Министерство науки и высшего образования

Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ**

**РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ имени В.Ф. УТКИНА»**

Кафедра «Космические технологии»

Отчет о выполнении лабораторных работ

по дисциплине

«Экспертные системы и ИИ»

Выполнил:

Студент группы 748

Понкратов А. С.

Проверил:

доц. каф. КТ

Наумов Д.А.

Рязань 2021

**ЦЕЛЬ РАБОТЫ:** ознакомление с проблемой кластерного анализа при интеллектуальной обработке данных в информационных системах; изучение алгоритмов кластеризации, использующих построение минимального остовного дерева; приобретение навыков в программной реализации изученных алгоритмов и в компьютерном проведении кластерного анализа.

**Краткое описание алгоритмов кластеризации**

Для построения минимального остовного дерева будет использоваться алгоритм Прима для решения задачи построения минимального маршрута между несколькими городами. Таблица с городами и расстояниями между ними приведена ниже.

Мinsk NY Msk Ber Kiev

Мinsk 0 -1 45 20 45

NY -1 0 40 35 15

Msk 45 40 0 35 10

Ber 20 35 35 0 70

Kiev 45 15 10 70 0

**Код программы**

matr = [

[0, -1, 45, 20, 45],

[-1, 0, 40, 35, 15],

[45, 40, 0, 35, 10],

[20, 35, 35, 0, 70],

[45, 15, 10, 70, 0]

]

def search\_min(tr, vizited):#1 место для оптимизации

min=max(max(tr))

index2 = 0

for ind in vizited:

for index, elem in enumerate(tr[ind]):

if elem>0 and elem<min and index not in vizited:

min=elem#веса путей

index2=index# индекс города

return [min, index2]

def prim(matr):

toVisit=[i for i in range(1,len(matr))]# города кроме начального(0)

vizited=[0]

result=[0]# начнем с минска

for index in toVisit:

weight, ind=search\_min(matr, vizited)

result.append(weight)#в результат будут заноситься веса

vizited.append(ind)# содержит карту пути

result[len(matr) - 1], result[len(matr) - 2] = result[len(matr) - 2], result[len(matr) - 1]

return result

print(prim(matr))

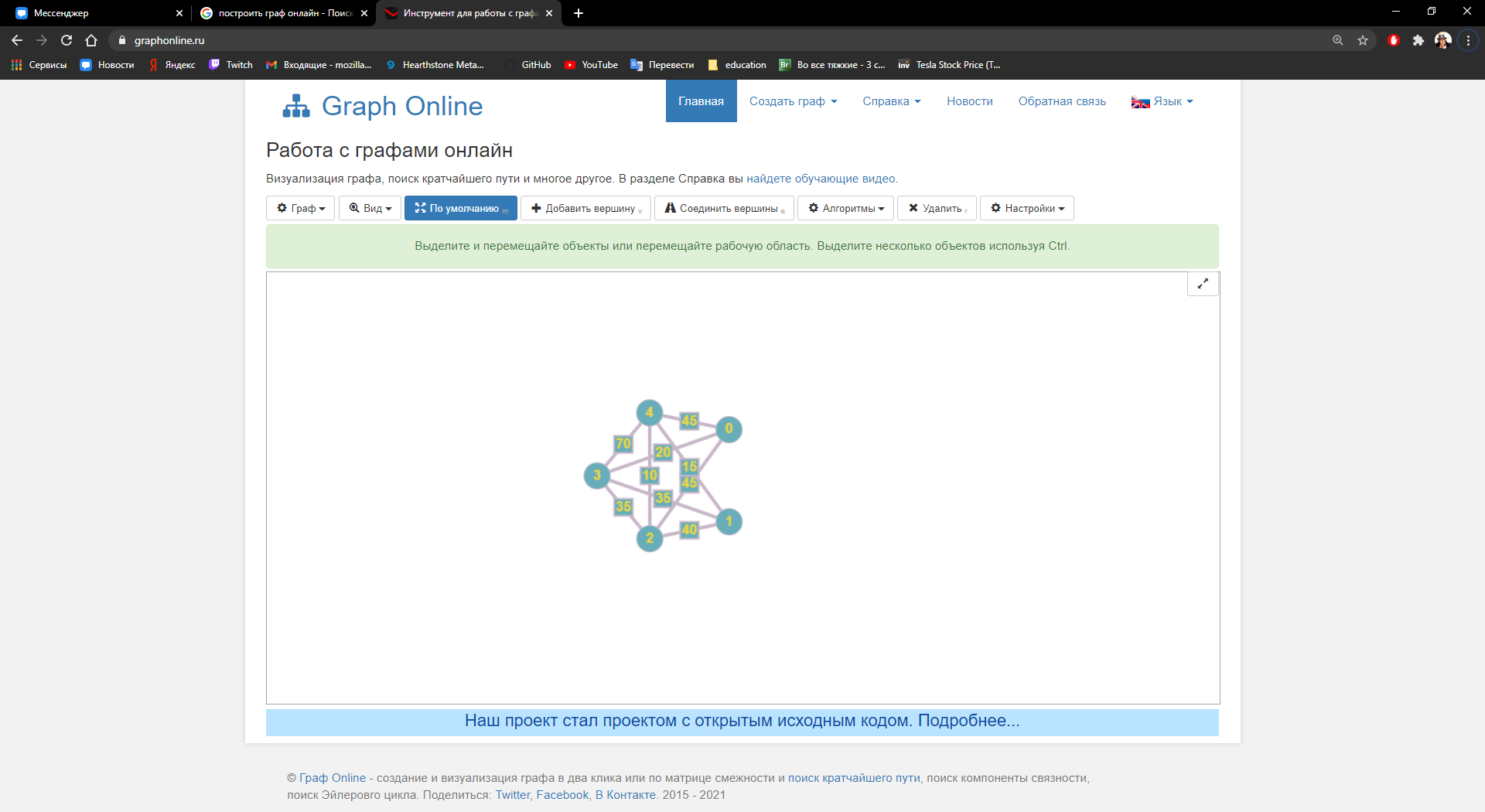


Рис. 1 – Взвешенный граф, составленный из вершин-городов, и весов – расстояний между городами

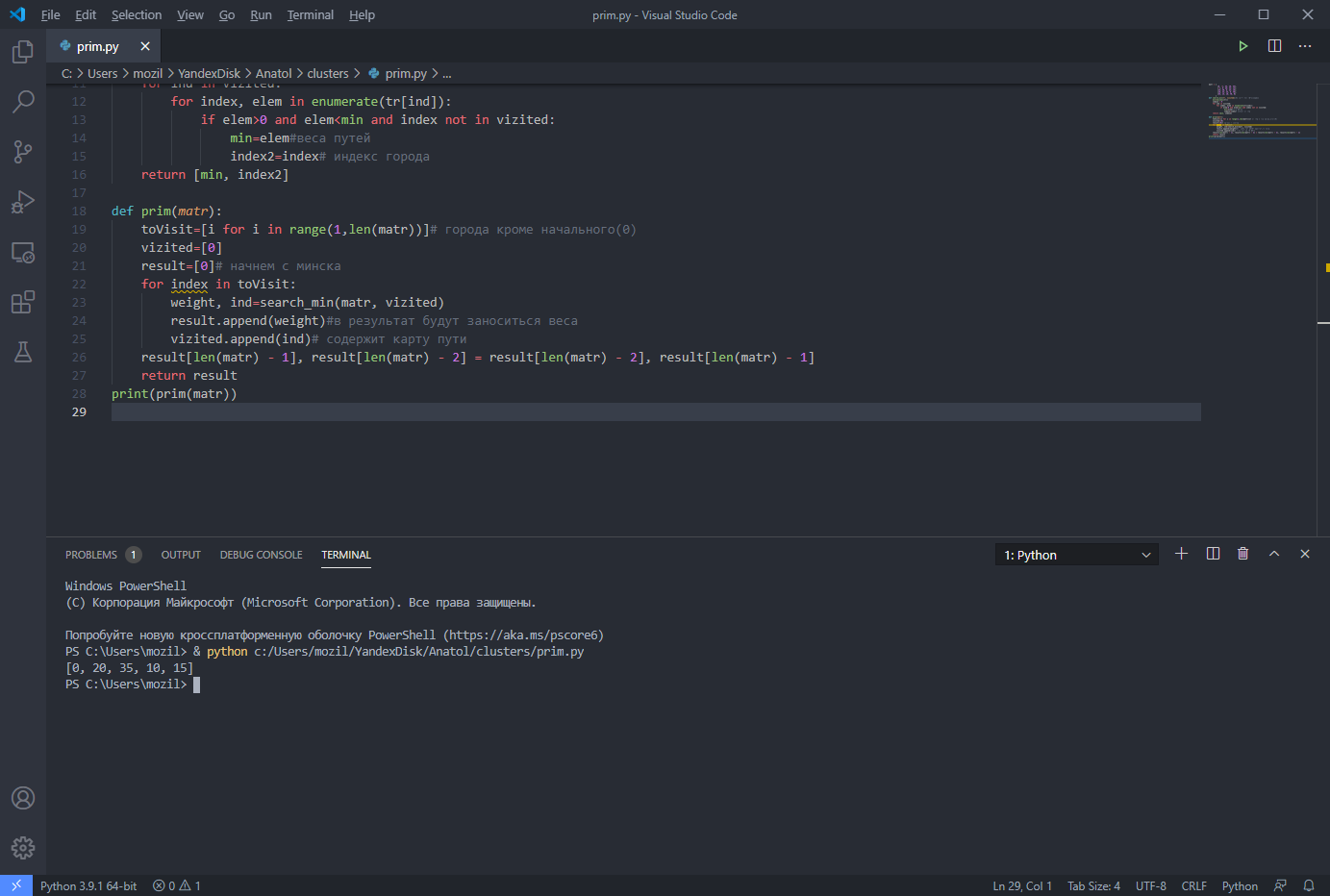


Рис. 2 – Результат работы алгоритма – кратчайший маршрут между городами

**Алгоритм k-средних**

Алгоритм k-средних разбивает множество элементов [векторного пространства](https://ru.wikipedia.org/wiki/%D0%9B%D0%B8%D0%BD%D0%B5%D0%B9%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D1%81%D1%82%D1%80%D0%B0%D0%BD%D1%81%D1%82%D0%B2%D0%BE) на заранее известное число кластеров *k*.

Начальными данными послужат координаты точек и количество кластеров.

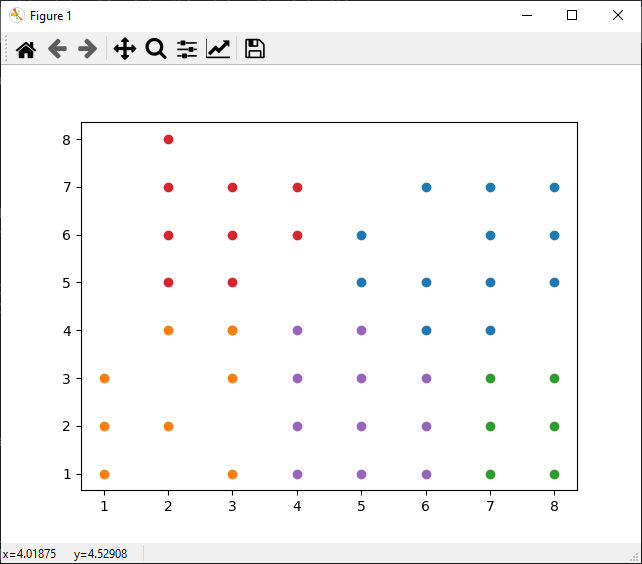


Рис. 3 – Работа алгоритма при k = 5.