НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ФАКУЛЬТЕТ ІНФОРМАТИКИ І ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №4

з дисципліни **«**Комп’ютерне моделювання**»**

Виконав:

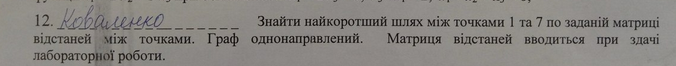
студент групи ІО-42, ФІОТ

Коваленко В’ячеслав Сергійович

ЗК : 4209

м. Київ 2016 р.

Завдання



Лістинг програми

from copy import deepcopy

vertex\_count = 7

init\_char = str(vertex\_count-1)

table = [[0, 2, 3, 5, 0, 0, 0],

[0, 0, 3, 7, 5, 0, 0],

[0, 0, 0, 5, 2, 2, 0],

[0, 0, 0, 0, 6, 0, 5],

[1, 0, 0, 0, 0, 7, 0],

[3, 0, 0, 4, 0, 0, 1],

[0, 0, 1, 0, 3, 0, 0]]

rev\_table = deepcopy(table)

possible\_ways = []

direct\_arr = [[], [], [], [], [], [], []]

transition\_costs = [0, 0, 0, 0, 0, 0, 0]

def clear\_transition\_costs():

for i in range(vertex\_count):

transition\_costs[i] = 0

def costs\_equal():

col = len(direct\_arr[0]) - 1

for i in range(vertex\_count):

if i != 0 and (sum(direct\_arr[i]) == 0):

direct\_arr[i].append(transition\_costs[i])

elif direct\_arr[i][col] > transition\_costs[i]:

direct\_arr[i].append(transition\_costs[i])

else:

direct\_arr[i].append(direct\_arr[i][col])

for i in range(vertex\_count):

if direct\_arr[i][col] != direct\_arr[i][col+1]:

return False

return True

def build\_direct\_arr():

for i in range(vertex\_count):

direct\_arr[i].append(table[0][i])

are\_different\_colls = True

steps = 1

while are\_different\_colls:

steps += 1

clear\_transition\_costs()

get\_sum(0, 0, steps)

if costs\_equal():

are\_different\_colls = False

def get\_next\_vertexes(vertex):

next\_vertex\_list = []

for i in range(vertex\_count):

if table[vertex][i] != 0:

next\_vertex\_list.append(i)

return next\_vertex\_list

def get\_sum(vertex, sum, steps):

next\_vertex\_list = get\_next\_vertexes(vertex)

if steps == 1:

for i in next\_vertex\_list:

if transition\_costs[i] == 0 or transition\_costs[i] > (sum + table[vertex][i]):

transition\_costs[i] = (sum + table[vertex][i])

else:

for i in next\_vertex\_list:

get\_sum(i, sum + table[vertex][i], steps - 1)

def cut\_last\_coll():

col\_number = len(direct\_arr[0])

if(col\_number > 1):

col\_number -= 1

for i in range(vertex\_count):

direct\_arr[i].pop(col\_number)

def reverse\_pass(weight, row, sequence):

if weight == 0:

possible\_ways.append(sequence)

return

possible\_previous\_vertexes = [i for i in range(vertex\_count-1) if rev\_table[row][i] > 0]

previous\_vertexes = [i for i in possible\_previous\_vertexes if (weight - rev\_table[row][i]) >= 0]

if len(previous\_vertexes) > 0:

if row == vertex\_count-1:

for i in previous\_vertexes:

seq = init\_char + str(i)

reverse\_pass(weight-rev\_table[row][i], i, seq)

else:

for i in previous\_vertexes:

temp = sequence[:]

temp += str(i)

reverse\_pass(weight-rev\_table[row][i], i, temp)

def reverse\_table():

for i in range(vertex\_count-1):

for j in range(i+1, vertex\_count):

rev\_table[i][j], rev\_table[j][i] = rev\_table[j][i], rev\_table[i][j]

def main():

print('Initial table')

for i in table:

print(i)

build\_direct\_arr()

print('Direct pass Matrix')

for i in direct\_arr:

print(i)

cut\_last\_coll()

reverse\_table()

reverse\_pass(direct\_arr[vertex\_count-1][len(direct\_arr[0])-1], vertex\_count-1, None)

if len(possible\_ways) >= 1:

result\_ways = [way[::-1] for way in possible\_ways if (way != None and (way[0] == str(vertex\_count-1)) and (way[len(way) - 1] == '0'))]

print('Result : ')

if len(result\_ways) > 0:

for result in result\_ways:

temp = ''

for c in result:

temp += c + ' -> '

print(temp[:-3])

else:

print('No possible ways')

if \_\_name\_\_ == '\_\_main\_\_':

main()

Результати програми

