Министерство высшего образования и науки

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 21ВВ2:

Хабибулин А.М.

Щеглов Д.А.

Приняли:

Митрохин М.А.

Юрова О.В.

**Задание 1**

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.

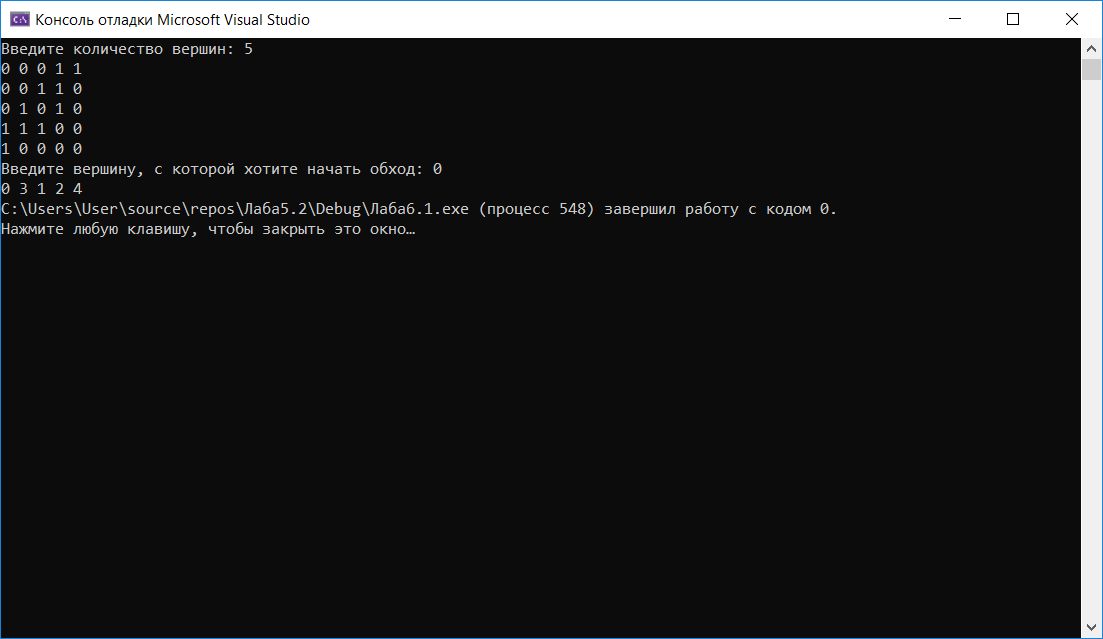
3.\* Реализуйте процедуру обхода в глубину для графа, представленного списками смежности.

**Задание 2\***

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

**Ход работы:**

Сгенерировали с помощью генератора случайных чисел матрицу  
смежности для неориентированного графа. Матрица выводится на экран.  
 Осуществили процедуру обхода графа в глубину. Мы выбираем вершину, с которой начинается обход графа. Затем осуществляется переход на первую встречающуюся вершину. Пройденные вершины отмечаются и программа их больше не обходит. Как только в процессе обхода переход на следующую вершину становится невозможен, происходит возврат на один шаг назад и ищется следующий вариант продвижения.



**Результат работы программы - рис.1**

**Листинг**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <locale.h>

void obhod(int num, int\* visited, int\*\* M, int dlin) //обход графа в глубину

{

visited[num] = 1;

printf("%d ", num);

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

if (M[num][i] == 1 && visited[i] == 0) {

obhod(i, visited, M, dlin);

}

}

}

int main(void)

{

setlocale(LC\_ALL, "Rus");

int\*\* a;

int N;

printf("Введите количество вершин: ");

scanf("%d", &N);

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

srand(time(NULL));

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

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

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

b[j] = 0;

}

a[i] = b;

}

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

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

if (i != j) {

int num = rand() % 2;

a[i][j] = num;

a[j][i] = num;

}

}

}

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

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

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

}

printf("\n");

}

int dlin = N;

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

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

visits[i] = 0;

}

printf("Введите вершину, с которой хотите начать обход: ");

scanf("%d", &N);

obhod(N, visits, a, dlin);

getchar();

}

Сделали не рекурсивный обход графа в глубину.

void DFS(int num, int\* visited, int\*\* M, int dlin)

{

using namespace std;

stack <int> s;

s.push(num);

visited[num] = 1;

while (!s.empty())

{

int num = s.top();

s.pop();

printf("%d ", num);

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

{

if (M[num][i] == 1 && visited[i] == 0)

{

s.push(i);

visited[i] = 1;

}

}

}

}

**Вывод:**

В ходе работы была написана программа, которая генерирует неориентированный граф и осуществляет его обход в глубину.  Идея такого обхода состоит в том, чтобы начав обход из какой-либо вершины всегда переходить по первой встречающейся в процессе обхода связи в следующую вершину, пока существует такая возможность. Как только в процессе обхода исчерпаются возможности прохода, необходимо вернуться на один шаг назад и найти следующий вариант продвижения. Таким образом, итерационно выполняя описанные операции, будут пройдены все доступные для прохождения вершины. Чтобы не заходить повторно в уже пройденные вершины, необходимо их пометить как пройденные.