Министерство науки и высшего образования Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем и программной инженерии

**Практическая работа № 4.1**

**«Мосты»**

**Выполнил**:

ст. гр. ПРИ-120

Д. А. Грачев

**Принял**:

Преподаватель

О. Н. Шамышева

Владимир, 2021

ЦЕЛЬ РАБОТЫ

Реализовать алгоритм поиска мостов в графе используя DFS на языке программирования Python.

ВЫПОЛНЕНИЕ РАБОТЫ

Листинг программы:

from PIL import Image, ImageDraw, ImageFont

from random import randint

videoDimensions = (1280, 1280)

img = Image.new("RGB", videoDimensions, color = '#FFE4C4')

imgDrawer = ImageDraw.Draw(img)

font = ImageFont.truetype('C:\Windows\Fonts\Arial.ttf', 30)

file = open('2 semestr/practice/Zadanie4/data.txt', 'r')

N = int(file.readline())

A = []

for i in range(N):

    A.append([])

while True:

    tmp = file.readline().rsplit()

    if tmp == []:

        break

    tmp[0], tmp[1] = int(tmp[0]) - 1, int(tmp[1]) - 1

    A[tmp[0]].append(tmp[1])

    A[tmp[1]].append(tmp[0])

used = [False]\*N

tin, fup = [0] \* N, [0] \* N

bridges = []

def dfs (v, timer, p = -1):

    used[v] = True

    tin[v] = timer

    fup[v] = timer

    for to in A[v]:

        if (to == p):

            continue

        if (used[to]):

            fup[v] = min (fup[v], tin[to])

        else:

            dfs (to, timer + 1, v)

            fup[v] = min (fup[v], fup[to])

            if (fup[to] > tin[v]):

                bridges.append([v, to])

def find():

    for i in range(N):

        if(not(used[i])):

            dfs(i, 0)

def new\_coord(X,Y):

    n = len(X)

    Y[0] = 10

    for i in range(n):

        X[i] = randint(50, videoDimensions[0] - 100)

        Y[i] = Y[i - 1] + videoDimensions[1] // n - 10

    return X, Y

def draw\_graph(x, y):

    for i in range(N):

        for j in A[i]:

            imgDrawer.line([(x[i], y[i]), (x[j], y[j])], 'black', 2)

    for i in range(N):

        imgDrawer.ellipse([(x[i] - 25, y[i] - 25), (x[i] + 25, y[i] + 25)], 'white', 'black', 2)

        imgDrawer.text((x[i] - 9, y[i] - 12), str(i), (0, 0, 0), ImageFont.truetype('C:\Windows\Fonts\Arial.ttf', 30))

def draw\_bridge(x, y):

    for i in bridges:

        imgDrawer.line([(x[i[0]], y[i[0]]), (x[i[1]], y[i[1]])], 'red', 4)

        imgDrawer.ellipse([(x[i[0]] - 25, y[i[0]] - 25), (x[i[0]] + 25, y[i[0]] + 25)], 'white', 'black', 2)

        imgDrawer.ellipse([(x[i[1]] - 25, y[i[1]] - 25), (x[i[1]] + 25, y[i[1]] + 25)], 'white', 'black', 2)

        imgDrawer.text((x[i[0]] - 9, y[i[0]] - 12), str(i[0]), (0, 0, 0), ImageFont.truetype('C:\Windows\Fonts\Arial.ttf', 30))

        imgDrawer.text((x[i[1]] - 9, y[i[1]] - 12), str(i[1]), (0, 0, 0), ImageFont.truetype('C:\Windows\Fonts\Arial.ttf', 30))

X = [0]\*N

Y = [0]\*N

new\_coord(X, Y)

draw\_graph(X, Y)

find()

draw\_bridge(X, Y)

imgDrawer.text((100, 1200), 'Путь: ', (0,0,0), font)

imgDrawer.text((200, 1200), str(bridges), (0,0,0), font)

img.show()

Скриншот работы программы и данные из файла data.txt представлены на Рис.1.1 и 1.2.

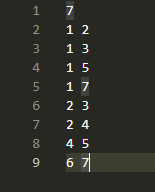
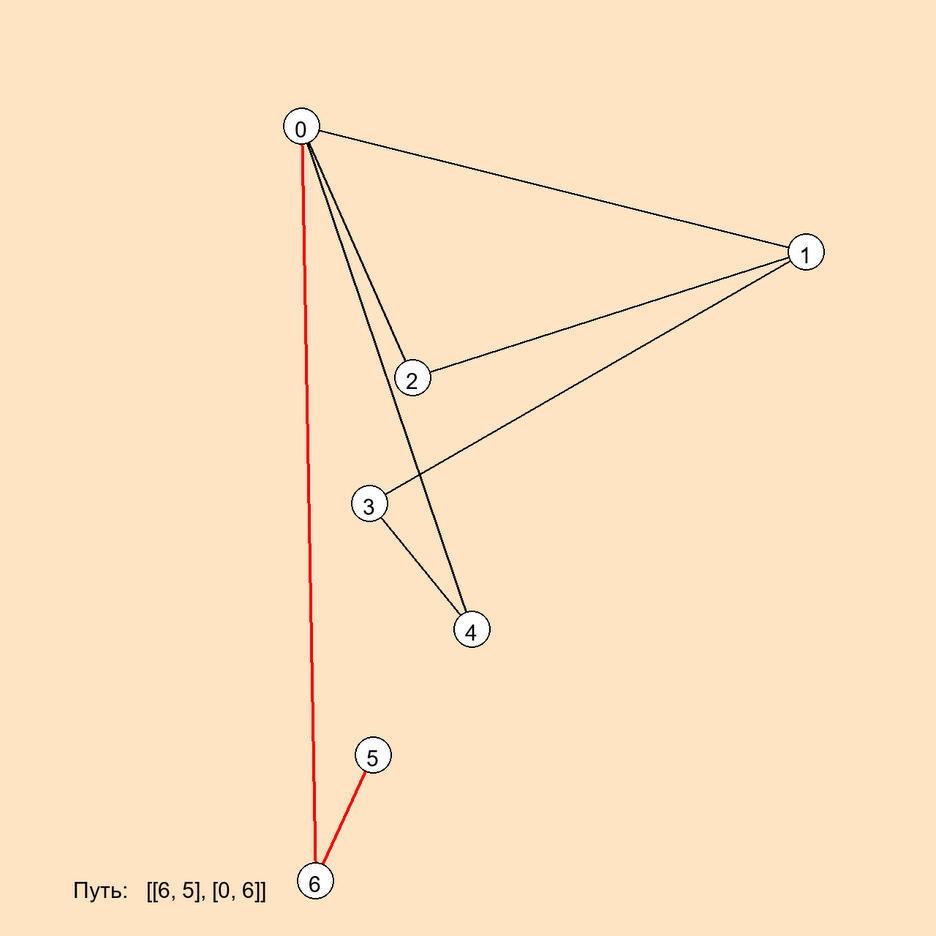


Рисунок 1.1 и 1.2. Результат работы программы и data.txt

ВЫВОД

В ходе выполнения работы были получены практические навыки по использованию алгоритма DFS для поиска мостов в графе.