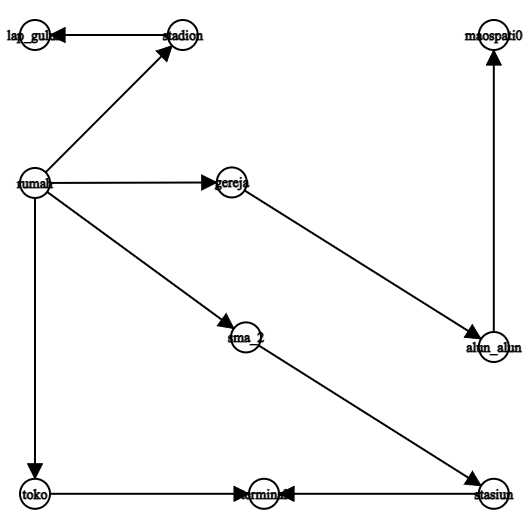
1. Gambarkan peta sekitar rumah kalian dengan minimal 10 titik dalam bentuk graf berarah (20 poin)

**Jawaban:**



1. 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)
2. Tampilkan hasil adjacency listnya (5 poin)
3. Buatkan 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)
4. Tampilkan hasil adjacency list berupa nama tempat (15 poin)
5. Tampilkan hasil graf menggunakan library graphics.h (10 poin)
6. Tambahkan modifikasi lain (1-10 poin)

Kode Program:

#include <iostream>

#include <list>

*// #include <graphics.h>*

using namespace std;

struct nodeName {

    string name;

};

class Titik {

private:

    int id;

    string nama;

    int x;

    int y;

public:

    Titik() {}

    Titik(int id, string nama, int x, int y) {

        this->id = id;

        this->nama = nama;

        this->x = x;

        this->y = y;

    }

    int getID() {

        return id;

    }

    string getNama() {

        return nama;

    }

    int getX() {

        return x;

    }

    int getY() {

        return y;

    }

};

class Map {

private:

    int node;

    list<int> \*adjList;

    int \*\*adjMatrix;

    nodeName \*nodeInfo;

    Titik \*titikInfo;

public:

    Map(int node, nodeName nodeInfo[], Titik titikInfo[]) {

        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];

        }

        this->titikInfo = new Titik[node];

        for (int i = 0; i < node; i++) {

            this->titikInfo[i] = titikInfo[i];

        }

    }

    ~Map() {

        delete[] adjList;

        delete[] nodeInfo;

        delete[] titikInfo;

    }

    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 << titikInfo[v].getNama() << " -> ";

            for (i = adjList[v].begin(); i != adjList[v].end(); ++i) {

                cout << titikInfo[\*i].getNama();

                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;

        }

    }

    list<int>\* getAdjList() {

        return adjList;

    }

    int getNode() {

        return node;

    }

};

*// void drawGraph(Titik titikInfo[], Map myMap) {*

*//     int gd = DETECT, gm;*

*//     initgraph(&gd, &gm, NULL);*

*//     for (int i = 0; i < myMap.getNode(); i++) {*

*//         circle(titikInfo[i].getX(), titikInfo[i].getY(), 5);*

*//         outtextxy(titikInfo[i].getX() + 10, titikInfo[i].getY(), titikInfo[i].getNama().c\_str());*

*//     }*

*//     for (int i = 0; i < myMap.getNode(); i++) {*

*//         for (int tujuan : myMap.getAdjList()[i]) {*

*//             line(titikInfo[i].getX(), titikInfo[i].getY(), titikInfo[tujuan].getX(), titikInfo[tujuan].getY());*

*//         }*

*//     }*

*//     delay(5000);*

*//     closegraph();*

*// }*

int main() {

    cout << "Peta Rumah" << endl;

    int node = 10;

    nodeName nodeInfo[] = {

        {"rumah"},

        {"sma\_2"},

        {"stadion"},

        {"toko"},

        {"stasiun"},

        {"gereja"},

        {"alun\_alun"},

        {"terminal"},

        {"lap\_gulun"},

        {"maospati"}

    };

    Titik titikInfo[] = {

        Titik(0, "Rumah", 100, 100),

        Titik(1, "SMA\_2", 200, 100),

        Titik(2, "Stadion", 300, 100),

        Titik(3, "Toko", 100, 200),

        Titik(4, "Stasiun", 200, 200),

        Titik(5, "Gereja", 300, 200),

        Titik(6, "Alun Alun", 100, 300),

        Titik(7, "Terminal", 200, 300),

        Titik(8, "Lapangan Gulun", 300, 300),

        Titik(9, "Maospati", 400, 300)

    };

    Map myMap(node, nodeInfo, titikInfo);

    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();

*// drawGraph(titikInfo, myMap);*

    return 0;

}

Output:

