Министерство образования и науки РФ
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
Санкт-Петербургский государственный
электротехнический университет «ЛЭТИ» имени В.И. Ульянова (Ленина)

Факультет компьютерных технологий и информатики
Кафедра вычислительной техники

Курсовая работа
по дисциплине «Алгоритмы и структуры данных»
«Графы»

Вариант№21

Выполнил: студент группы 5005
Зинатуллин Азат Салаватович

Проверил: старший преподаватель
Колинько Павел Георгиевич

Цель работы

Научиться представлять графы в памяти компьютера и научиться с ними работать, в частности вычислять кратчайшие пути между вершинами.

1. Задание

Вычислить матрицу расстояний между всеми парами вершин неориентированного графа с нагруженными рёбрами.

2. Математическая формулировка задачи и описание алгоритма.

Пусть веса дуг заданы в виде матрицы D. Будем решать задачу методом динамического программирования. Обозначим за T(i,j,k) длину кратчайшего пути между вершинами i и j, который в качестве промежуточных содержит только вершины с номерами, не превосходящими k. Рассмотрим случай, когда k=0. Это означает, что промежуточных вершин в путях быть не может. Значит, путь будет существовать между теми вершинами, между которыми по условию есть дуга. Тогда матрицу T(i,j,0) построим на основе матрицы D следующим образом.
Во-первых, расстояние между вершинами, между которыми нет дуги, положим равным бесконечности.

Во-вторых, из вершины в неё саму всегда можно добраться за нулевое число шагов.
Искомый кратчайший путь может либо проходить через вершину с номером *k*, либо нет. Если он проходит через эту вершину, то его можно разбить на две части: путь от *i* до *k* и путь от *k* до *j*. Поскольку оба этих пути должны являться кратчайшими, имеем следующее рекуррентное соотношение:

$$T(j.i.k) = \begin{cases} \infty, \text{ if } k=0 \text{ ; } i \text{ != } j \text{ ; } (i,j) \text{ != } E; \\ \min (D(i,j;0)), \text{ if } k=0, i=j; \\ \min (T(i,j,k-1); T(i,k,k-1) + T(k,j,k-1)) \text{ , if } k>0. \end{cases}$$

Сложность алгоритма: O(N^3).

3. Обоснование выбора способа представления графа в памяти.

В данном задании граф удобно представить массивом пар, в котором рёбра заданы парой инцидентных вершин.
Требует памяти 2xN, где N-кол-во вершин графа.
Матрица - симметрична по диагонали, так как по условию задания граф неориентированный (смежные вершины связаны петлями).
На пересечении i и j элемента задается вес рёбра.

4. Результаты прогона программы с генерацией случайного.

INF - значит, что вершины не являются смежными по отношению друг к другу.

```
Введите кол-во вершин неориентированного взвешенного графа:
5

Матрица смежности исходного графа:

0 INF 30 61 25
INF 0 66 6 63
30 66 0 38 INF
61 6 38 0 10
25 63 INF 10 0

Результат выполнения алгоритма Флойда-Уоршелла
Матрица кратчайших путей между вершинами графа:

0 41 30 35 25
41 0 44 6 16
30 44 0 38 48
35 6 38 0 10
25 16 48 10 0
```

Тест 1.

```
Введите кол-во вершин неориентированного взвешенного графа:
4

Матрица смежности исходного графа:

0 65 INF 15
65 0 14 64
INF 14 0 38
15 64 38 0

Результат выполнения алгоритма Флойда-Уоршелла
Матрица кратчайших путей между вершинами графа:

0 65 53 15
65 0 14 52
53 14 0 38
15 52 38 0
```

Тест 2.

5. Выводы

В ходе выполнения курсовой работы было реализовано представление неориентированного графа в памяти ЭВМ в виде массива пар, и вычисление кратчайших путей между вершинами при помощи алгоритма Флойда-Уоршелла, который имеет кубическую временную сложность.

6. Список использованных источников

1. Алгоритмы и структуры данных: методические указания к лабораторным работам, практическим занятиям и курсовому проектированию. Ч. 1. Вып. 1701 (для заочников) / сост.: П. Г. Колинько. — СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2017. — 64 с.

2. Курс видео уроков по С++. https://www.youtube.com/playlist?list=PLbmlzoDQrXVFC13GjpPrJxl6mzTiX65gs

3. Алгоритм Флойда-Уоршелла http://cybern.ru/algoritm-flojda-uorshella.html

4. Использовал, как справочник по некоторым функциям. http://cppstudio.com/cat/274/

7. Приложение

main.cpp - Файл с программой

```
// Исходный код программы.

#include <iostream>
#include <cstdlib>
#include <ctime>

using namespace std;

const int inf = 200;

int main()
{
        srand(time(NULL));
        int n,i,j,k,d[10][10];  // d[10][10] - двумерный масив для хранения матрицы
        cout << "Введите кол-во вершин неориентированного взвешенного графа: "<<endl;
        scanf("%d",&n);     // ввод кол-ва вершин с клавиатуры

        cout << "\nМатрица смежности исходного графа: \n"<<endl;
        for (i=0;i<n;++i)
            for (j=0;j<n;++j)
            {
                d[i][j]=rand()%100;         // Задает вес ребра (не больше 100)
                d[j][i]=d[i][j];            // Матрица симметрична, так как граф неориентирован.

                if (i==j) d[i][j]=min(d[i][j],0);
                if (d[i][j]>70) d[i][j]=d[j][i]=inf;    // Если длина пути между вершинами > 70, то
                                                        // они не являются смежными.
            }

        // Вывод матрицы инциденций графа
        for (i=0;i<n;++i,printf("\n"))
            for (j=0;j<n;++j)
                if (d[i][j]==inf) printf("INF "); else printf("%d ",d[i][j]); // Вершины не смежны

        // Реализация алгоритма Флойда - Уоршелла
        // Имеет сложность O(N^3), где N - число вершин графа.
        for (k=0;k<n;++k)
            for (i=0;i<n;++i)
                for (j=0;j<n;++j)
                    if (d[i][k]<inf && d[k][j]<inf) d[i][j]=min(d[i][j],d[i][k]+d[k][j]);
```

```cpp
        cout << "\nРезультат выполнения алгоритма Флойда-Уоршелла"<<endl;
        cout << "Матрица кратчайших путей между вершинами графа: \n"<<endl;

        // Вывод матрицы кратчайших путей
        for (i=0;i<n;++i,printf("\n"))
            for (j=0;j<n;++j)
                if (d[i][j]==inf) printf("INF "); else printf("%d ",d[i][j]);

    return 0;
}
```