Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №6

**по курсу «Логика и основы алгоритмизации в инженерных задачах»**

на тему «Поиск расстояний в графе»

Выполнили:

студент группы 20ВВ2

Ермолаев А.Д.

Мартынов Н.В.

Принял:

д.т.н., профессор

Митрохин М.А.

к.т.н., доцент

Юрова О.В.

**Пенза 2021**

**Цель работы:**  составить программу, которая будет осуществлять процедуру обхода графа в ширину и будет выводить расстояние от выбранной вершины до других, которые ей смежны.

**Лабораторные задания:**

1.Сгенерируйте (используя генератор случайных чисел) матрицу смежности для неориентированного графа G. Выведите матрицу на экран.

2. Для сгенерированного графа осуществите процедуру обхода в ширину с реализацией поиска расстояния от выбранной вершины до всех остальных.

**Листинг:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <queue>

#include <iostream>

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

#include <locale.h>

using namespace std;

queue <int> q;

void BFSD(int v, int\*\* M, int\* Vis, int Size)

{

int i;

q.push(v); Vis[v] = 0;

while (!q.empty())

{

v = q.front();

q.pop();

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

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

{

if ((Vis[i] == -1) && (M[v][i] == 1))

{

q.push(i);

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

}

}

}

}

void main()

{

int\*\* G = NULL, i = 0, j = 0, S = 0, \* Visited = NULL, N;

char ReVis = 0;

setlocale(LC\_ALL, "Rus");

srand(time(NULL));

printf("Введите размерность матрицы смежности: "); //Ввод размерности матрицы

scanf("%d", &S);

while (S <= 0) //Проверка корректности введенного числа

{

printf("Некорректно введённое число! Введите целое положительное число: ");

scanf("%d", &S);

}

G = (int\*\*)malloc(S \* sizeof(int)); //Выделяем память

Visited = (int\*)malloc(S \* sizeof(int));

for (i = 0;i < S;i++) //Помечаем вершины как "непосещенные"

Visited[i] = -1;

for (i = 0;i < S;i++) //Генерация матрицы

{

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

for (j = 0;j < S;j++)

{

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

if (G[i][i] != 0)

G[i][j] = 0;

}

}

for (i = 0;i < S;i++) //Вывод матрицы на экран

{

printf("\n");

for (j = 0;j < S;j++)

{

G[j][i] = G[i][j];

printf("%2d", G[i][j]);

}

}

printf("\n\nС какой вершины начать обход в ширину? ");

scanf("%d", &N);

while (N<1 || N>S) //Проверка номера введенной вершины

{

printf("Неверное значение! Попробуйте еще раз: ");

scanf("%d", &N);

}

printf("\nОбход графа в ширину: ");

i = N - 1;

BFSD(i, G, Visited, S); //Обход графа в ширинв

printf("\nВектор расстояний от выбранной вершины до всех остальных: ");

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

if (Visited[i] != -1) {

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

}

else {

printf(" нет\_связи ");

}

}

printf("\n");

free(Visited); //Очистка памяти

free(G);

}

**Псевдокод:**

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 “не посещенную”;

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

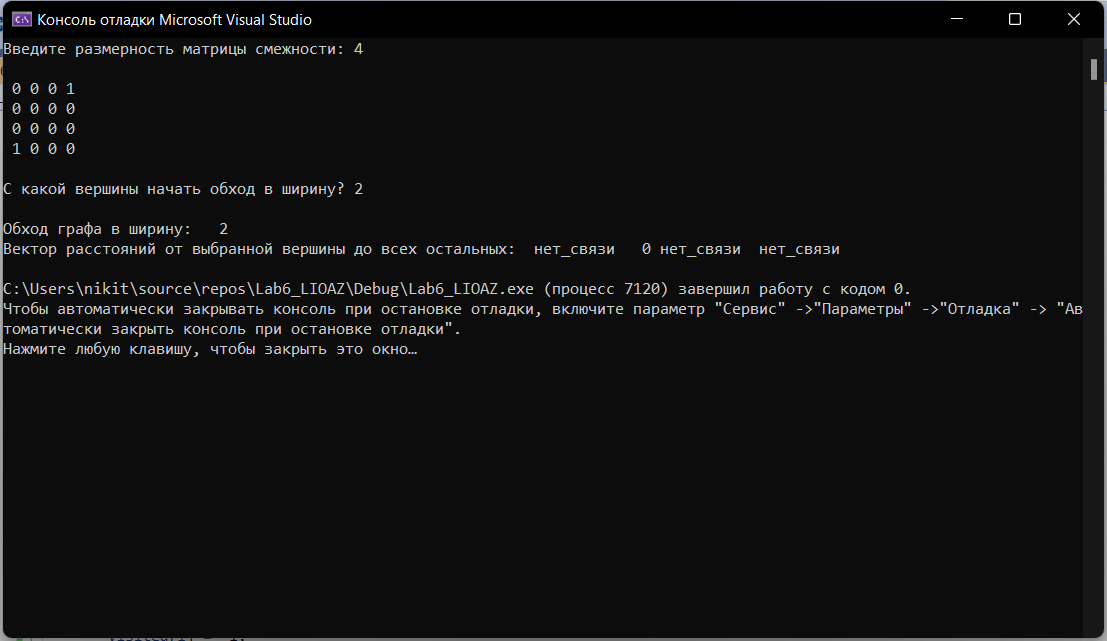
2.9. ЕСЛИ G(v,i) = = 1И DIST = = -1

2.10. ТО

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

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

**Результаты программы:**





**Вывод:** составили программу, осуществляющую поиск расстояний в графе.