - 1 ? "" : " -> ");
65         }
66     }
67 }
```

```

68 // membuat function printTable untuk mencetak matrix membentuk tabel dengan parameter from dan to
69 void printTable(int from, int to) {
70     for (int i = from; i <= to; i++) {
71         for (int j = 0; j < to; j++) {
72             cout << "\t" << adj[i][j].second;
73         }
74         cout << endl;
75     }
76 }
77 };
78
79 int main() {
80     Graph g;
81     int N;
82     cout << "Masukkan Jumlah Kota : ";
83     cin >> N;
84     cout << "Nilai Cost Matrix" << endl;
85
86     for (int i = 1; i <= N; i++) {
87         cout << "Cost Element Baris ke-: " << i << endl;
88         for (int j = 1; j <= N; j++) {
89             int weight; // deklarasi bobot
90             cin >> weight;
91             adj[i].push_back({j, weight});
92         }
93         cout << endl;
94     }
95
96     cout << "Cost List : " << endl;
97     g.printTable(1, N);
98     cout << endl;
99
100     int asal, tujuan;
101
102     cout << "Masukkan Kota Asal : ";
103     cin >> asal;
104     cout << "Masukkan Kota Tujuan : ";
105     cin >> tujuan;
106
107     g.dijkstra(asal);
108     g.printRoute(asal, tujuan);
109
110     cout << endl;
111     cout << "Minimun Cost : " << dist[tujuan] << endl;
112
113     system("pause");
114
115     return 0;
116 }

```

2. Screenshot Output

```

Masukkan Jumlah Kota : 4
Nilai Cost Matrix
Cost Element Baris ke-: 1
0
1
4
3

Cost Element Baris ke-: 2
5
3
7
6

Cost Element Baris ke-: 3
2
0
5
3

Cost Element Baris ke-: 4
3
5
2
9

Cost List :
      0      1      4      3
      5      3      7      6
      2      0      5      3
      3      5      2      9

Masukkan Kota Asal : 2
Masukkan Kota Tujuan : 3
Rute terpendek dari kota 2 ke kota 3 adalah:
2 -> 3
Minimum Cost : 7
Press any key to continue . . .

```

3. Screenshot jam pengerjaan

01.37
26/06/2025

4