Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №5

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «Обход графа в ширину»

Выполнил:

студент группы 20ВВ2

Барсуков Н.И.

Проверили:

Митрохин М.А.

Юрова О.В.

Пенза 2021

**Задание 1:**

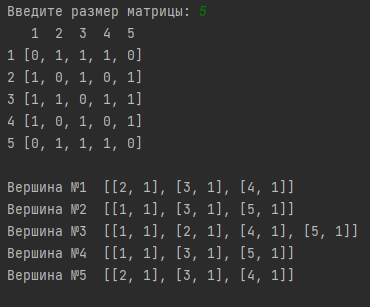
1. Сгенерируйте (используя генератор случайных чисел) матрицу смежности для неориентированного графа G. Выведите матрицу на экран.

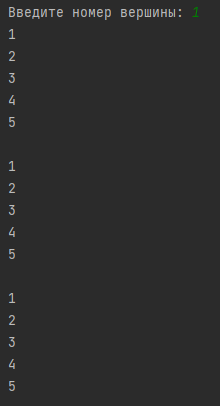
2. Для сгенерированного графа осуществите процедуру обхода в ширину, реализованную в соответствии с приведенным выше описанием.

3. \*Реализуйте процедуру обхода в ширину для графа, представленного списками смежности.

**Задание 2\*:**

1. Для матричной формы представления графов реализуйте алгоритм обхода в ширину с использованием очереди, построенной на основе структуры данных «список»





**Листинг**

import Graph  
import Operations  
  
n = int(input("Введите размер матрицы: "))  
  
graph = []  
# graph = [[0, 1, 1, 0, 0, 1],  
# [1, 0, 0, 1, 0, 0],  
# [1, 0, 0, 0, 1, 0],  
# [0, 1, 0, 0, 0, 0],  
# [0, 0, 1, 0, 0, 0],  
# [1, 0, 0, 0, 0, 0]]  
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.bfs\_matrix(graph, Visits, node-1)  
print()  
  
Visits = [0 for \_ in range(n)]  
Operations.bfs\_line(line, Visits, node-1)  
print()  
  
Visits = [0 for \_ in range(n)]  
Operations.bfs\_matrix\_real(graph, Visits, node-1)  
print()

**Функция bfs\_matrix**

def bfs\_matrix(graph, visits, node):  
 queue = [node]  
 while len(queue) > 0:  
 item = queue.pop(0)  
 visits[item] = 1  
 print(item+1)  
 for i in range(len(graph)):  
 if graph[item][i] == 1 and visits[i] == 0:  
 queue.append(i)  
 visits[i] = 1

**Функция bfs\_line**

def bfs\_line(graph, visits, node):  
 queue = [node]  
 while len(queue) > 0:  
 item = queue.pop(0)  
 visits[item] = 1  
 print(item+1)  
 for i in graph[item]:  
 if visits[i[0]-1] == 0:  
 queue.append(i[0]-1)  
 visits[i[0]-1] = 1

**Функция bfs\_matrix\_real**

def bfs\_matrix\_real(graph, visits, node):  
 queue = Queue()  
 queue.add(node)  
 while queue.len != 0:  
 item = queue.pop()  
 visits[item] = 1  
 print(item+1)  
 for i in range(len(graph)):  
 if graph[item][i] == 1 and visits[i] == 0:  
 queue.add(i)  
 visits[i] = 1