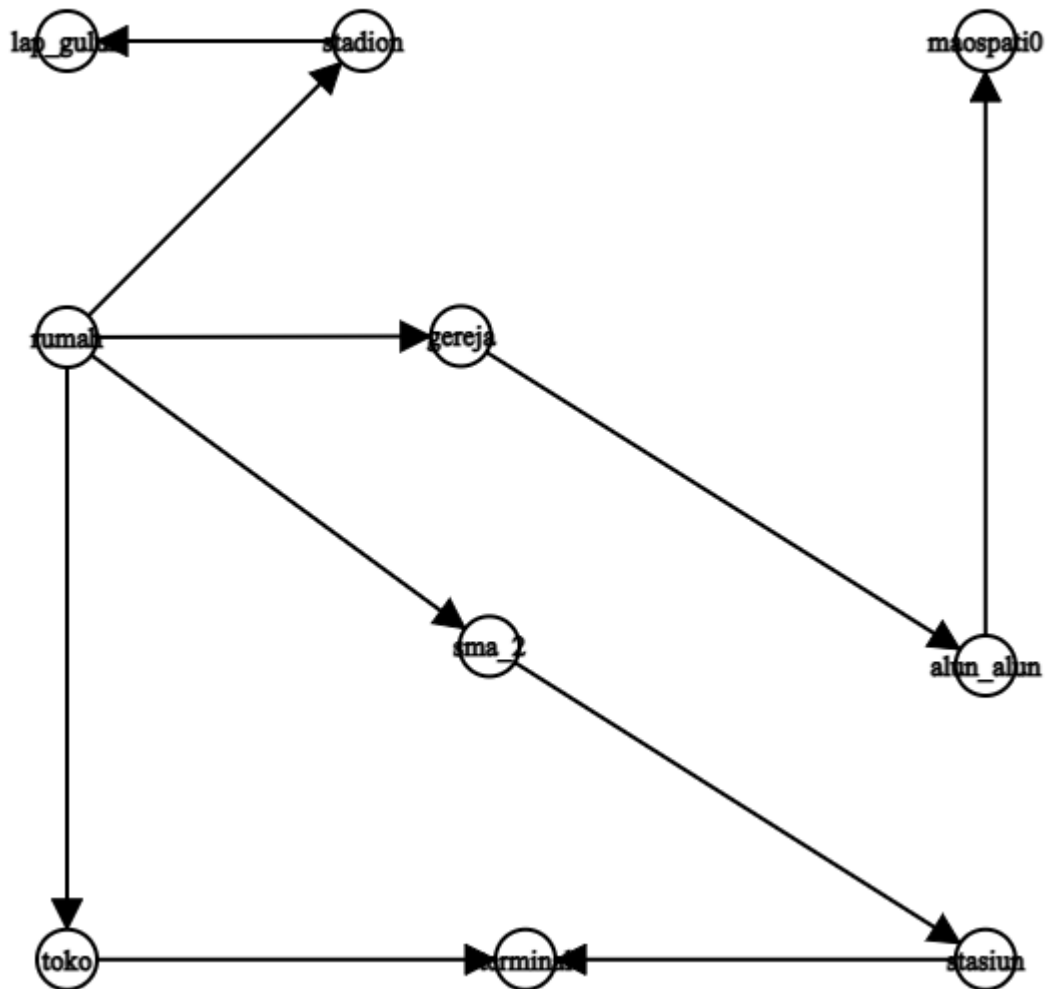


1. Gambarkan peta sekitar rumah kalian dengan minimal 10 titik dalam bentuk graf berarah (20 poin)

Jawaban:



2. Dengan menggunakan Class Peta yang telah saya contohkan, implementasikan peta yang telah kalian buat (nomor 1) ke dalam sebuah program dengan representasi graf adjacency list (20 poin)
3. Tampilkan hasil adjacency listnya (5 poin)
4. Buatlah Class baru bernama "Titik" untuk menyimpan ID titik, nama tempat (misal "Rumah", "Minimarket", "Apotek", dll), titik koordinat x, dan titik koordinat y. Instansiasi class "Titik" untuk menyimpan info titik pada peta. (20 poin)
5. Tampilkan hasil adjacency list berupa nama tempat (15 poin)
6. Tampilkan hasil graf menggunakan library graphics.h (10 poin)
7. Tambahkan modifikasi lain (1-10 poin)

Kode Program:

```
#include <iostream>
#include <list>
```

```
#include <stack>
using namespace std;

struct nodeName {
    string name;
};

class Map {
private:
    int node;
    list<int> *adjList;
    int **adjMatrix;
    nodeName *nodeInfo;

public:
    Map(int node, nodeName nodeInfo[]) {
        this->node = node;
        this->adjListInit(node);
        this->adjMatrixInit(node);
        this->nodeInfo = new nodeName[node];
        for (int i = 0; i < node; i++) {
            this->nodeInfo[i] = nodeInfo[i];
        }
    }

    ~Map() {
        delete[] adjList;
        delete[] nodeInfo;
    }

    void adjListInit(int node) {
        adjList = new list<int>[node];
    }

    void adjMatrixInit(int node) {
        adjMatrix = new int *[node];
        for (int i = 0; i < node; i++) {
            adjMatrix[i] = new int[node];
            for (int j = 0; j < node; j++) {
                adjMatrix[i][j] = 0;
            }
        }
    }

    void addRoute(int titik_awal, int titik_tujuan) {
        adjList[titik_awal].push_back(titik_tujuan);
        adjMatrix[titik_awal][titik_tujuan] = 1;
        adjMatrix[titik_tujuan][titik_awal] = 1;
    }
}
```

```
void ShowAdjList() {
    list<int>::iterator i;

    for (int v = 0; v < node; v++) {
        cout << nodeInfo[v].name << " -> ";
        for (i = adjList[v].begin(); i != adjList[v].end(); ++i) {
            cout << nodeInfo[*i].name;
            if (next(i, 1) != adjList[v].end()) {
                cout << " -> ";
            }
        }
        cout << endl;
    }
}

void showAdjMatrix() {
    for (int i = 0; i < node; i++) {
        for (int j = 0; j < node; j++) {
            cout << adjMatrix[i][j] << " ";
        }
        cout << endl;
    }
}

};

int main() {
    cout << "Peta Rumah" << endl;
    int node = 10;
    nodeName nodeInfo[] = {
        {"rumah"},
        {"sma_2"},
        {"stadion"},
        {"toko"},
        {"stasiun"},
        {"gereja"},
        {"alun_alun"},
        {"terminal"},
        {"lap_gulun"},
        {"maospati"}
    };

    Map myMap(node, nodeInfo);

    myMap.addRoute(0, 1);
    myMap.addRoute(0, 2);
    myMap.addRoute(0, 3);
    myMap.addRoute(0, 4);
    myMap.addRoute(1, 4);
    myMap.addRoute(2, 8);
    myMap.addRoute(3, 7);
```

```
myMap.addRoute(4, 7);
myMap.addRoute(5, 6);
myMap.addRoute(6, 9);

cout << endl;
cout << "Adjacency List" << endl;
myMap.ShowAdjList();

cout << endl;
cout << "Adjacency Matrix" << endl;
myMap.showAdjMatrix();

return 0;
}
```

Penjelasan Kode:

<pre>class Map { private: int node; list<int> *adjList; int **adjMatrix; nodeName *nodeInfo; ... }</pre>	<p>Ini adalah definisi dari kelas Map yang akan mewakili graf.</p> <p>Data Anggota (private):</p> <ul style="list-style-type: none"> • int node: Menyimpan jumlah simpul dalam graf. • list<int> *adjList: Mewakili daftar ketetanggaan dalam bentuk array, di mana setiap elemen adalah daftar simpul terhubung dari simpul tertentu. • int **adjMatrix: Mewakili matriks ketetanggaan, di mana elemen adjMatrix[i][j] adalah 1 jika ada jalur dari simpul i ke simpul j, dan 0 jika tidak. • nodeName *nodeInfo: Array dari struktur nodeName untuk menyimpan informasi tentang setiap simpul.
<pre>public: Map(int node, nodeName nodeInfo[]) { this->node = node; this->adjListInit(node); this->adjMatrixInit(node); this->nodeInfo = new nodeName[node]; for (int i = 0; i < node; i++) { this->nodeInfo[i] = nodeInfo[i]; } } }</pre>	<p>Metode-Metode Publik:</p> <ul style="list-style-type: none"> • Map(int node, nodeName nodeInfo[]) { ... }: Constructor kelas Map yang menginisialisasi objek Map dengan jumlah simpul dan informasi simpul yang diberikan. • ~Map() { ... }: Destructor untuk membersihkan memori yang dialokasikan secara dinamis.

```
~Map() {
    delete[] adjList;
    delete[] nodeInfo;
}

void adjListInit(int node) {
    adjList = new list<int>[node];
}

void adjMatrixInit(int node) {
    adjMatrix = new int *[node];
    for (int i = 0; i < node; i++) {
        adjMatrix[i] = new int[node];
        for (int j = 0; j < node; j++) {
            adjMatrix[i][j] = 0;
        }
    }
}

void addRoute(int titik_awal, int
titik_tujuan) {

adjList[titik_awal].push_back(titik_tujuan);
    adjMatrix[titik_awal][titik_tujuan] = 1;
    adjMatrix[titik_tujuan][titik_awal] = 1;
}

void ShowAdjList() {
    list<int>::iterator i;

    for (int v = 0; v < node; v++) {
        cout << nodeInfo[v].name << " -> ";
        for (i = adjList[v].begin(); i !=
adjList[v].end(); ++i) {
            cout << nodeInfo[*i].name;
            if (next(i, 1) != adjList[v].end()) {
                cout << " -> ";
            }
        }
        cout << endl;
    }
}

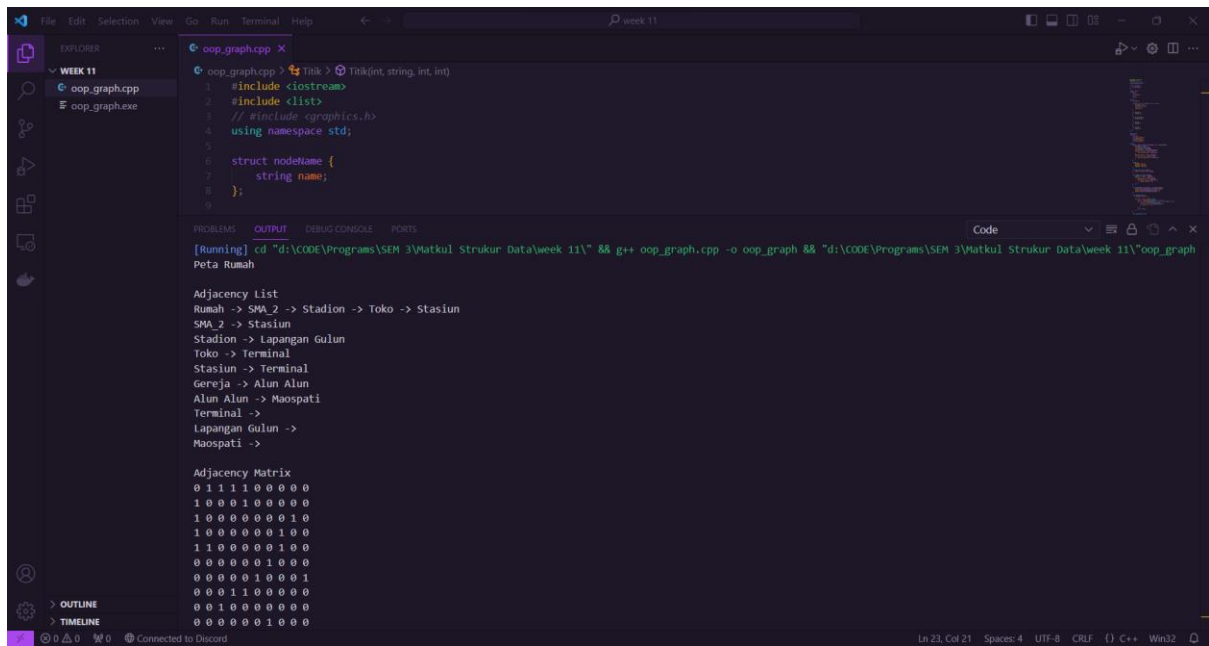
void showAdjMatrix() {
    for (int i = 0; i < node; i++) {
        for (int j = 0; j < node; j++) {
            cout << adjMatrix[i][j] << " ";
        }
        cout << endl;
    }
}
```

- void adjListInit(int node) { ... }:
Menginisialisasi adjList dengan array dari list<int>.
- void adjMatrixInit(int node) { ... }:
Menginisialisasi adjMatrix dengan matriks berukuran node x node dan mengisi elemennya dengan 0.
- void addRoute(int titik_awal, int titik_tujuan) { ... }:
Menambahkan rute antara dua simpul ke adjList dan adjMatrix.
- void ShowAdjList() { ... }:
Menampilkan representasi daftar ketetanggaan dari graf.
- void showAdjMatrix() { ... }:
Menampilkan representasi matriks ketetanggaan dari graf.

<pre> } } </pre>	
<pre> int main() { cout << "Peta Rumah" << endl; int node = 10; nodeName nodeInfo[] = { {"rumah"}, {"sma_2"}, {"stadion"}, {"toko"}, {"stasiun"}, {"gereja"}, {"alun_alun"}, {"terminal"}, {"lap_gulun"}, {"maospati"} }; Map myMap(node, nodeInfo); myMap.addRoute(0, 1); myMap.addRoute(0, 2); myMap.addRoute(0, 3); myMap.addRoute(0, 4); myMap.addRoute(1, 4); myMap.addRoute(2, 8); myMap.addRoute(3, 7); myMap.addRoute(4, 7); myMap.addRoute(5, 6); myMap.addRoute(6, 9); cout << endl; cout << "Adjacency List" << endl; myMap.ShowAdjList(); cout << endl; cout << "Adjacency Matrix" << endl; myMap.showAdjMatrix(); return 0; } </pre>	<p>Ini adalah fungsi utama dari program yang akan dieksekusi saat program dimulai.</p> <p>Di dalam fungsi main:</p> <ul style="list-style-type: none"> • Objek myMap dari kelas Map dibuat dengan 10 simpul dan informasi simpul yang diberikan. • Beberapa rute ditambahkan menggunakan fungsi addRoute untuk membuat hubungan antar simpul. • Kemudian, daftar ketetanggaan dan matriks ketetanggaan ditampilkan menggunakan fungsi ShowAdjList dan showAdjMatrix.

Output:

Marcelinus Alvinanda Chrisantya
5027221012 / Strukdat B



```
File Edit Selection View Go Run Terminal Help
oop_graph.cpp x
1 #include <iostream>
2 #include <list>
3 // #include <graphics.h>
4 using namespace std;
5
6 struct nodeName {
7     string name;
8 };
9
[Running] cd "d:\CODE\Programs\SEM 3\Matkul Strukur Data\week 11\" && g++ oop_graph.cpp -o oop_graph && "d:\CODE\Programs\SEM 3\Matkul Strukur Data\week 11\"oop_graph
Peta Rumah

Adjacency List
Rumah -> SMA_2 -> Stadion -> Toko -> Stasiun
SMA_2 -> Stasiun
Stasiun -> Lapangan Gulun
Toko -> Terminal
Stasiun -> Terminal
Gereja -> Alun Alun
Alun Alun -> Maospati
Terminal ->
Lapangan Gulun ->
Maospati ->

Adjacency Matrix
0 1 1 1 1 0 0 0 0 0
1 0 0 0 1 0 0 0 0 0
1 0 0 0 0 0 0 0 1 0
1 0 0 0 0 0 0 1 0 0
1 1 0 0 0 0 0 1 0 0
0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 1 0 0 0 1
0 0 0 1 1 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0
```