Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»

Кафедра теоретических основ компьютерной безопасности и криптографии

**Отчет по практическому заданию №1**

**Создание ориентированного графа**

студента 5 курса 531 группы

специальности 10.05.01 «Компьютерная безопасность»

факультета компьютерных наук и информационных технологий

Минуситова Амиля Куанышкалиевича

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| Проверил  Доцент | \_\_\_\_\_\_\_\_\_\_ | И. И. Слеповичев |
|  | подпись, дата |  |

Саратов 2022

**Задание:**

Создать ориентированный граф, по заданным дугам.

**На входе:** текстовый файл с описанием графа в виде списка дуг:

(v1, u1, n1), (v2, u2, n2), ..., (vk, uk, nk)

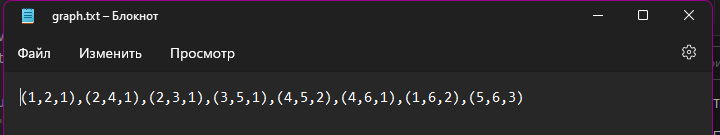
где vi- начальная вершина дуги i, ui- конечная вершина дуги i, ni- порядковый номер дуги в списке всех заходящих в вершину ui дуг.

**На выходе:**

а)Ориентированный граф с именованными вершинами и линейно упорядоченными дугами (в соответствии с порядком из текстового файла) записанный в файл формата XML.

б) Сообщение об ошибке в формате файла, если ошибка присутствует.

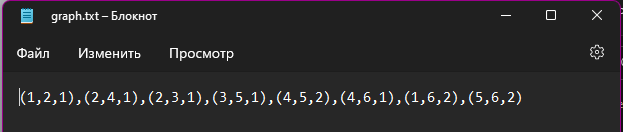
Входной файл

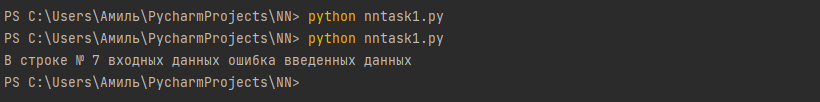


Выходной файл



Изменив в 7 дуге порядковый номер дуги входящей в вершину 6 получаем сообщение об ошибке.





Исходный текст программы:

import xml.etree.ElementTree as xml  
from re import split  
  
  
def write\_file(edges, v):  
 graph = xml.Element("graph")  
 for i in v:  
 vertex = xml.SubElement(graph, "vertex")  
 vertex.text = 'v' + str(i)  
 for i in edges:  
 arc = xml.SubElement(graph, "arc")  
 fromx = xml.SubElement(arc, "from")  
 fromx.text = 'v' + i[0]  
 to = xml.SubElement(arc, "to")  
 to.text = 'v' + i[2]  
 order = xml.SubElement(arc, "order")  
 order.text = i[4]  
 tree = xml.ElementTree(graph)  
 tree.write('graph1.xml')  
  
  
def read\_graph():  
 with open('graph.txt', 'r') as input\_graph:  
 j = 0  
 for line in input\_graph:  
 j += 1  
 edges = split('\),\(', line)  
 n = len(edges)  
 for i in range(n):  
 edges[i] = edges[i].replace('(', '')  
 edges[i] = edges[i].replace(')', '')  
  
 for i in range(n - 1):  
 a = edges[i]  
 # print(edge1)  
 for j in range(1, n):  
 b = edges[j]  
 # print(edge2)  
 if a != b:  
 if a[2] == b[2] and a[4] == b[4]:  
 print('В строке № {} входных данных ошибка введенных данных'.format(str(j)))  
 exit()  
 if a[0] == b[0] and a[2] == b[2]:  
 print('В строке № {} входных данных ошибка введенных данных'.format(str(j)))  
 exit()  
 edges.sort(key=lambda i: (i[0], i[2]))  
 c = edges  
 k = []  
 for i in range(n):  
 k.append(([int(edges[i][0]), int(edges[i][2]), int(edges[i][4])]))  
 edges = k  
 # print(edges)  
 v = []  
 for i in range(n):  
 v.append(edges[i][0])  
 v.append(edges[i][1])  
 v.sort()  
 vertex = []  
 for x in v:  
 if x not in vertex:  
 vertex.append(x)  
 return c, vertex  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 c, vertex = read\_graph()  
 write\_file(c, vertex)