Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительной техники»

**Отчет**

По лабораторной работе №8

По курсу «Логика и основы алгоритмизации в инженерных задачах»

На тему «Определение характеристик графов»

Выполнили студенты гр.20ВВ4

Горбунов Н.А.

Погосян М.Д.

Проверили:

Акифьев И. В.

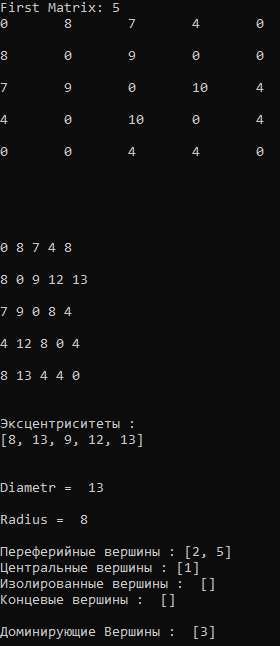
Юрова О. В.

Пенза, 2021

**Цель:** изучить определение характеристик графов.

**Задание 1:**

1. Сгенерировали (используя генератор случайных чисел) матрицу смежности для неориентированного взвешенного графа *G*. Вывели матрицу на экран.
2. Определили радиус и диаметр графа *G*, используя матрицу смежности графа.
3. Определили подмножества периферийных и центральных вершин графа *G*, используя матрицу смежности.
4. Нашли изолированные, концевые и доминирующие вершины.

Вывод программы:  


**Листинг задания 1:**

import random

import sys

G = []

c = []

N1 = int(input("First Matrix: "))

#[1,1,1,0,0,0,0,0,0,0]

chance = [0,0,0,1,1,1,1,1,1,1]

for i in range(N1):

G.append([])

for j in range(N1):

G[i].append(random.choice(chance))

if(G[i][j] == 1):

G[i][j] = random.randint(1,10)

if(i == j):

G[i][j] = 0

for i in range(N1):

for j in range(N1):

G[i][j] = G[j][i]

print(G[i][j],end='\t')

print('\n')

print('\n'\*3)

Isolirovannie = []

''' Поиск Изолированной '''

for i in range(len(G)):

count = 0

for j in range(len(G[i])):

count += 1

if(G[i][j] != 0):

break

if(G[i][j] == 0 and count == len(G)):

print(count)

Isolirovannie.append(i+1)

konc = []

''' Поиск концевой '''

for i in range(len(G)):

count = 0

for j in range(len(G[i])):

if(G[i][j] != 0):

count += 1

if(j == len(G)-1 and count == 1):

konc.append(i+1)

Domination = []

''' Поиск доминирующей '''

for i in range(len(G)):

count = 0

for j in range(len(G[i])):

if(G[i][j] != 0):

count += 1

if(j == len(G)-1 and count == len(G) - 1):

Domination.append(i+1)

Ex = []

j = 0

Diametr = 0

radius = sys.maxsize

for v in range(len(G)):

def BFS(v):

vis[v] = 0

queue.append(v)

while(bool(queue)):

s = queue[0]

#print(s+1)

queue.pop(0)

for i in range(len(G)):

if(G[s][i] > 0 and vis[i] > vis[s] + G[s][i]):

queue.append(i)

vis[i] = vis[s] + G[s][i]

vis = []

for i in range(len(G)):

vis.append(sys.maxsize)

queue = []

BFS(v)

for i in range(len(vis)):

if(vis[i] == sys.maxsize):

vis[i] = 0

Ex.append([])

for i in range(len(vis)):

print(vis[i],end=" ")

Ex[j].append(vis[i])

print("\n")

j += 1

Max = 0

for i in range(len(vis)):

if(vis[i] > Max):

Max = vis[i]

if(Max > Diametr and Max != 0):

Diametr = Max

if(Max < radius and Max != 0):

radius = Max

print("\nЭксцентриситеты : ")

Excentr = []

for i in range(len(Ex)):

Max = 0

for j in range(len(Ex[i])):

if(Ex[i][j] > Max):

Max = Ex[i][j]

Excentr.append(Max)

print(Excentr)

print("\n\nDiametr = ", Diametr)

print("\nRadius = ", radius)

per = []

middle = []

for i in range(len(Ex)):

if(Excentr[i] == Diametr):

per.append(i + 1)

if(Excentr[i] == radius):

middle.append(i + 1)

print("\nПереферийные вершины :" , per,"\nЦентральные вершины :", middle)

print("Изолированные вершины : ", Isolirovannie)

print("Концевые вершины : ", konc)

print("\nДоминирующие Вершины : ", Domination)

**Вывод:** изучили как определить радиус, диаметр и подмножества периферийных и центральных вершин графа. Также узнали, как определить изолированные, концевые и доминирующие вершины