Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №4

по курсу «Логика и основы алгоритмизации в инженерных задачах»

## на тему «Обход графа в глубину»

Выполнил:

студент группы 19ВВ2

Гусев Д. О.

Приняли:

Митрохин М. А.

Юрова О.В.

Пенза 2020

Цель работы

Выполнить алгоритм обхода графа в глубину.

Лабораторное задание

### **Задание 1**

### 1. Сгенерируйте (используя генератор случайных чисел) две матрицу

### смежности для неориентированного графа G. Выведите сгенерированные

### матрицы на экран.

### 2.Для сгенерированного графа осуществите процедуру обхода в

### глубину, реализованную в соответствии с приведенным выше описанием.

### Задание 2\*

### 1. Для матричной формы представления графов выполните преобразование рекурсивной реализации обхода графа к не рекурсивной.

### **Листинг**

#define CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <iostream>

#include <stack>

#include <time.h>

using namespace std;

void dfs\_n(int v, int\*\* p1, int n, int\* vis)

{

printf("Нерекурсивный");

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

vis[i] = 0;

}

stack <int> ss;

ss.push(v);

vis[v] = 1;

printf(">> %d\n", v+1);

while (!ss.empty())

{

v = ss.top();

ss.pop();

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

{

if ((p1[v][i] == 1) && (vis[i] == 0))

{

vis[i] = 1;

ss.push(i);

printf(">> %d\n", i + 1);

}

}

}

}

void dfs(int v, int\*\* p1, int n, int\* vis)

{

vis[v] = true;

printf(">> %d\n", v + 1);

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

{

if ((p1[v][i] == 1) and (vis[i] == 0))

{

dfs(i, p1, n, vis);

}

}

}

void vivod(int n, int\*\* p) {

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

{

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

{

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

}

printf("\n");

}

}

int main() {

setlocale(LC\_ALL, "Rus");

int i = 0, j = 0, n, v;

printf("Введите размер генерируемой матрицы: >> ");

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

int\*\* p1, \* vis;

srand(time(NULL));

p1 = (int\*\*)malloc(n \* sizeof(int\*)); //выделение памяти под всё, заполнение и отзеркаливание

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

{

p1[i] = (int\*)malloc(n \* sizeof(int));

}

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

for (i = 0; i < n; i++) //заполнение массива p1

{

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

{

if (i == j)

{

p1[i][j] = 0;

}

else

{

p1[i][j] = rand() % 2;

}

}

}

for (i = 0; i < n; i++) //отзеркаливание матрицы

{

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

{

p1[j][i] = p1[i][j];

}

}

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

vis[i] = 0;

}

printf("Введите вершину с которой начнется обход: >> ");

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

v -= 1;

printf("Рекурсивный\n");

dfs(v, p1, n, vis);

dfs\_n(v, p1, n, vis);

vivod(n, p1);

system("pause");

}

**Результаты работы программы**



## Вывод

Освоили алгоритм обхода в глубину. Реализовали его рекурсивный и не рекурсивный способ реализации.