|  |
| --- |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **"МИРЭА – Российский технологический университет"**  **РТУ МИРЭА** |

Институт комплексной безопасности и специального приборостроения

Кафедра КБ-14 «Цифровые технологии обработки данных»

**ОТЧЕТ   
о выполнении лабораторной работы №3**

**«Алгоритмы на графах»**

**по дисциплине   
«Алгоритмы и структуры данных»**

**Вариант № 10**

Выполнил: студент 2 курса

группы \_\_\_\_\_\_\_\_\_\_\_

шифр \_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_

*(фио студента)*

Проверил: *\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Москва 2021 г.

**Вариант № 10.**

**Алгоритм:** транзитивная редукция ориентированного графа G = (V, Е)   
определяется как произвольный граф G' — (V, Е'), имеющий то же   
множество вершин, но с минимально возможным числом дуг (E’ ⊄   
E), транзитивное замыкание которого совпадает с транзитивным   
замыканием графа G, (причем если граф G ацикличен, то его   
транзитивная редукция единственна). Реализуйте программу   
транзитивной редукции графа

**Способ представления графа:** Матрица инцидентности

**Теория о Графах.**

Граф — это математический объект, который состоит из точек и линий, которые их соединяют. Точки называют вершинами графа, а линии — ребрами. Граф, ребра которого имеют направления, называется ориентированным, если же ребра графа не имеют направления, то такой граф называется неориентированным.



Рисунок 1. – Пример ориентированного графа.

Матрица смежности — это вид представления графа в виде матрицы, когда пересечение столбцов и строк задаёт дуги. Используя матрицу смежности, можно задать вес дуг и ориентацию. Каждая строка и столбец матрицы соответствуют вершинам, номер строки соответствует вершине, из которой выходит дуга, а номер столбца - в какую входит дуга. Пример матрицы смежности графа, изображенного на рисунке 1, представлен на рисунке 2.



Рисунок 2. – Пример матрицы смежности.

**Матрица инцидентности** — одна из форм представления графа, в которой указываются связи между инцидентными элементами графа (ребро(дуга) и вершина). Столбцы матрицы соответствуют ребрам, строки — вершинам. Ненулевое значение в ячейке матрицы указывает связь между вершиной и ребром (их инцидентность).

В случае ориентированного графа каждой дуге <x,y> ставится в соответствующем столбце: «1» в строке вершины x и «-1» в строке вершины y; если связи между вершиной и ребром нет, то в соответствующую ячейку ставится «0».

Например, для графа изображенного на рисунке 1 матрица инцидентности будет выглядеть так:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |
| A | 1 | 0 | 0 | 0 | 0 | -1 | 0 | 0 | -1 | 0 | 0 |
| B | -1 | 1 | 1 | 0 | 0 | 0 | -1 | 0 | 0 | -1 | 0 |
| C | 0 | -1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| D | 0 | 0 | -1 | -1 | 0 | 1 | 0 | 0 | 0 | 0 | -1 |
| E | 0 | 0 | 0 | 0 | -1 | 0 | 1 | 1 | 0 | 0 | 0 |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 0 | -1 | 1 | 1 | 0 |
| G | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Для транзитивной редукции графа будем в циклах проходить по трем вершинам и проверять их достижимость. Главное свойство тут, что, если . Исходя из этого можно сделать вывод, что если первые два условия выполняются, то наличие ребра не обязательно. Такие ребра мы и будем удалять из графа.

**Листинг программы.**

import copy

def read\_inc\_matrix(filename):

    with open(filename, 'r') as f:

        dim = int(f.readline().strip())

        matrix = [[0] \* dim for \_ in range(dim)]

        inc\_matrix = []

        for line in f.readlines():

            inc\_matrix.append(list(map(int, line.split())))

        for i in range(len(inc\_matrix[0])):

            first = None

            first\_e = 0

            second = None

            second\_e = 0

            for j in range(dim):

                if inc\_matrix[j][i] != 0:

                    if first is None:

                        first = j

                        if inc\_matrix[j][i] > 0:

                            first\_e = inc\_matrix[j][i]

                    else:

                        second = j

                        if inc\_matrix[j][i] > 0:

                            second\_e = inc\_matrix[j][i]

                    if first and second:

                        break

            if first\_e :

                matrix[first][second] = first\_e

            if second\_e:

                matrix[second][first] = second\_e

    return matrix

def transitive\_reduction(matrix):

    out\_matrix = copy.deepcopy(matrix)

    for i in range(len(matrix)):

        for j in range(len(matrix)):

            for k in range(len(matrix)):

                if (i, j) == (j, k) or (i, j) == (i, k):

                    continue

                if matrix[i][j] and matrix[j][k]:

                    out\_matrix[i][k] = 0

    return out\_matrix

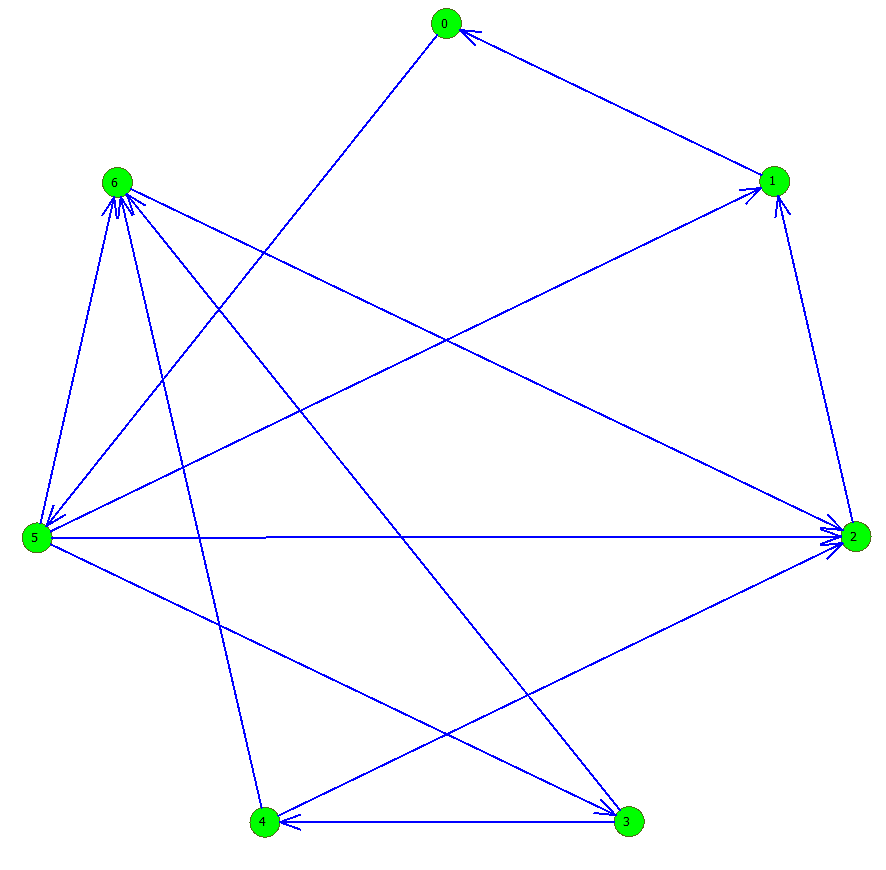
matrix = read\_inc\_matrix('graph\_task10.txt')

print("Исходная матрица смежности графа G:")

print("\n".join(" ".join(map(str, row)) for row in matrix))

print("Для заданного графа G матрица смежности графа G':")

print("\n".join(" ".join(map(str, row)) for row in transitive\_reduction(matrix)))

**Скриншот работы программы:**

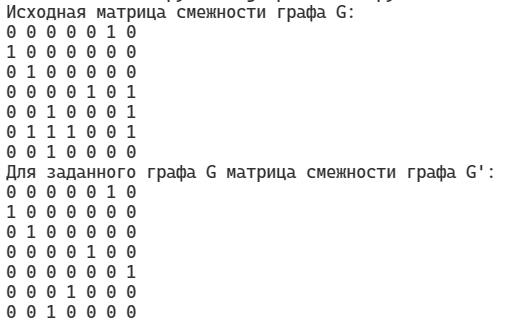
Рисунок 3. – Пример ориентированного ненагруженного графа.

Рисунок 4. – Пример работы программы для графа из рисунка 3.

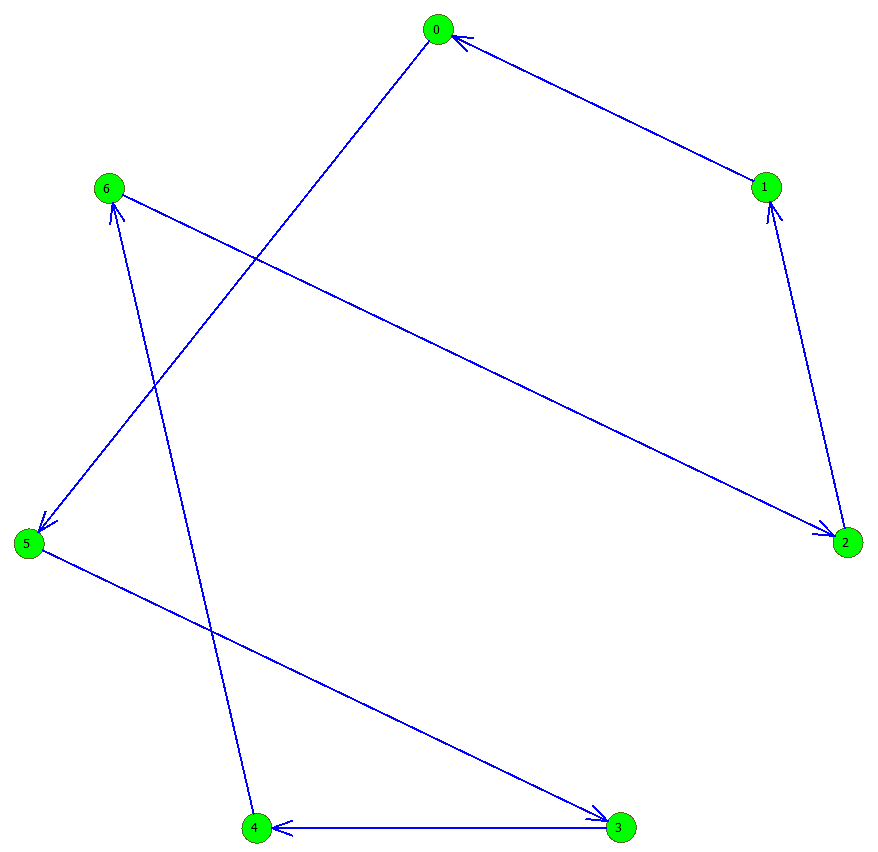


Рисунок 5. Граф, полученный после работы алгоритма из графа на рисунке 3.

**Выводы.**

В результате выполнения данной работы была изучена структура граф, ее свойства и операции над ней. Так же был реализован алгоритм для транзитивной редукции орграфа.

**Литература:**

1. Алгоритмы: построение и анализ. Кормен Т.Х., Лейзерсон Ч.И.

2. https://star-wiki.ru/wiki/Transitive\_reduction

3. https://ru.wikipedia.org/wiki/ Транзитивное\_сокращение