**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ПЕНЗЕНСКИЙ ГОСУДРАСТВЕННЫЙ УНИВЕРСИТЕТ**

**ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ**

Факультет: «**Вычислительная техника**»

Кафедра: «**Математическое обеспечение и применение ЭВМ**»

Направление подготовки: **09.03.04 «Программная инженерия»**

**Алгоритмы на графах. Алгоритмы Дейкстры.  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

#### По дисциплине «Дискретная математика»

#### ОТЧЕТ

**По лабораторной работе №19**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |
| --- | --- |
| Выполнили: | Угроватов Д. Лялин Н. |
| Группа: | 16ВП1 |
| Проверил: | доц. Горюнов Ю.Ю. |

Пенза 2018

**Алгоритмы на графах. Алгоритмы Дейкстры.**

**Задание:**

1. Создать программу на языке Python, реализующую алгоритм Дейкстры, для поиска кротчайших путей из вершины «0» в вершину «5» графа. Матрица смежности:

**Ход работы:**

**Код файла labа18.py:**

def drawGr(n, M, R):

import pylab

from math import sin, cos, pi

pylab.figure(figsize = (5, 5))

def outL(x,y,k):

return pylab.text(x, y, str(k), size = 12)

for k in range(n):

for j in range(k,n):

xP = [R\*cos(2\*pi\*p/n) for p in [k, j]]

yP = [R\*sin(2\*pi\*q/n) for q in [k, j]]

pylab.plot(xP, yP, 'k.')

C=cos(2\*pi\*k/n); S=sin(2\*pi\*k/n)

if C>=0 and S>=0: outL(xP[0] + 0.15, yP[0], k) # I-я четверть,

if C<=0 and S>=0:outL(xP[0] - 0.75, yP[0], k) # II-я четверть,

if C<=0 and S<=0: outL(xP[0] - 0.75, yP[0] - 0.4, k) # III-я четверть,

if C>=0 and S<0:outL(xP[0] + 0.4, yP[0] - 0.1, k) # IV-я четверть,

# проводим ребра графа:

for k in range(n):

for j in range(k, n):

if M[k][j] >0:

xL = [R\*cos(2\*pi\*p/n) for p in [k, j]]

yL = [R\*sin(2\*pi\*q/n) for q in [k, j]]

pylab.plot(xL, yL, 'k-')

def algDijkstra(pBeg, n, M):

# Вход: pBeg - начальная вершина, n - порядок графа,

# M - матрица весов ребер.

# Выход: [d, From], где

# d – список кратчайших расстояний из начальной до всех остальных вершин,

# From – список, From[i] – номер вершины, из которой попали в i-ю вершину.

d = [-1 for i in range(n)] # расстояния от начальной вершины,

From = [-1 for i in range(n)] # из какой вершины пришли,

W = [0 for i in range(n)] # все вершины пометили, как не посещённые,

d[pBeg] = 0 # вес начальной вершины,

def Env(k, n, M): # Окружение вершины,

en = []

for i in range(n):

if M[k][i] >0 and W[i] ==0: en += [i]

return en

def popQue(Q): # удалить первый элемент из очереди

if Q == []:

return [-1, Q]

x = Q[0]; del Q[0]

return [x, Q]

Q = [pBeg] # начальную вершину в очередь,

while Q != []: # пока очередь непустая,

[x, Q] = popQue(Q) # вершина x была первой в очереди,

W[x] = 1 # пометили ее, как посещённую,

E = Env(x, n, M) # окружение x, в которых не были

if E != []:

for k in E:

if M[x][k]>0:

d0 = d[x] + M[x][k]

if d[k] == -1: d[k]=d0; From[k]=x

else:

if d0<d[k]: d[k] = d0

From[k] = x

for k in E: Q +=[k]

return [d,From]

n=6

pBeg=0

M = [[0,1,4,10,-1, 6],[1,0,2,-1,-1,2],[4,2,0,3,9,9],[10,-1,3,0,9,-1],[-1,-1,3,6,0,-1], [4,1,3,-1,-1,0]]

drawGr(n, M,1)

Res = algDijkstra(pBeg, n, M)

print('Кратчайшие расстояния из ',pBeg,'-й вершины до:')

for k in range(n):

print(k,'-й =',Res[0][k])

pEnd = 5

print('Кратчайший путь от ',pBeg,'-й до ',pEnd,'-й вершины:')

x = pEnd; path = [x]

From = Res[1]

while x != -1:

x=From[x]

if x != -1:

path = [x] + path

print(path)

**Скриншот результата выполнения программы**

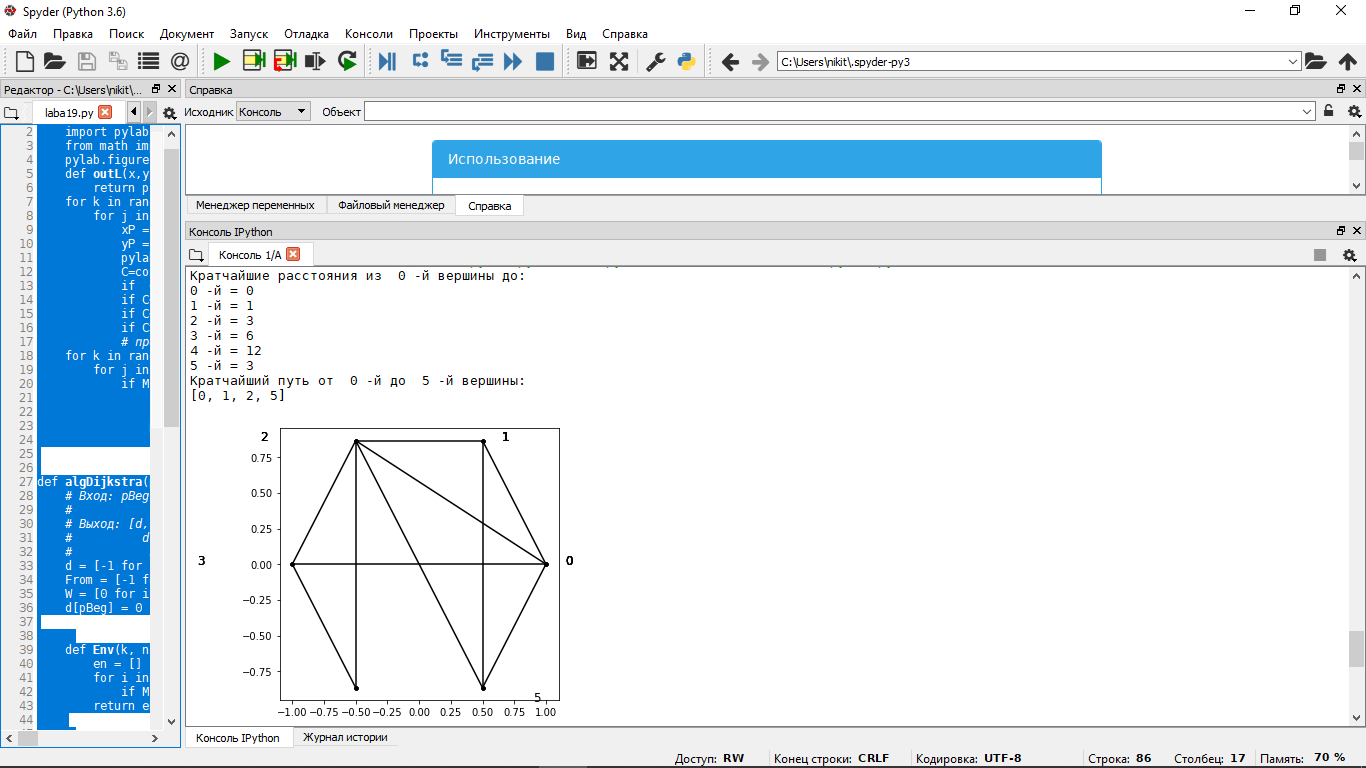


Рисунок 1 - Результат работы программы

**Вывод**

В результате выполнения лабораторной работы создали программу на языке Python, реализующую алгоритм Дейкстры, для поиска кротчайших путей из вершины «0» в вершину «5» графа с матрицей смежности: