Министерство образования и науки РФ

СевГУ

Кафедра ИС

Отчёт

по лабораторной работе №5

«Поиск кратчайших путей на графах»

Выполнил:

ст.гр. ИТ-12д

Воронин И.Ю.

Проверил:

Заикина Е.Н.

Севастополь

2015

**1. ЦЕЛЬ РАБОТЫ**

Целью работы является изучение алгоритмов поиска кратчайших путей на графах на примере метода динамического программирования

**2. ВАРИАНТЫ ЗАДАНИЙ**

Нахождение кратчайшего пути в графах при помощи метода динамического программирования, при это вводя вручную сам граф.

**3.СТРУКТУРНЫЕ СХЕМЫ**

Работа основных программ представлены в следующих структурных схемах.







**4.КОД ПРОГРАММЫ**

#include <iostream>

#include <conio.h>

#include <Windows.h>>

const int INF = 1000000000; // бесконечность

bool used[100] = {0}; // массив для пометок

int top[100] = {0}; // топологический список

int g[100][100] = {0}; // матрица смежности

int n; // количество вершин

int l; // l-я вершина для добавления

int s; // стартовая вершина

int f; // конечная вершина

int d[100] = {0}; // массив ответов

using namespace std;

extern void solve();

extern int topSort();

extern int dfs(int v);

int main(){

setlocale(LC\_ALL,"Russian");

cout<<"Введите количество узлов: ",cin>>n;

cout<<"Введите стартовую вершину: ", cin>>s;

cout<<"Введите конечную вершину: ", cin>>f;

for (int i=0;i<100;i++) d[i]=INF;

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

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

cin>>g[i][j];

unsigned long long time1 = GetTickCount();

for(int i=0;i<5000;i++)

solve();

cout<<endl<<"Самый короткий путь равен "<<d[f]<<"."<<endl;;

double time = GetTickCount()- time1;

cout<<"Дейкстра была выполнена(5000 запусков) за "<<time<<" мс."<<'\n';

//топологический поиск

used[100] = {0};

time1 = GetTickCount();

for(int i=0;i<5000;i++)

topSort();

cout<<endl<<"Самый короткий путь равен "<<d[f]<<"."<<endl;;

time = GetTickCount()- time1;

cout<<"Топологический метод был выполнен(5000 запусков) за "<<time<<" мс."<<'\n';

getch();

return 0;

}

void solve() {

// n - количество вершин

// g[n][n] - матрица смежности, g[i][j] = 0, если текущего ребра нет

// d[n] - массив ответов

// s - стартовая вершина

// used - массив для пометок вершин

int i,j, // i-я итерация, j - для поиска минимальной

v, // минимальная вершина

to, // ребро из вершины v в to

len; // длины len

d[s] = 0;

for(i = 0; i < n; i++) {

v = -1;

for(j = 0; j < n; j++) // поиск вершины с минимальным d[j]

if(!used[j] && (v == -1 || d[j] < d[v]))

v = j;

used[v] = true; // пометка вершины

for(to = 0; to < n; to++) {

if(g[v][to]) {

len = g[v][to];

if(d[v] + len < d[to])

d[to] = d[v] + len;

}

}

}

}

int dfs(int v) {

if(used[v])

return 0;

used[v] = true;

for(int to = 0; to < n; to++)

if(g[v][to])

dfs(to);

top[l++] = v; // добавление вершины v в отсортированный список

}

int topSort() {

l = 0; // номер добавляемой вершины в отсортированный список

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

dfs(i); // запустить пробежку из всех вершин

reverse(top, top+l); // развернуть массив

return 0;

}

**5.РЕЗУЛЬТАТЫ РАБОТЫ ПРОГРАММЫ.**

Тест программы будет производиться на следующем графе(рис.5.1):



Рисунок 5.1-Граф тестовой программы.

Запуск программы выглядит следующим образом.(рис.5.2)

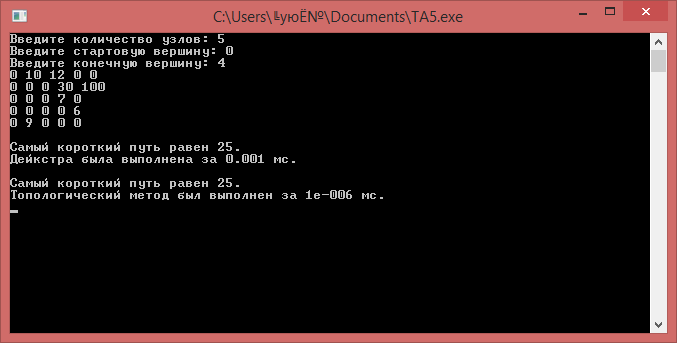


Рисунок 5.2-Тестовый запуск программы.

При помощи данной программы мы можем найти кратчайший путь в направленном графе. В данном случае быстрее это будет выполнять функция, написанная с применением алгоритма динамического программирования оказалась быстрее чем, алгоритм Дейкстры.

**6.ВЫВОДЫ**

В данной работе были изучены методы поиска кратчайшего пути в графах на примере кода языка С++. Были сравнены два алгоритма посика и выбран наиболее быстрый. Также было детально изучено понятие графа, его виды и свойства.