Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №7

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «Обход графа в глубину»

**Выполнили студенты группы 21вв1:**

Нечаев А.Д.

Киреев Д.А.

**Приняли**

Юрова О.В.

Акифьев И.В.

Пенза 2022

**Методические указания.**

**Задание**

1. Сгенерируйте (используя генератор случайных чисел) матрицу смежности для неориентированного графа *G*. Выведите матрицу на экран.
2. Для сгенерированного графа осуществите процедуру обхода в глубину, реализованную в соответствии с приведенным выше описанием.

**Описание метода решения задачи:**

**Вход**: G – матрица смежности графа.

**Выход**: номера вершин в порядке их прохождения на экране.

**Алгоритм ПОГ**

1.1. для всех i положим NUM[i] =  False пометим как "не посещенную";

1.2. **ПОКА** существует "новая" вершина v

1.3. **ВЫПОЛНЯТЬ** DFS (v).

**Алгоритм** DFS(v):

2.1. пометить v как "посещенную" NUM[v] = True;

2.2. вывести на экран v;

2.3. **ДЛЯ** i = 1 **ДО** size\_G **ВЫПОЛНЯТЬ**

2.4.   **ЕСЛИ**  G(v,i) = = 1**И** NUM[i] = = False

2.5. **ТО**

2.6. {

2.7.     DFS(i);

2.8. }

**Листинг:**

**Задание:**

#include <iostream>

#include <locale.h>

void dfs(int\* (\*mass), bool\* be, int pos, int n)

{

if (be[pos] == false)

{

be[pos] = true;

if (pos != 0) { printf("->"); }

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

}

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

{

if (mass[pos][i] != 0 && be[i] == false)

{

dfs(mass, be, i, n);

}

}

}

int main()

{

setlocale(LC\_ALL, "Rus");

int\*\* a, n, k = 0;

int i = 0, j = 0, w;

printf("Введите кол-во строк и столбцов\n");

scanf\_s("%d", &n);

a = (int\*\*)malloc(n \* sizeof(int\*));

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

{

a[i] = (int\*)malloc(n \* sizeof(int));

}

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

{

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

{

//w = rand() % 2;

/\*if ((w == 1)) || (w == 2))

{

a[i][j] = 1;

a[j][i] = 1;

}

else {

a[i][j] = 0;

a[j][i] = 0;

}

if (i == j)

{

a[i][j] = 0;

}\*/

if (i == j) { a[i][j] = 0; }

else

{

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

a[j][i] = a[i][j];

}

}

}

for (int i = 0; i < n + 1; i++)

{

for (int j = 0; j < n + 1; j++)

{

if (i == 0 || j == 0)

{

if (i == 0) { printf("%d ", j); }

else { printf("%d", i); }

}

else

{

printf(" %d", a[i - 1][j - 1]);

}

}

printf("\n");

}

int pos = 0;

bool\* be = (bool\*)malloc(n \* sizeof(bool));

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

{

be[i] = false;

}

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

{

if (be[i] == false)

{

dfs(a, be, i, n);

}

}

free(a);

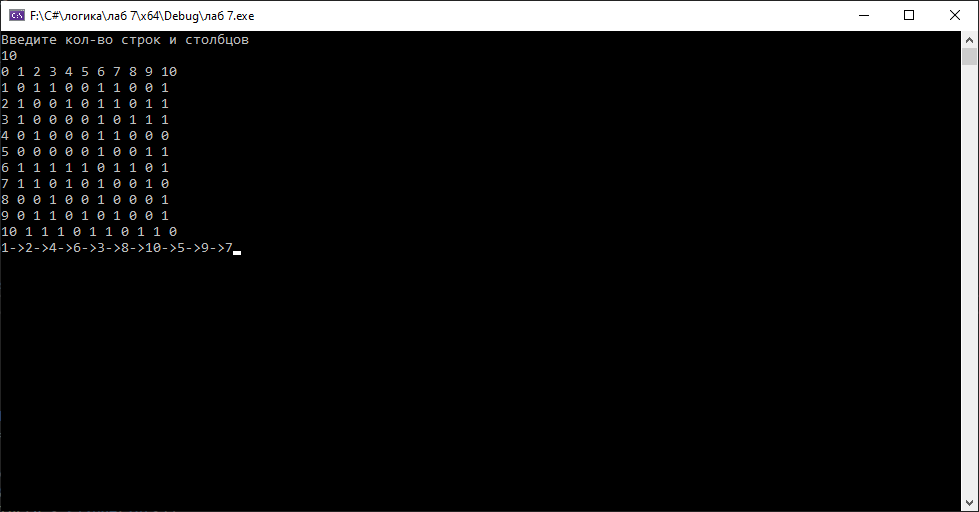
getchar();

getchar();

}

**Результаты работы программы:**

**Задание:**

****

**Вывод:** в ходе выполнения лабораторной работы мы выполнили обход графа в глубину. Результаты работы программ совпали с результатами трассировки, следовательно программы работают без ошибок.