МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
Кафедра «Вычислительной техники»

**ОТЧЁТ**

по лабораторной работе №7  
по курсу «Логика и основы алгоритмизации в инженерных задачах»  
на тему «Обход графа в глубину»

Выполнили:

Студенты группы 23ВВВ2

Лисов Е.А.

Кочегин В.В.

Приняли:

Митрохин М. А.  
Юрова О.В.

Пенза 2024

**Цель работы**

Изучение обхода графа в глубину. Закрепление навыков его генерации, реализации обхода.

**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.

**Ход работы**

**Задание 1.**

1. Для генерации матрицы смежности для неориентированного графа был разработан код:

void createG(int\*\* G, int N) {

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

for (int j = i + 1; j < N; j++) {

int edge = rand() % 2;

G[i][j] = edge;

G[j][i] = edge;

}

}

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

G[i][i] = 0;

}

}

2. для сгенерированного графа была осуществлена процедура обхода в глубину:

void DFS(int\*\* G, int N, int\* NUM, int v) {

NUM[v] = 1;

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

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

if (G[v][i] == 1 && NUM[i] == 0) {

DFS(G, N, NUM, i);

}

}

}

3.\* Реализована процедура обхода в глубину для графа, представленного списками смежности:

void DFS(Node\*\* adjList, int N, int\* NUM, int v) {

NUM[v] = 1;

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

Node\* temp = adjList[v];

while (temp != NULL) {

if (NUM[temp->vertex] == 0) {

DFS(adjList, N, NUM, temp->vertex);

}

temp = temp->next;

}

}

**Задание 2.**

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

void DFS(int\*\* G, int N, int\* NUM, int v) {

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

int top = -1;

stack[++top] = v;

while (top != -1) {

int vec = stack[top--];

if (NUM[vec] == 0) {

NUM[vec] = 1;

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

}

for (int i = N - 1; i >= 0; i--) {

if (G[vec][i] == 1 && NUM[i] == 0) {

stack[++top] = i;

}

}

}

}

### Результаты работы программы

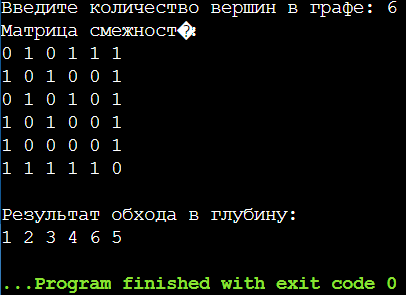


Рисунок 1 — Результаты работы программы для задания 1. Подпункт 1,2

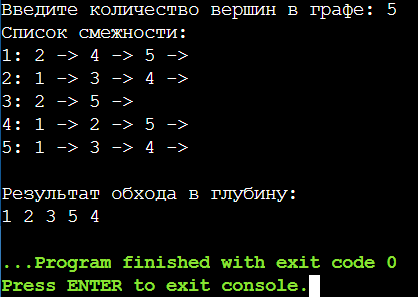


Рисунок 2 — Результаты работы программы для задания 1. Подпункт 3

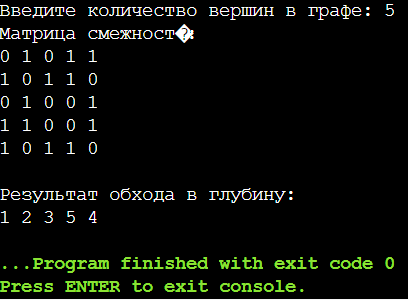


Рисунок 3 — Результаты работы программы для задания 2

### Вывод

В ходе выполнения работы были изучен принцип обхода графа в ширину и его реализация (не рекурсивная и рекурсивная).