Министерство науки и высшего 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 22BВВ1

Митрошин Ю.Е

Коннов А.Д

Приняли:

Акифьев И.В

Юрова О.В

Пенза 2023

**Название**

Обход графа в глубину

**Общие сведения.**

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.

Одним из способов обхода графов является поиск 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.

Таким образом, можно предложить следующую рекурсивную реализацию алгоритма обхода в глубину.

**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 "не посещенную";

1.2. **ПОКА** существует "новая" вершина v

1.3. **ВЫПОЛНЯТЬ** DFS (v).

**Алгоритм** DFS(v):

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

2.2. вывести на экран v;

2.3. **ДЛЯ** i = 1 **ДО**size\_G**ВЫПОЛНЯТЬ**

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

2.5. **ТО**

2.6. {

2.7.     DFS(i);

2.8. }

Реализация состоит из подготовительной части, в которой все вершины помечаются как не помеченные (п.1.1) и осуществляется запуск процедуры обхода для вершин графа (п.1.2, 1.3). И непосредственно процедуры обхода, которая помечает текущую (т.е. ту, в которой на текущей итерации находится алгоритм) вершину как посещенную (п. 2.1). Затем выводит номер текущей вершины на экран (п.2.2) и в цикле просматривает **v**-ю строку матрицы смежности графа G(v,i). Как только алгоритм встречает смежную с **v** не посещенную вершину (п.2.4), то для этой вершины вызывается процедура обхода (п.2.7).

Например, пусть дан граф (рисунок 1), заданный в виде матрицы смежности:

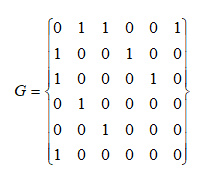
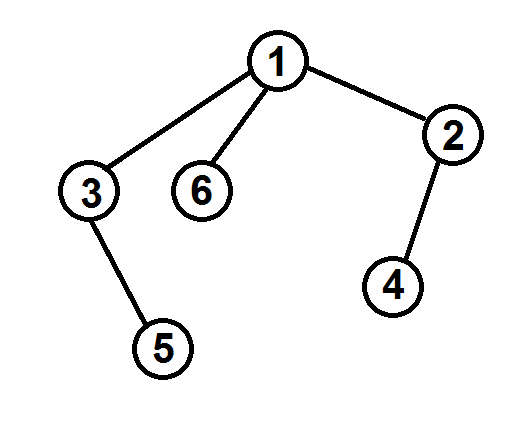
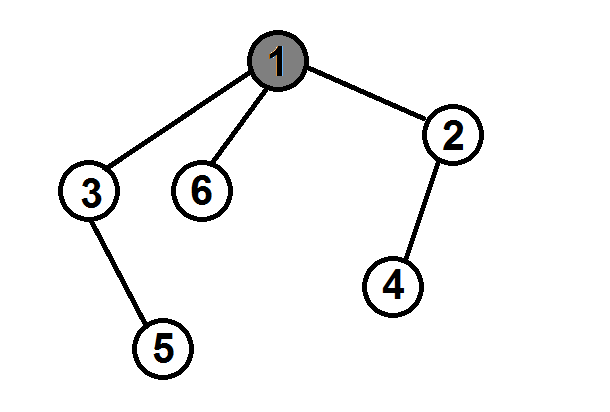


Рисунок 1 – Граф

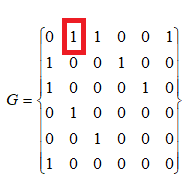
Тогда, если мы начнем обход из первой вершины, то на шаге 2.1 она будет помечена как посещенная (NUM[1] = True), на экран будет выведена единица.



https://lh7-us.googleusercontent.com/6tKo08tVP66d9eaCkZo_LqO0lsVjmZycXLqZcPu2Ty1UKsErmuJ64W2h7HZUPJSwOrtiqHyKHBU4yrkwyWRRFwog7Yi-74vLdowCbCmZscACvenRfm12DVQjsIG112-CHkUphbrxKx2m9ETxI9r-ug

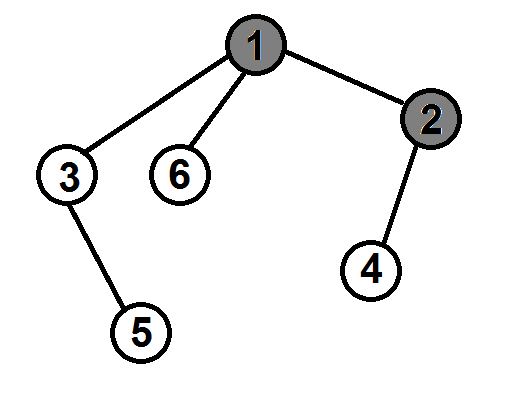
Рисунок 2 – Вызов DFS(1)

При просмотре 1-й строки матрицы смежности



будет найдена смежная вершина с индексом 2 (G(1,2) = =1), которая не посещена (NUM[2] = = False) и будет вызвана процедура обхода уже для нее -  DFS(2).

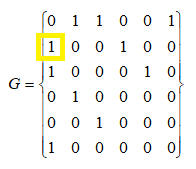
На следующем вызове на шаге 2.1  вершина 2 будет помечена как посещенная (NUM[2] = True), на экран будет выведена двойка.



https://lh7-us.googleusercontent.com/7tSNBQRSrY0iD_j1Q6PZsfY_QPY8XPlGo2NO26kzHAj87tFj6XHHyD1RvkHNBF-G1vKp5zStpLt2GfxRoQehuFWQ42LX8Zt6m0mzcVHWOd82GoB33g6SIOoGf63PhxlkRft0OE09_-8PGXsbF0yP0g

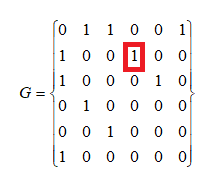
Рисунок 3 – Вызов DFS(2)

И алгоритм перейдет к просмотру второй строки матрицы смежности. Первая смежная с вершиной 2 - вершина с индексом 1(G(2,1) = =1),

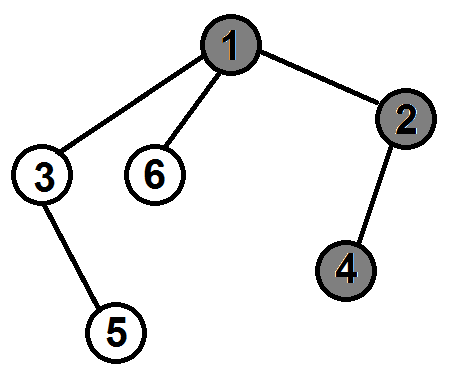


которая к настоящему моменту уже посещена (NUM[1] = = True) и процедура обхода для нее вызвана не будет. Цикл 2.3 продолжит просмотр матрицы смежности.

Следующая найденная вершина, смежная со второй, будет иметь индекс 4 (G(2,4) = =1), она не посещена (NUM[4] = = False) и для нее будет вызвана процедура обхода -  DFS(4).



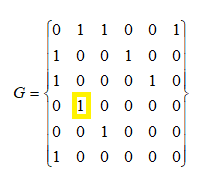
Вершина 4 будет помечена как посещенная (NUM[4] = True), на экран будет выведена четверка.



https://lh7-us.googleusercontent.com/I5tX9HraaY0BLtmX55GwCxwlu3HKOAsKfvDE3hMPyjiulAqraqJMVNdqamQvV-UazsQyH0Ut4lLwmqO2T-gtMMc7upjMz7j9xc-j0xJY4o221at88DAdo7EZEnApg1-hFqq2z1U84pBZk54n21fCfw

Рисунок 4 – Вызов DFS(4)

При просмотре 4-й строки матрицы будет найдена вершина 2, но она уже посещена (NUM[2] = True), поэтому процедура обхода вызвана не будет.

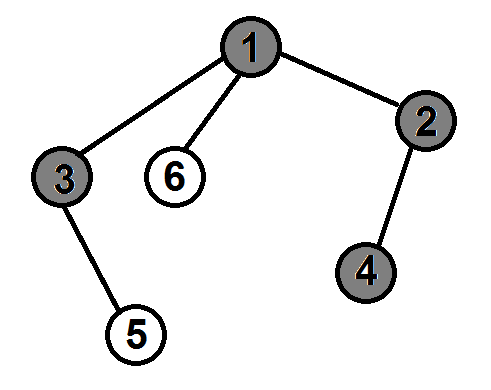


Цикл 2.3 завершится и для текущего вызова DFS(4) процедура закончит свою работу, вернувшись к точке вызова, т.е. к моменту просмотра циклом 2.3 строки с индексом 2 для вызова DFS(2).

В вызове DFS(2) цикл 2.3 продолжит просмотр строки 2 в матрице смежности, и, пройдя её до конца завершится. Вместе с этим завершится и  вызов процедуры DFS(2), вернувшись к точке вызова - просмотру циклом 2.3 строки с индексом 1 для вызова DFS(1).

При просмотре строки 1 циклом 2.3 в матрице смежности будет найдена следующая не посещенная, смежная с 1-й, вершина с индексом 3 (G(1,2) = =1 и NUM[3] = = False) и для нее будет вызвана DFS(3).

Вершина 3 будет помечена как посещенная (NUM[3] = True), на экран будет выведена тройка.



https://lh7-us.googleusercontent.com/QseY_9k56mwORO8LLTexyB2A-7zJx2MnbHKfPmU--mPOJu-gp0943mIJjGrK-A_22OmyciibZLSJvmUeJ3q3uuIMBn-TXloeSsaWbXxqN1okRo3yOxGiTEduJ0xjFIk4m2s0JwIF4ezoNvksrqkZYw

Рисунок 4 – Вызов DFS(3)

Работа алгоритма будет продолжаться до тех пор, пока будут оставаться не посещенные вершины, т.е. для которых NUM[i] = = False.

В конце работы алгоритма все вершины будут посещены. А на экран будут выведены номера вершин в порядке их посещения алгоритмом.



https://lh7-us.googleusercontent.com/9w44Eeh6IpjDvG1jYYDsityA-cyvIzJC-hARlK1rhpy071O97tAQGxjeSuvi2QNbWIXKuwDxCPpNZ7g0ELKUWvbw3XaiQEf6lTCCGIzTuu5hrbZqdp1u5YvxXKZ83BYg3mQIhQWkUbBK-9G0bdzUbg

Рисунок 5 – Результат работы обхода

**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.

**Листинг**

import random

size1 = int(input("Введите размер матрицы M1: "))

M1 = []

for i in range(size1):

row = []

for j in range(size1):

row.append(0)

M1.append(row)

for i in range(size1):

for j in range(i + 1, size1):

M1[i][j] = M1[j][i] = random.randint(0, 1)

lists = [[] for \_ in range(size1)]

for i in range(size1):

for j in range(size1):

if M1[i][j] == 1:

lists[i].append(j)

def epth\_first\_search(G):

def DFS(v):

NUM[v] = True

print(v, end=' ')

for i in range(size\_G):

if G[v][i] == 1 and not NUM[i]:

DFS(i)

size\_G = len(G)

NUM = [False] \* size\_G

if not NUM[0]:

DFS(0)

def depth\_first\_search\_iterative(adj\_matrix):

num\_vertices = len(adj\_matrix)

visited = [False] \* num\_vertices

stack = [0]

visited[0] = True

print(0, end=' ')

while stack:

current\_vertex = stack[-1]

found = False

for i in range(num\_vertices):

if adj\_matrix[current\_vertex][i] == 1 and not visited[i]:

stack.append(i)

visited[i] = True

print(i, end=' ')

found = True

break

if not found:

stack.pop()

def depth\_first\_search(adj\_list):

def DFS(vertex):

visited[vertex] = True

print(vertex, end=' ')

for neighbor in adj\_list[vertex]:

if not visited[neighbor]:

DFS(neighbor)

num\_vertices = len(adj\_list)

visited = [False] \* num\_vertices

if not visited[0]:

DFS(0)

print("Матрица смежности для M1:")

for row in M1:

print(row)

print("\nСписок смежности для M1: ")

for i, verticles in enumerate(lists):

print(f"Вершина {i}: {verticles}")

print("Результат обхода в глубину:")

epth\_first\_search(M1)

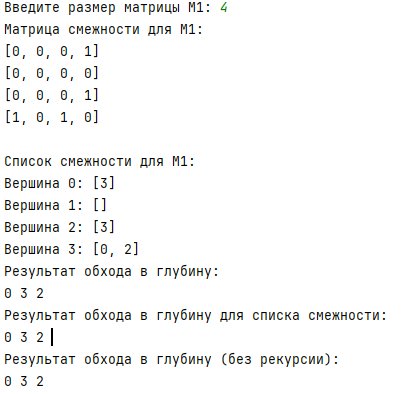
print("\nРезультат обхода в глубину для списка смежности:")

depth\_first\_search(lists)

print("\nРезультат обхода в глубину (без рекурсии):")

depth\_first\_search\_iterative(M1)

**Результат работы программы:**

****

**Вывод:** в ходе лабораторной работы мы научились работать с операцией обхода графов в глубину.