Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по лабораторной работе №21**

Дисциплина: «Основы теории алгоритмов и структуры данных»

Тема: Графы

Вариант 1

Выполнил:

Студент группы РИС-20-1б

Азмагулов Артём Вадимович

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2021**

**Цель работы**

Ознакомление с графами и демонстрация алгоритма Дейкстры

**Постановка задачи**

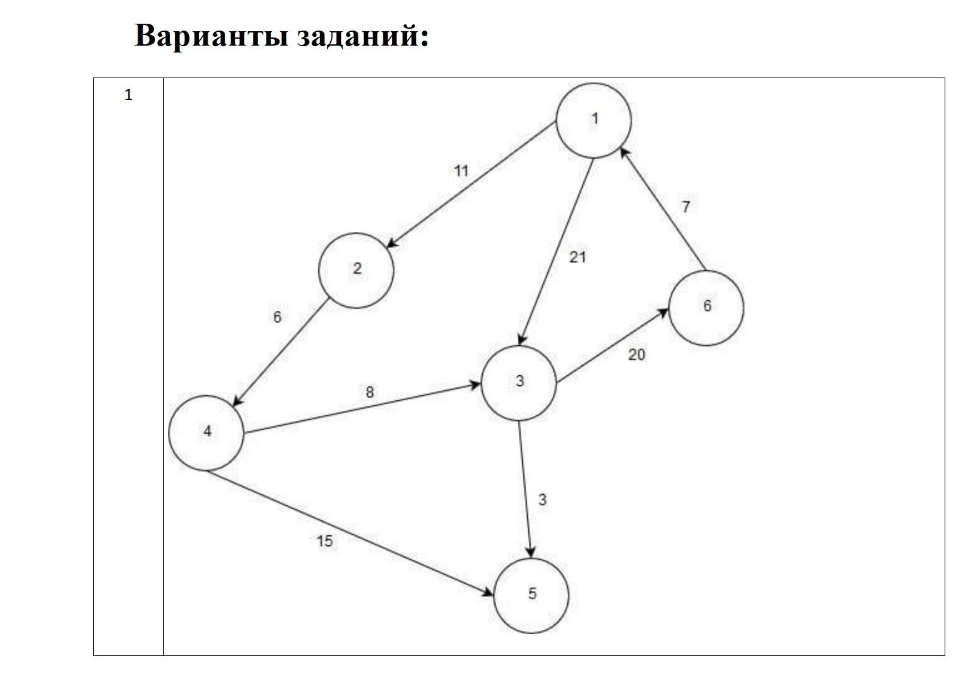
Реализовать алгоритм Дейкстры для графа, соответствующего выбранному варианту.

1. Без использования Qt.

2. Интерфейс на усмотрение разработчика.

3. Визуализация графа в OpenGL.

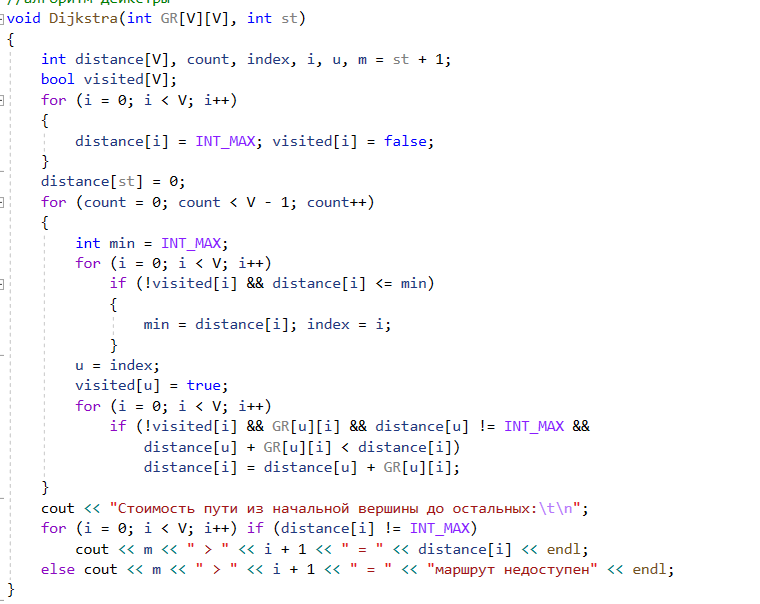
4. Реализуется только метод Дейкстры



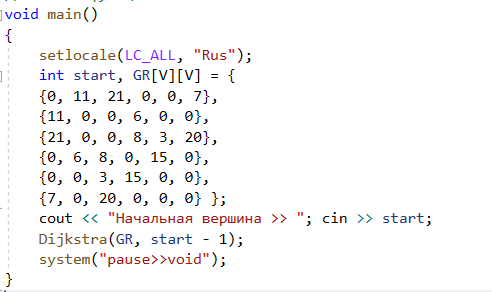
**Анализ задачи**

**1.** Для решения задачи необходимо:

**1.1.** Реализовать алгоритм Дейкстры с приемом елементов из двумерного массива



**1.2.** Реализовать главную функцию по вызову алгоритма Дейкстры



**2.**В ходе работы были использованы следующие типы данных:

**2.1.** Все данные дерева принимались как элементы целочисленного типа:



**3.**Для решения задачи данные были представлены в следующем виде:

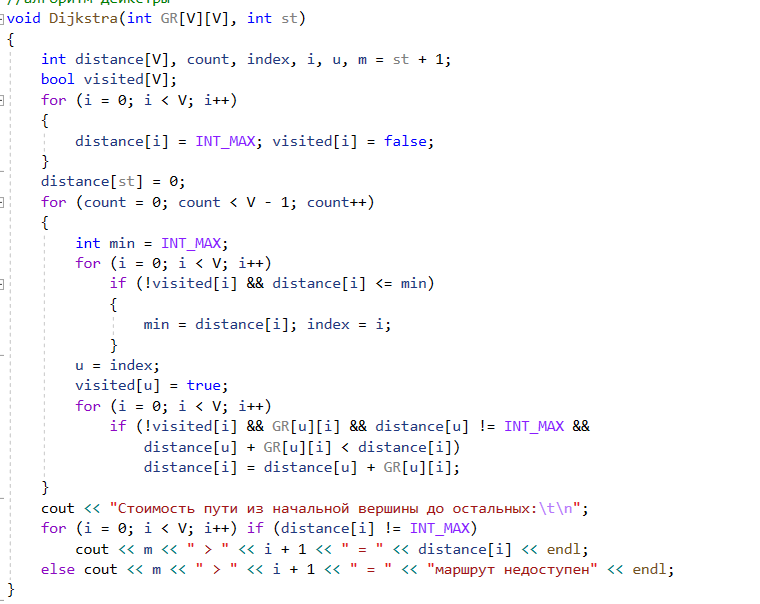
**3.1.**Данные введенные пользователем принимают целочисленное значение

**4.**Для операций ввода и вывода использовались следующие операторы и функции:

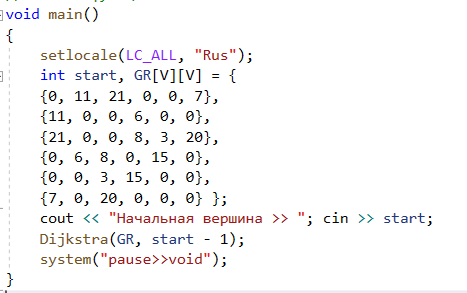
**4.1.** Вывод данных на консоль реализован с помощью оператора cout.



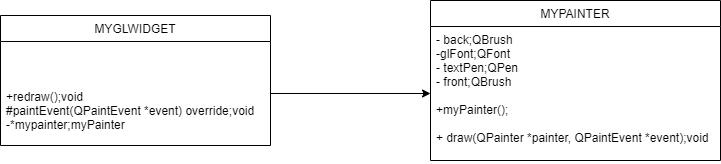
**5.**Поставленные задачи будут решены следующими действиями:

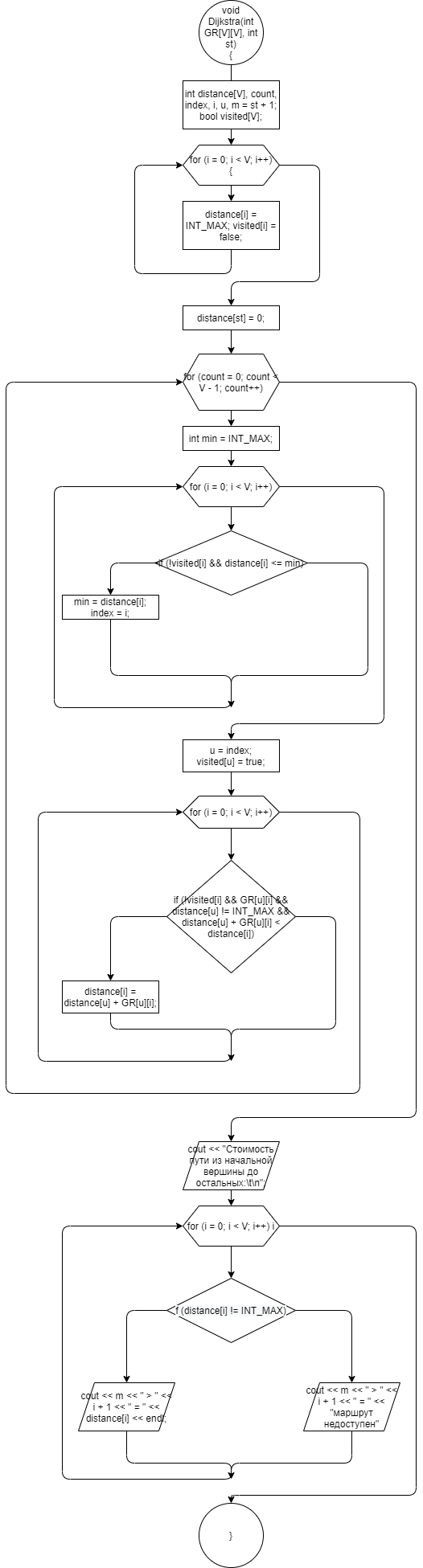
**5.1.** К каждому элементу графа от начальной точки присваивается максимально большое расстояние, затем идет переход по графам и переприсваивание расстояний найденных вершин

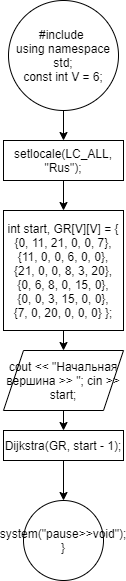
**5.2.** В главной функции происходит создание таблицы смежности с дальнейшим вызовом функции с алгоритмом Дейкстры, с указанием в нем количества графов и начальную координату



**Блок-схема программы**

****





**Решение**

#include <iostream>

using namespace std;

const int V = 6;

//алгоритм Дейкстры

void Dijkstra(int GR[V][V], int st)

{

int distance[V], count, index, i, u, m = st + 1;

bool visited[V];

for (i = 0; i < V; i++)

{

distance[i] = INT\_MAX; visited[i] = false;

}

distance[st] = 0;

for (count = 0; count < V - 1; count++)

{

int min = INT\_MAX;

for (i = 0; i < V; i++)

if (!visited[i] && distance[i] <= min)

{

min = distance[i]; index = i;

}

u = index;

visited[u] = true;

for (i = 0; i < V; i++)

if (!visited[i] && GR[u][i] && distance[u] != INT\_MAX &&

distance[u] + GR[u][i] < distance[i])

distance[i] = distance[u] + GR[u][i];

}

cout << "Стоимость пути из начальной вершины до остальных:\t\n";

for (i = 0; i < V; i++) if (distance[i] != INT\_MAX)

cout << m << " > " << i + 1 << " = " << distance[i] << endl;

else cout << m << " > " << i + 1 << " = " << "маршрут недоступен" << endl;

}

//главная функция

void main()

{

setlocale(LC\_ALL, "Rus");

int start, GR[V][V] = {

{0, 11, 21, 0, 0, 7},

{11, 0, 0, 6, 0, 0},

{21, 0, 0, 8, 3, 20},

{0, 6, 8, 0, 15, 0},

{0, 0, 3, 15, 0, 0},

{7, 0, 20, 0, 0, 0} };

cout << "Начальная вершина >> "; cin >> start;

Dijkstra(GR, start - 1);

system("pause>>void");

}

cout << endl << endl <<"Количество элементов равных ключу: "<< col;

// getch();

}}

**Скриншоты результатов работы программы**

