Министерство 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 20ВВ2

Барсуков Н.И.

Проверили:

Митрохин М.А.

Юрова О.В.

Пенза 2021

**Задание 1:**

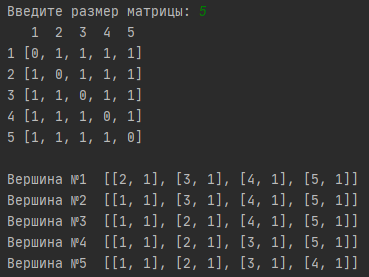
1. Сгенерируйте (используя генератор случайных чисел) матрицу смежности для неориентированного графа G. Выведите матрицу на экран.

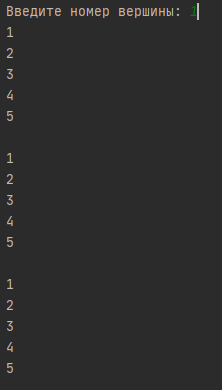
2. Для сгенерированного графа осуществите процедуру обхода в глубину, реализованную в соответствии с приведенным выше описанием.

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 рекрсивной.





**Листинг**

import Graph  
import Operations  
  
n = int(input("Введите размер матрицы: "))  
  
graph = []  
line = []  
Graph.generator\_matrix(graph, n)  
Graph.print\_matrix(graph, 0, n)  
Graph.generator\_line(line, graph, n)  
  
Visits = [0 for \_ in range(n)]  
  
node = int(input("Введите номер вершины: "))  
Operations.depth\_first\_search\_matrix(graph, Visits, node-1)  
print()  
  
Visits = [0 for \_ in range(n)]  
Operations.depth\_first\_search\_line(line, Visits, node-1)  
print()  
  
Visits = [0 for \_ in range(n)]  
Operations.depth\_first\_search\_stack(graph, Visits, node-1)

**Функция depth\_first\_search\_matrix**

def depth\_first\_search\_matrix(graph, visits, node):  
 print(node+1)  
 visits[node] = 1  
 for j in range(len(graph)):  
 if graph[node][j] == 1 and visits[j] == 0:  
 depth\_first\_search\_matrix(graph, visits, j)

**Функция depth\_first\_search\_line**

def depth\_first\_search\_line(graph, visits, node):  
 print(node+1)  
 visits[node] = 1  
 for j in graph[node]:  
 if visits[j[0]-1] == 0:  
 depth\_first\_search\_line(graph, visits, j[0]-1)

**Функция depth\_first\_search\_stack**

def depth\_first\_search\_stack(graph, visits, node):  
 stack = [node]  
 while len(stack) > 0:  
 item = stack.pop()  
 if visits[item] == 0:  
 visits[item] = 1  
 print(item+1)  
 for i in range(len(graph[item])-1, -1, -1):  
 if graph[item][i] == 1:  
 stack.append(i)