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»

**Выполнили:**

студенты группы 20ВВ3

Пантюшов Егор

Шмелёв Данила

**Приняли:**

Юрова О.В.

Митрохин М.А.

Пенза 2021

# Название

Обход графа в глубину

**Цель работы** – изучение алгоритма обхода графа в глубину.

# Методические указания

Обход графа – одна из наиболее распространенных операций с

графами. Задачей обхода является прохождение всех вершин в графе.

Обходы применяются для поиска информации, хранящейся в узлах графа,

нахождения связей между вершинами или группами вершин и т.д.

# Лабораторное 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.

**Псевдокод**

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 &quot;не посещенную&quot;;

1.2. ПОКА существует &quot;новая&quot; вершина v

1.3. ВЫПОЛНЯТЬ DFS (v).

Алгоритм DFS(v):

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

2.2. вывести на экран v;

2.3. ДЛЯ i = 1 ДО size\_G ВЫПОЛНЯТЬ

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

2.5. ТО

2.6. {

2.7. DFS(i);

2.8. }

# Листинг

#define \_CRT\_NONSTDC\_NO\_WARNINGS

#define \_CRT\_SECURE\_NO\_WARNINGS

#define HEADER ("Лабораторная работа №4\nВыполнили: Шмелёв Д. и Пантюшов Е.\n")

#include <stdio.h>

#include <locale.h>

#include <stdlib.h>

#include <time.h>

#include <stdbool.h>

void dfs(int\*\* G, int\* NUM, int v, int n)

{

NUM[v] = true;

printf(" %d ", v);

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

{

if ((G[v][i] == 1) && (NUM[i] == false))

{

dfs(G, NUM, i, n);

}

}

}

void task\_1\_1(int\*\* G, int n)

{

printf("\nЗадание 1.\n\nПункт 1.\nМатрица смежности:\n");

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

{

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

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

{

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

}

}

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

{

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

{

if (i == j)

{

G[i][j] = 0;

}

else

{

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

}

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

}

printf("\n");

}

}

void task\_1\_2(int\*\* G, int n)

{

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

int v;

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

{

NUM[i] = false;

}

printf("\nПункт 2.\nВведите точку входа: ");

scanf("%d", &v);

printf("Результат работы алгоритма обхода в глубину(матрица): ");

dfs(G, NUM, v, n);

free(NUM);

}

int main()

{

setlocale(LC\_ALL, "RUS");

srand(time(NULL));

printf(HEADER);

int n;

printf("Введите размерность матрицы: ");

scanf("%d", &n);

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

task\_1\_1(G, n);

task\_1\_2(G, n);

free(G);

system("PAUSE");

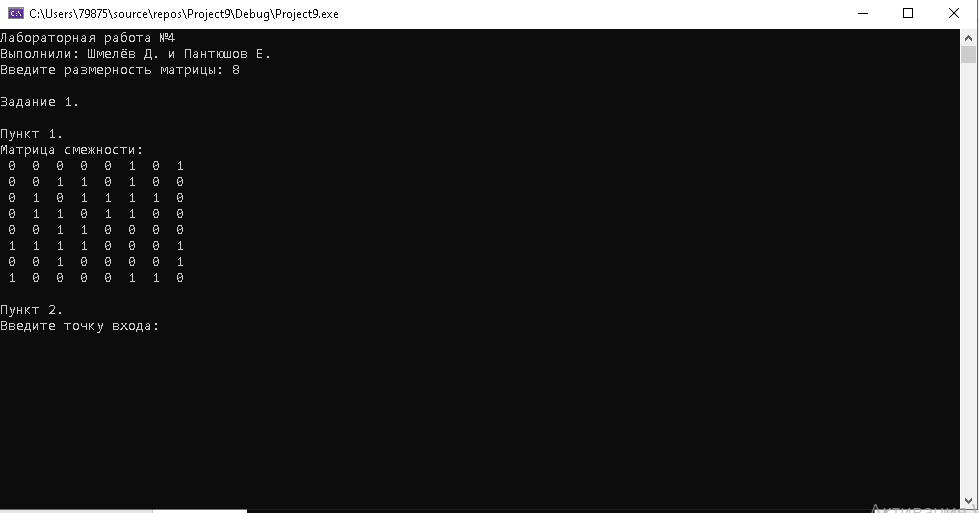
return 0;

}

# Результат работы программы

**Пункт 1.**

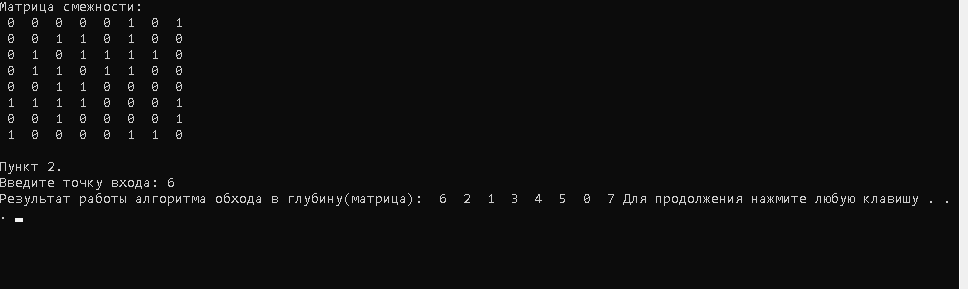
Результаты работы программы показаны на рисунке 1.



**Рисунок 1 – Результат работы программы**

**Пункт 2.**

Результаты работы программы показаны на рисунке 2.



**Рисунок 2 – Результат работы программы**

# Вывод

В ходе выполнения лабораторной работы была разработана программа, в который был реализован алгоритм обхода графа в глубину.