Министерство 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 24ВВВ1:

Кодиров А. Н.

Качкуркин А.В.

Приняли:

Юрова О. В.

Деев М. В.

Пенза 2025

**Лабораторное задание:**

### 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.

### ****Теоретическая часть****

#### **1. Суть алгоритма обхода графа в глубину (DFS)**

**Обход в глубину (Depth-First Search, DFS)** — это фундаментальный алгоритм для обхода или поиска в структурах данных "граф" и "дерево". Идея алгоритма заключается в том, чтобы, начиная с выбранной вершины (стартовой), двигаться вглубь графа, насколько это возможно, прежде чем вернуться назад (произвести **backtracking**).

**Ключевые принципы:**

* **Рекурсия или Стек:** Алгоритм естественным образом реализуется с помощью рекурсии или явного использования стека (LIFO — "последним пришел, первым ушел"). Это позволяет запоминать вершины, из которых нужно продолжить обход после достижения "тупика".
* "Погружение вглубь": Алгоритм стремится идти по одному пути, пока не посетит все вершины на этом пути, и только затем переходит к следующим путям.
* **Отслеживание посещений:** Чтобы не зациклиться и не посещать вершины повторно, используется массив visited (или visited), который отмечает уже пройденные вершины.

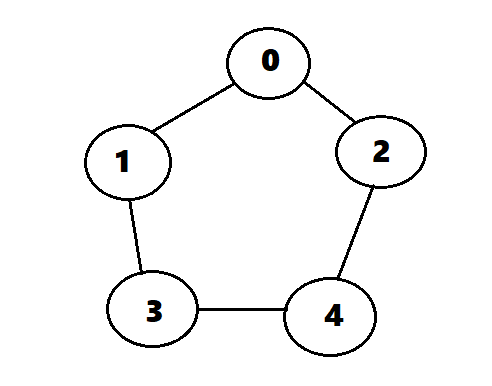
#### **Псевдокод алгоритма DFS (рекурсивная версия)**

1. **Вход**: G – матрица смежности графа.  
   **Выход**: номера вершин в порядке их прохождения на экране.
2. **Алгоритм ПОГ**  
   1.1. ВВЕСТИ numG (количество вершин)  
   1.2. visited[i] = 0 для всех i от 0 до numG-1  
   1.3. ВВЕСТИ v (стартовая вершина)  
   1.4. ВЫПОЛНЯТЬ DFS(v)
3. **Алгоритм DFS(v)**:  
   2.1. visited[v] = 1 (пометить v как "посещенную")  
   2.2. ВЫВЕСТИ v  
   2.3. ДЛЯ i = 0 ДО numG-1 ВЫПОЛНЯТЬ  
   2.4.      ЕСЛИ G[v][i] == 1 И visited[i] == 0  
   2.5.      ТО  
   2.6.      {  
   2.7.           DFS(i)  
   2.8.      }

#### **Визуализация процесса обхода**

Рассмотрим простой граф из 5 вершин и его матрицу смежности.

**Граф:**



**Матрица смежности:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 |
| 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 |
| 2 | 1 | 0 | 0 | 0 | 1 |
| 3 | 0 | 1 | 0 | 0 | 1 |
| 4 | 0 | 0 | 1 | 1 | 0 |

**Листинг**

**#define \_CRT\_SECURE\_NO\_WARNINGS**

**#include <stdio.h>**

**#include <conio.h>**

**#include <stdlib.h>**

**#include <iostream>**

**#include <locale.h>**

**void DFS(int\*\* G, int numG, int\* visited, int s) {**

**visited[s] = 1;**

**printf("%3d", s);**

**for (int i = 0; i < numG; i++) {**

**if (G[s][i] == 1 && visited[i] == 0)**

**DFS(G, numG, visited, i);**

**}**

**}**

**int main()**

**{**

**setlocale(LC\_ALL, "");**

**int\*\* G;**

**int\* visited;**

**int numG;**

**int current;**

**printf("=================================\n");**

**printf(" ALGORITHM DFS (Depth First Search)\n");**

**printf("=================================\n");**

**printf("Количество вершин: ");**

**scanf("%d", &numG);**

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

**visited = (int\*)malloc(numG \* sizeof(int));**

**for (int i = 0; i < numG; i++) {**

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

**visited[i] = 0;**

**}**

**printf("\nМатрица смежности:\n");**

**printf("-------------------\n");**

**for (int i = 0; i < numG; i++) {**

**for (int j = 0; j < numG; j++) {**

**G[i][j] = (i == j ? 0 : rand() % 2);**

**}**

**}**

**for (int i = 0; i < numG; i++) {**

**for (int j = 0; j < numG; j++) {**

**printf("%3d", G[i][j]);**

**}**

**printf("\n");**

**}**

**printf("\n-------------------\n");**

**printf("Стартовая вершина: ");**

**scanf("%d", &current);**

**printf("\n");**

**printf("Путь обхода: ");**

**DFS(G, numG, visited, current);**

**printf("\n\n");**

**for (int i = 0; i < numG; i++) {**

**free(G[i]);**

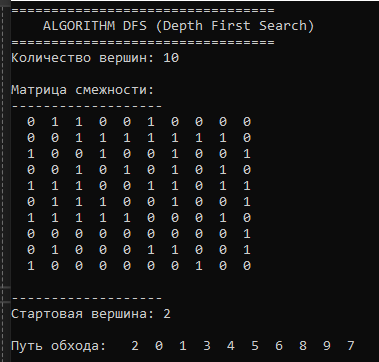
**}**

**free(G);**

**free(visited);**

**return 0;**

**}**

****

**Вывод:** В ходе лабораторной работы мы научились реализовывать алгоритм поиска в глубину (DFS) для обхода графов, включая создание матрицы смежности, организацию рекурсивного обхода вершин, отслеживание посещенных узлов и визуализацию процесса обхода графа.