**Отчет**

Выполнил студент Шакирзянов Руслан 4214

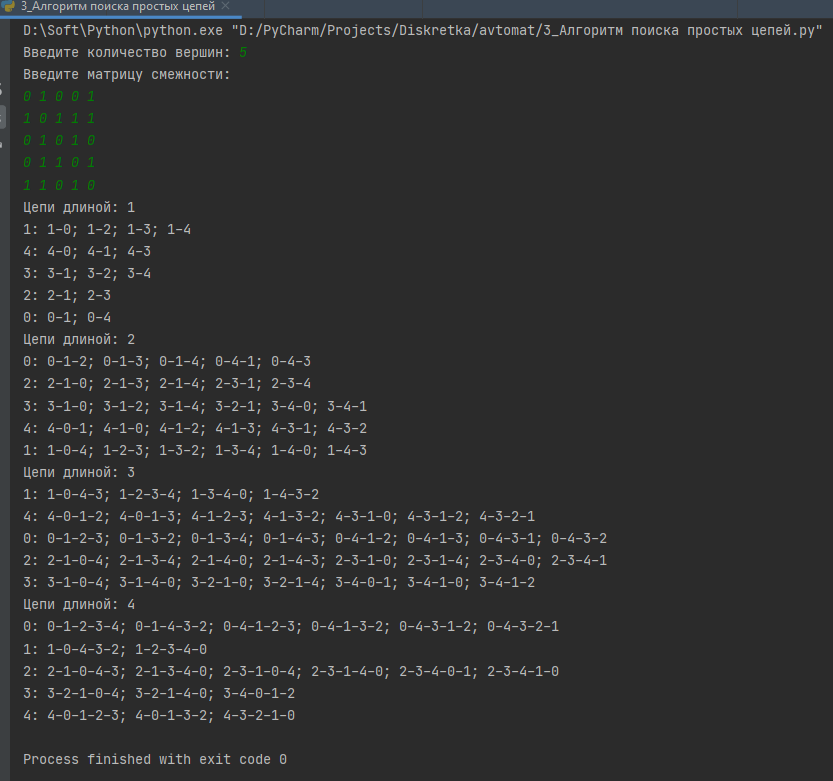
**Задача**

|  |  |  |
| --- | --- | --- |
| Алгоритм поиска простых цепей | Одна из форм представления графов (матрица смежности, матрица инциденций, списки смежности) – по выбору | Список цепей с разбиением по группам (по числу вершин) |

**Процедурное решение задачи**

Для нахождения всех простых цепей графа пройдем по всем вершинам и в цикле при помощи списков смежности будем рекурсивно вызывать функцию поиска с уменьшением длины цепи необходимой найти, когда длинна цепи будет равна 1 рекурсия закончится и вернет цепь длинной заданной изначально, в цикле также будем изменять длину цепи которую нужно найти, так мы найдем все простые цепи всех возможных длин

**Пример работы**



**Код программы**

def SmejToSmejList(vertex\_num, smejnost\_matrix):  
 smejnost\_lists = [[i] for i in range(vertex\_num)] # создание списка смежности с номерами вершин  
 for i in range(len(smejnost\_matrix)):  
 for j in range(len(smejnost\_matrix[i])): # проход по матрице смежности  
 if smejnost\_matrix[i][j] != 0: # если найдена смежная вершина  
 smejnost\_lists[i].append([j, smejnost\_matrix[i][j] if smejnost\_matrix[i][j] != 2 else 1]) # добавить в список смежности пару в формате [вершина, вес]  
 return smejnost\_lists  
  
  
def SimpleCep(vertex\_start, length, smejnost\_list, dict\_of\_chains, vertex\_num): # функция нахождения простых цепей  
 chain = [vertex\_start] # масив депи с изначально заданной ставтовой вершиной  
 chains = [] # массив всех цепей  
 if (vertex\_start, length) in dict\_of\_chains: # если кортеж из стартовой вершины и данной длинны уже есть в словаре цепей то вывести его  
 return dict\_of\_chains[(vertex\_start, length)]  
 if length == 1: # добавление цепей в список если длинна цепи равна 1 (фактически переписывание списка смежности для каждой вершины в удобном формате)  
 for i in range(vertex\_num):  
 if i in smejnost\_list[vertex\_start] and vertex\_start != i:  
 for j in range(smejnost\_list[vertex\_start].count(i)):  
 chain.append(i)  
 chains.append(chain)  
 chain = [vertex\_start]  
 dict\_of\_chains[(vertex\_start, length)] = chains # добавление цепей длинны 1 в словарь цепей  
 return chains  
 else:  
 for i in range(vertex\_num): # проход по всем вершинам  
 if i in smejnost\_list[vertex\_start] and vertex\_start != i: # проверка на то что статовая вершина не равна i для избежания лишних операций  
 for j in range(smejnost\_list[vertex\_start].count(i)):  
 for k in SimpleCep(i, length - 1, smejnost\_list, dict\_of\_chains, vertex\_num): # рекурсивный вызов функции с уменьшением длины цепей  
 if vertex\_start not in k:  
 chain = [vertex\_start] + k  
 chains.append(chain)  
 chain = []  
 dict\_of\_chains[(vertex\_start, length)] = chains # добавление цепей в словарь цепей  
 return chains  
  
  
def main():  
 vertex\_num = int(input("Введите количество вершин: ")) # запрос навведенение количества вершин  
 print("Введите матрицу смежности:")  
 smejnost\_matrix = [[int(i) for i in input().split()] for j in range(vertex\_num)] # ввод матрицы смежности с клавиатуры  
  
 smejnost\_list\_temp = SmejToSmejList(vertex\_num, smejnost\_matrix)  
 smejnost\_list = []  
 for i in range(vertex\_num):  
 temp\_list = []  
 for j in range(1, len(smejnost\_list\_temp[i])): # изменение структуры списка смежности из предыдущих работ  
 temp\_list.append(smejnost\_list\_temp[i][j][0])  
 smejnost\_list.append(temp\_list)  
  
 simple\_cep\_list = []  
 dict\_of\_chains = dict()  
 for i in range(vertex\_num):  
 for j in range(vertex\_num - 1, 1, -1):  
 simple\_cep\_list.append(SimpleCep(i, j, smejnost\_list, dict\_of\_chains, vertex\_num)) # заполнение списка простых цепей  
 length = 0  
 for i in list(sorted(dict\_of\_chains.keys(), key=lambda x: x[1])):  
 if i[1] > length:  
 length += 1  
 print(f"Цепи длиной: {length}") # красивый вывод всех простых цепей  
 print(f"{i[0]}: {'; '.join('-'.join(str(t) for t in j) for j in dict\_of\_chains[i])}")  
  
  
main()  
""" примеры  
5  
0 1 0 0 1  
1 0 1 1 1  
0 1 0 1 0  
0 1 1 0 1  
1 1 0 1 0  
"""