МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра ТЭВН

Лабораторная работа № 4

“Определение числовых характеристик графов”

Вариант № 19

Факультет: ФЭН

Группа: ЭН2-31

Студент: Полозов А.А.

Преподаватель: Петрищев А.В.

Новосибирск 2024

# 1. Цель работы

Знакомство с понятием графа, видами графов, способами их представления, основными числовыми характеристиками, алгоритмами анализа и синтеза.

Получение навыков представления графов для вычисления их числовых характеристик и вычисление некоторых из них.

# 2. Задание

Условия задания приведены в таблице № 1.

Таблица № 1 – условия задания

|  |  |  |
| --- | --- | --- |
| **Вариант** | **Дано** | **Найти** |
| 7 | Случайный ненаправленный граф (число вершин не более 12, число рёбер не более 24).  Способ задания графа – матрица смежности | 1. Степени всех нечётных вершин.  2. Число компонент связности. |

# 3. Пояснения к заданию

## Математическая постановка задачи

**Граф** – это множество вершин, которые соединяются множеством рёбер.

**Матрица смежности** графа с конечным числом вершин , пронумерованных числами от до (нумерация начинается с нуля в связи с особенностями языка Python) – это квадратная целочисленная матрица размера , в которой значение элемента равно числу рёбер из -ой вершины графа в -ю вершину.

Из теории графов известно, что количество рёбер полного графа равно

## Описание алгоритма решения

**Генерация случайной матрицы смежностей :**

1. Генерация случайного натурального числа .
2. Объявление .
3. Пока :
   1. Генерация случайного натурального числа .
   2. Возвращение к 2.
4. Генерация пустой матрицы размера .
5. Объявление счётчика .
6. Пока :
   1. Генерация случайного натурального числа .
   2. Генерация случайного натурального числа .
   3. Если и , то:
      1. .
   4. Возвращение к 6.

**Определение степеней всех нечётных вершин:**

1. Объявление множества степеней .
2. Цикл со счётчиком
   1. Объявление переменной .
   2. Цикл со счётчиком
      1. .
   3. Добавить в конец списка переменную .
   4. Если :
      1. Написать: «Степень вершины равна »

**Определение числа компонент связности:**

1. Объявление списка компонент связности .
2. Объявление счётчика компонент .
3. Объявление списка посещённых вершин .
4. Объявление счётчика посещённых вершин .
5. Пока :
   1. Объявление счётчика порядкового номера вершины в -ой компоненте .
   2. Пока :
      1. .
      2. Цикл со счётчиком :
         1. Если :
            1. .
            2. .
      3. .
   3. .
   4. Пока :
      1. .
   5. Добавить к .
   6. .
   7. Добавить к в качестве нового элемента .
   8. .
6. Удалить у списка последний элемент.
7. Написать: «Число компонент равно ».

## Блок-схемы алгоритмов программы

Блок-схемы алгоритмов программы представлены на рис. № 1 – 3.

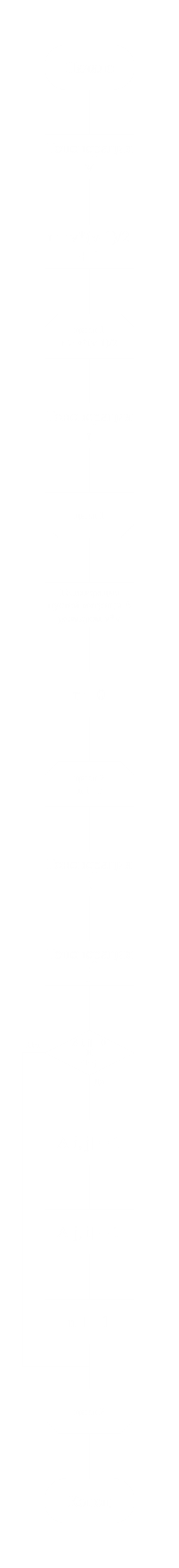


Рис. № 1 - Генерация случайной матрицы смежностей

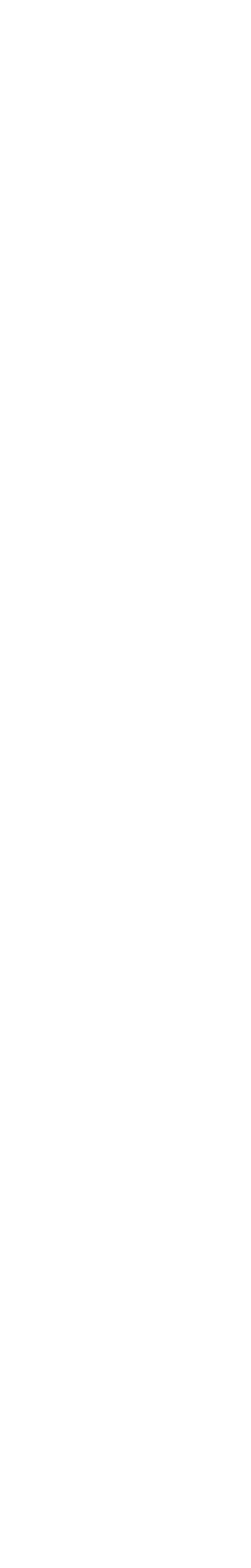


Рисунок № 2 - Определение степеней всех нечётных вершин

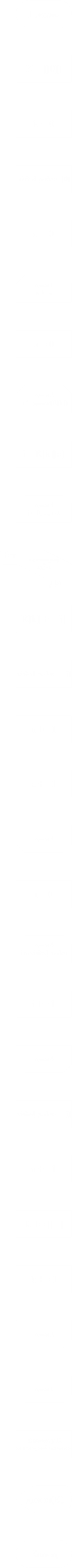


Рисунок № 3 - Определение числа компонент связности

## Листинг программы

import numpy as np

import random

## Генерация случайной матрицы смежностей A

# Генерация случайного натурального числа v

v = random.randint(1, 12)

# Генерация случайного натурального числа r

r = v\*(v-1)/2 + 1

while r > v\*(v-1)/2:

    r = random.randint(0, 24)

print('v =', v)

print('r =', r)

# Генерация пустой матрицы A размером v\*v

A = np.zeros((v, v), dtype=np.int64)

# Объявление счётчика n

n = 0

while n != r:

    # Гененерация случайного натурального числа i

    i = random.randint(0, v-1)

    # Гененерация случайного натурального числа j

    j = random.randint(0, v-1)

    if A[i, j] == 0 and i != j:

        A[i, j] = 1

        A[j, i] = 1

        n += 1

print(A)

## Определение степеней всех нечётных вершин

# Объявление множества степеней S

S = []

print('(Отсчёт вершин начинается с 0)')

for i in range(v):

    s = 0

    for j in range(v):

        s += A[i, j]

    S.append(s)

    if i % 2 != 0:

        print('Степень вершины', i, 'равна', S[i])

## Определение числа компонент связности

# Объявление списка компонент связности K

K = [[0]]

# Объявление счётчика компонент k

k = 0

# Объявление списка посещённых вершин visited\_nodes

visited\_nodes = [0]

# Объявление счётчика посещённых вершин n

n = 0

while n != v:

    # Объявление счётчика порядкового номера вершины в k-ой компоненте a

    a = 0

    while a != len(K[k]):

        i = K[k][a]

        for j in range(v):

            if not(j in visited\_nodes) and A[i, j] == 1:

                K[k] += [j]

                visited\_nodes += [j]

                n += 1

        a += 1

    i = 0

    while i in visited\_nodes:

        i += 1

    visited\_nodes += [i]

    n += 1

    K += [[i]]

    k += 1

K.pop()

print('Число компонент равно', len(K))