Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«Пермский национальный исследовательский политехнический университет» (ПНИПУ)

Электротехнический факультет

Кафедра информационных технологий и автоматизированных систем

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

**Лабораторная работа по Графам**

**по дисциплине**

**«Основы алгоритмизации и программирования»**

**семестр** 2

Выполнил студент гр. ИВТ-21-1б

Балдин Алексей Игоревич

Проверил:

Яруллин Денис Владимирович

\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

(оценка) (подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

г. Пермь - 2022

**Цель**

Получить практические навыки работы с графами.

**Постановка задачи**

Реализовать граф, а также алгоритм Дейкстры, выполнив все необходимые действия. Выполнение начать с вершины 1.

**Исходные данные**

Вариант 5:

Заполнил матрицу смежности данными из варианта. Указал начальный индекс – 0(так как начинать с элемента 1). Далее реализовал самый обычный алгоритм Дейкстры. Для вывода квадратной матрицы использовал вложенный цикл for. Для вывода массива кратчайших путей, также был использован цикл for.

Код программы:

#include <iostream>

int SIZE = 6;

const int n = 10000; // заглушка для прохода в цикл

using namespace std;

int main()

{

setlocale(LC\_ALL, "rus");

int a[6][6] = // матрица связей

{ /\*/1./\*/{0,0,5,16,0,0},

/\*/2./\*/{0,0,33,0,9,20},

/\*/3./\*/{5,33,0,0,7,0},

/\*/4./\*/{16,0,0,0,10,14},

/\*/5./\*/{0,9,7,10,0,0},

/\*/6./\*/{0,20,0,14,0,0}

};

int min\_d[6]; // минимальное расстояние

int count[6]; // посещенные вершины

int temp, minindex, min;

int begin\_index = 0;

cout << endl << "Матрица связей: " << endl << endl;

for (int i = 0; i < SIZE; i++)

{

for (int j = 0; j < SIZE; j++)

{

if (a[i][j] > 9)

{

cout <<" " << a[i][j];

}

else { cout << " " << a[i][j]<<" "; }

}

cout << endl;

}

//Инициализация вершин и расстояний

for (int i = 0; i < SIZE; i++)

{

min\_d[i] = n;

count[i] = 1;

}

min\_d[begin\_index] = 0;

do {

minindex = n;

min = n;

for (int i = 0; i < SIZE; i++)

{ // Если вершину ещё не обошли и вес меньше min

if ((count[i] == 1) && (min\_d[i] < min))

{

min = min\_d[i];

minindex = i;

}

}

if (minindex != n)

{

for (int i = 0; i < SIZE; i++)

{

if (a[minindex][i] > 0)

{

temp = min + a[minindex][i];

if (temp < min\_d[i])

{

min\_d[i] = temp;

}

}

}

count[minindex] = 0;

}

}

while (minindex < n);

cout << endl << "Кратчайшие расстояния до вершин: " << endl << endl;

for (int i = 0; i < SIZE; i++)

{

cout << "До " << (i+1) << " вершины: ";

cout << " " << min\_d[i] << endl << endl;

}

return 0;

}

Результат программы:

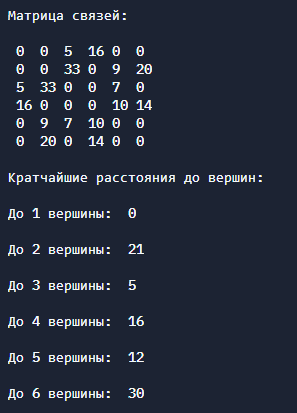


Диаграмма к задаче:

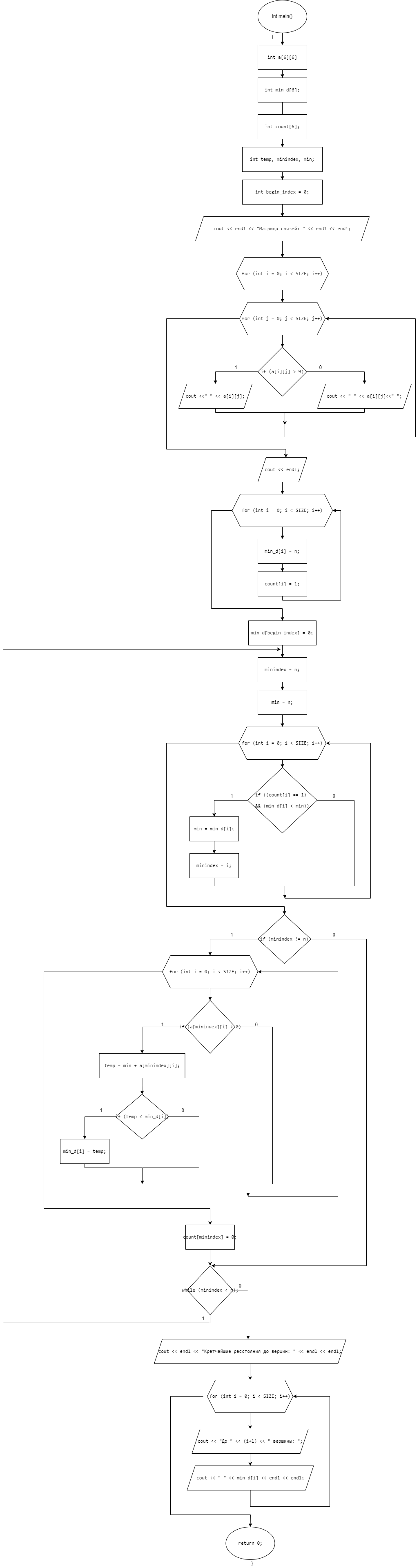


Рисунок 1