Пензенский государственный университет  
Кафедра «Вычислительной техники»

**Отчет**по лабораторной работе №6  
по дисциплине: «Логика и основы алгоритмизации в инженерных задачах»  
на тему: «Поиск расстояний в графе»

**Выполнил студент группы 19ВВ1:**

Балалаев А.А.

**Приняли:**

д.т.н. профессор Митрохин М. А.

к.т.н. Юрова О.В.

Пенза 2020.

**Цель работы:** изучить поиск расстояний в графе, научится реализовывать его используя обход в ширину.

**Теория**

Поиск расстояний – довольно распространенная задача анализа графов.

Вход: G – матрица смежности графа, v – исходная вершина.

Выход: DIST – вектор расстояний до всех вершин от исходной.

Алгоритм ПОШ

1.1. для всех i положим DIST [i] = -1 пометим как &quot;не посещенную&quot;;

1.2. ВЫПОЛНЯТЬ BFSD (v).

1.3 для всех i вывести DIST [i] на экран;

Алгоритм BFSD(v):

2.1. Создать пустую очередь Q = {};

2.2. Поместить v в очередь Q.push(v);

2.3. Обновить вектор расстояний DIST [ x ] = 0;

2.4. ПОКА Q != Æ очередь не пуста ВЫПОЛНЯТЬ

2.5. v = Q.front() установить текущую вершину;

2.6. Удалить первый элемент из очереди Q.pop();

2.7. вывести на экран v;

2.8. ДЛЯ i = 1 ДО size\_G ВЫПОЛНЯТЬ

2.9. ЕСЛИ G(v,i) = = 1И DIST = = -1

2.10. ТО

2.11. Поместить i в очередь Q.push(i);

2.12. Обновить вектор расстояний DIST [ i ] = DIST [ v ] + 1;

**Код программы**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <windows.h>

#include <stdio.h>

#include <time.h>

#include <conio.h>

#include <queue>

#include <locale.h>

using namespace std;

void rand\_Zap(int\*\* Matrix, int n) {

srand(time(NULL));

printf("G\n");

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

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

if (i == j) {

Matrix[i][j] = 0;

}

if (i < j) {

Matrix[i][j] = rand() % 2;

Matrix[j][i] = Matrix[i][j];

}

}

}

void print\_G(int\*\* Matrix, int n) {

printf(" ");

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

printf("%3d", i + 1);

printf("\n\n");

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

printf("%2d", i + 1);

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

printf("%3d", Matrix[i][j]);

}

printf("\n");

}

printf("\n");

}

void BFSD(int v, int\* dist, int\*\* G, int N)

{

queue <int> Q;

Q.push(v);

dist[v] = 0;

while (Q.empty() == false) {

v = Q.front();

Q.pop();

printf("%d ", v + 1);

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

if ((G[v][i] == 1) && dist[i] == -1) {

Q.push(i);

dist[i] = dist[v] + 1;

}

}

}

printf("\n");

}

int main()

{

setlocale(LC\_ALL, "Rus");

srand(time(NULL));

int N = 1;

scanf("%d", &N);

int\*\* G = (int\*\*)malloc(N \* sizeof(int\*));

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

{

G[i] = (int\*)malloc(N \* sizeof(int));

}

int\* dist;

dist = (int\*)malloc(N \* sizeof(int));

int i = 0, j = 0;

for (i = 0; i < 5; i++)

dist[i] = -1;

rand\_Zap(G, N);

print\_G(G, N);

/\*for (j = 0; j < N; j++)\*/

{

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

dist[i] = -1;

/\*printf("От вершины %d\n", j + 1);\*/

int p = 0, l = 0;

printf("Введите вершины\n");

scanf("%d %d", &p, &l);

printf("От вершины %d\n", p);

p -= 1;

l -= 1;

BFSD(p, dist, G, N);

printf("%d ", dist[l]);

/\*for (i = 0; i < N; i++)

{

if (dist[i] != -1)

printf("%d \t", dist[i]);

else

printf("Нет прохода \t");

}\*/

printf("\n");

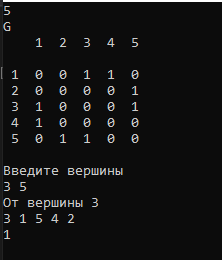
}

getchar();

getchar();

}

**Результат работы кода**



**Вывод:** изучил поиск расстояний в графе, научился реализовывать его, используя обход в ширину.