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

**Лабораторная работа № 1**

**по дисциплине**

**«Алгоритмы и структуры данных»**

**РЕКУРСИВНАЯ РЕАЛИЗАЦИЯ АЛГОРИТМОВ**

**Выполнил**:

ст. гр. ПРИ-120

Д. А. Грачев

**Принял**:

Палик П.Г.

Владимир, 2021

ЦЕЛЬ РАБОТЫ

Научиться реализовывать рекурсивные алгоритмы на языке Python.

ВЫПОЛНЕНИЕ РАБОТЫ

**Задание №1**

На языке Python реализовать алгоритм построения фрактала.

Листинг программы:

from tkinter import \*

def fractal(A,B,C,D,deep=10,alpha=0.1):

    '''

    Построение фрактала квадрата по 4 точкам\n

    deep – глубина\n

    alpha – коэффициент погружения(>1 – расширяется, <1 – уменьшается)

    '''

    if deep < 1:

        return

    for M, N in (A,B),(B,C),(C,D),(D,A):

        canv.create\_line([\*M],[\*N])

    A1 = (A[0]\*(1-alpha)+B[0]\*alpha, A[1]\*(1-alpha)+B[1]\*alpha)

    B1 = (B[0]\*(1-alpha)+C[0]\*alpha, B[1]\*(1-alpha)+C[1]\*alpha)

    C1 = (C[0]\*(1-alpha)+D[0]\*alpha, C[1]\*(1-alpha)+D[1]\*alpha)

    D1 = (D[0]\*(1-alpha)+A[0]\*alpha, D[1]\*(1-alpha)+A[1]\*alpha)

    fractal(A1,B1,C1,D1,deep-1,alpha)

root = Tk()

canv = Canvas(root, width=700, height=700, bg=»white»)

fractal((5,5),(695,5),(695,695),(5,695),deep=994,alpha=0.01)

canv.pack()

root.mainloop()

Скриншот работы программы представлен на Рис. 1.

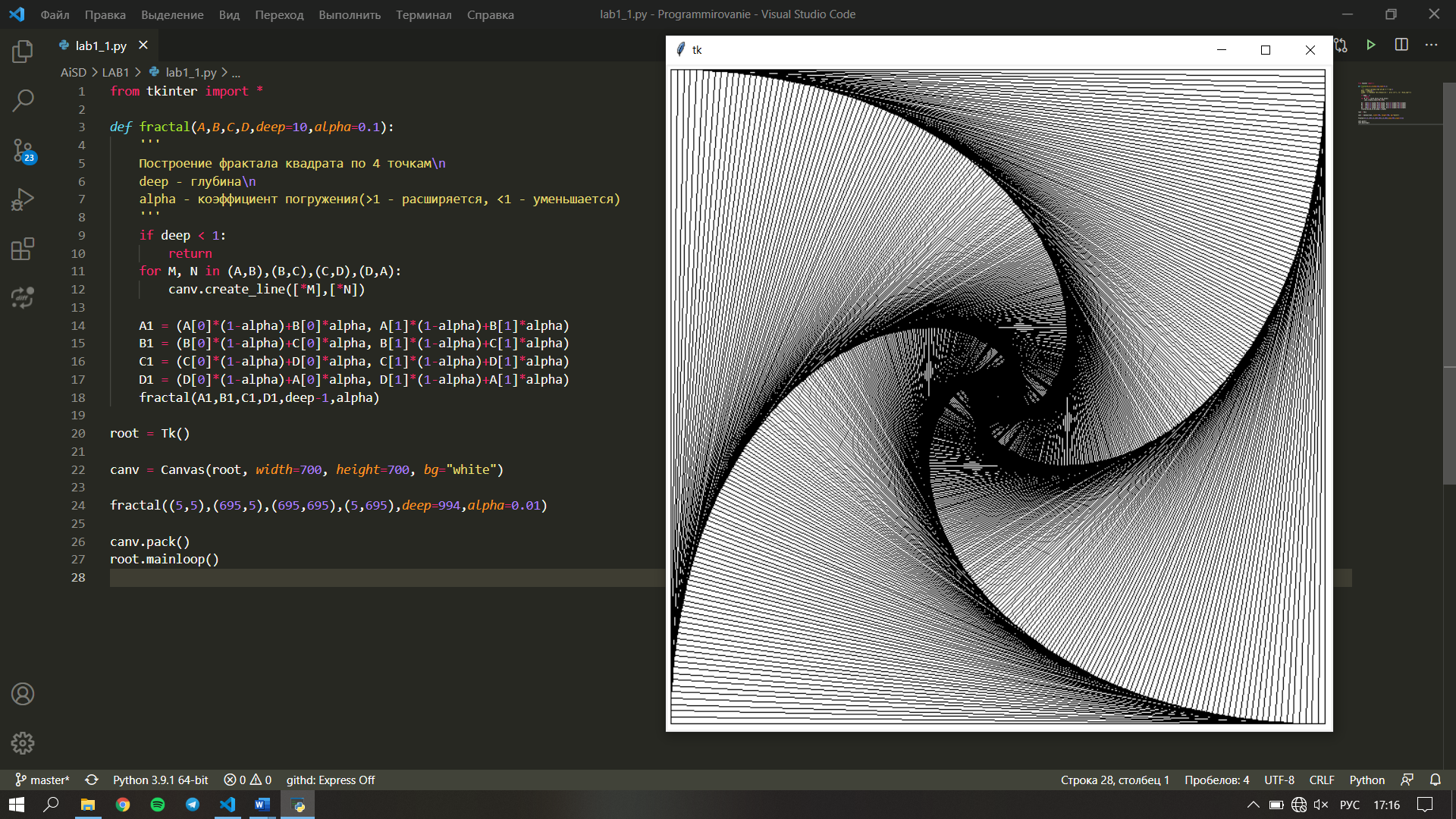


Рисунок 1. Скриншот работы программы

**Задание №2**

На языке Python составить алгоритм:

Пусть функция f(x) вещественной переменной x непрерывна на отрезке [a,b] и f(a)\*f(b) < 0. Составить рекурсивную программу нахождения на [a,b] какого либо вещественного корня функции f(x).

Листинг программы:

def f(x):

    return x\*\*2-4

def c(a,b,e=0.001):

    if abs(b-a) <= 2\*e:

        return (a+b)/2

    else:

        m = (a+b)/2

        if f(a)\*f(m) <= 0:

            b = m

        else:

            a = m

        return c(a,b,e)

a,b = map(int,input().split())

if f(a)\*f(b) < 0:

    print(c(a,b))

else:

    print («Не подходит по условиям!»)

Скриншот работы программы представлен на Рис. 2.

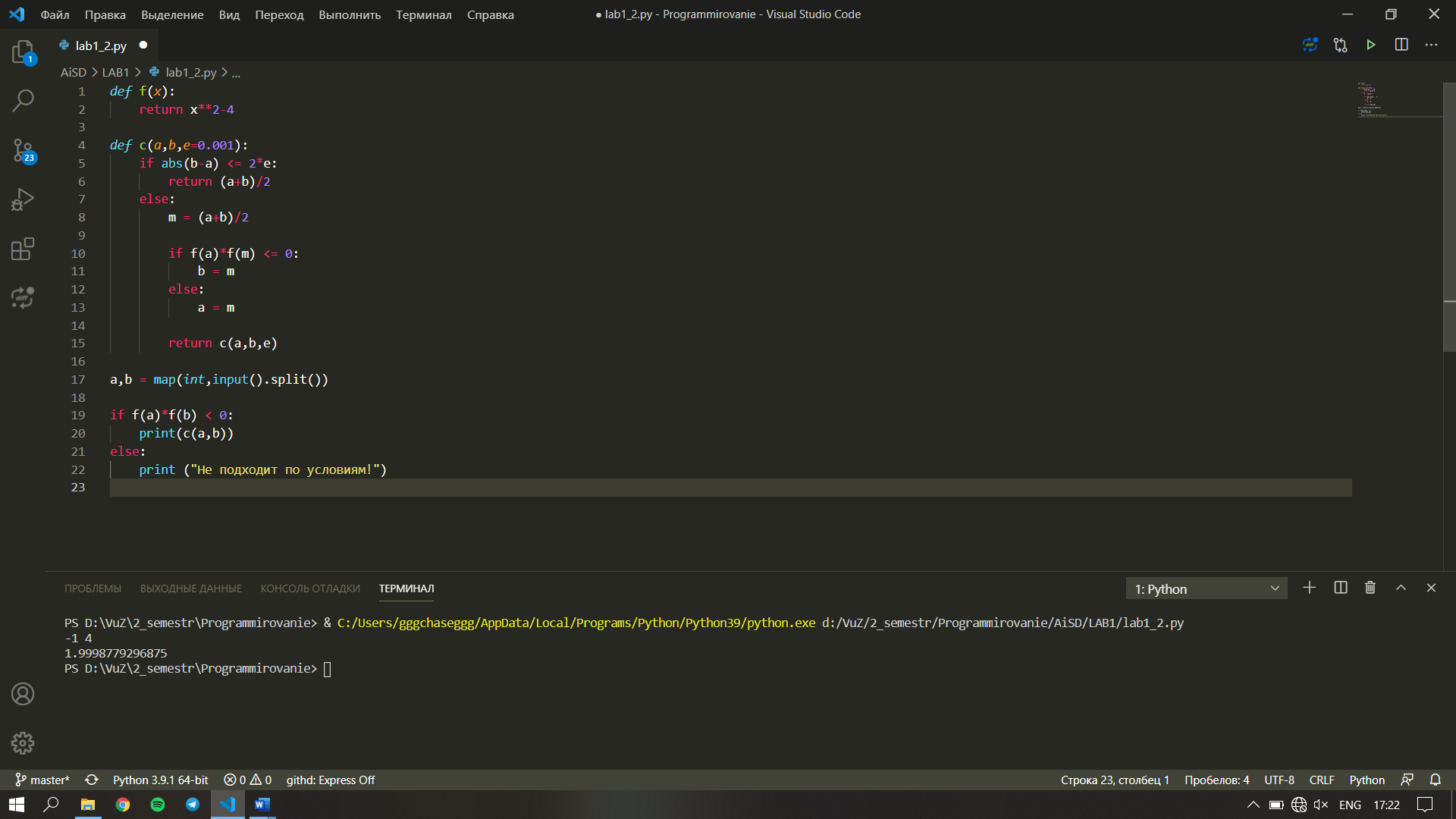


Рисунок 2. Скриншот работы программы

**Задание №3**

На языке Python реализовать алгоритм:

Задан словарь и слов одинаковой длины, например «барак, баран, банан, дурак, бутон, бутан, питон, барин». Для заданной пары слов определить цепочку так, чтобы каждое следующее слово отличалось не более чем на 2 буквы.

Листинг программы:

import random

def sravnenie(a,b):

    k = 0

    for I in range(len(a)):

        if not (a[i] in b):

            k += 1

            if k > 2:

                break

    k1 = 0

    for I in range(len(b)):

        if not (b[i] in a):

            k1 += 1

            if k > 2:

                break

    if k > 2 or k1 > 2:

        return False

    return True

def finder\_par():

    if len(slovar) == 1:

        return slovar[0]

    if len(cepochka) == 0:

        ind = random.randint(0,(len(slovar))-1)

        cepochka.append(slovar[ind])

        slovar.pop(ind)

    for I in range(len(slovar)):

        if sravnenie(cepochka[len(cepochka)-1],slovar[i]):

            cepochka.append(slovar[i])

cepochka = []

slovar = [‘барак’, ‘баран’, ‘банан’, ‘дурак’, ‘бутон’, ‘бутан’, ‘питон’, ‘барин’]

finder\_par()

print(cepochka)

Скриншот работы программы представлен на Рис. 3.

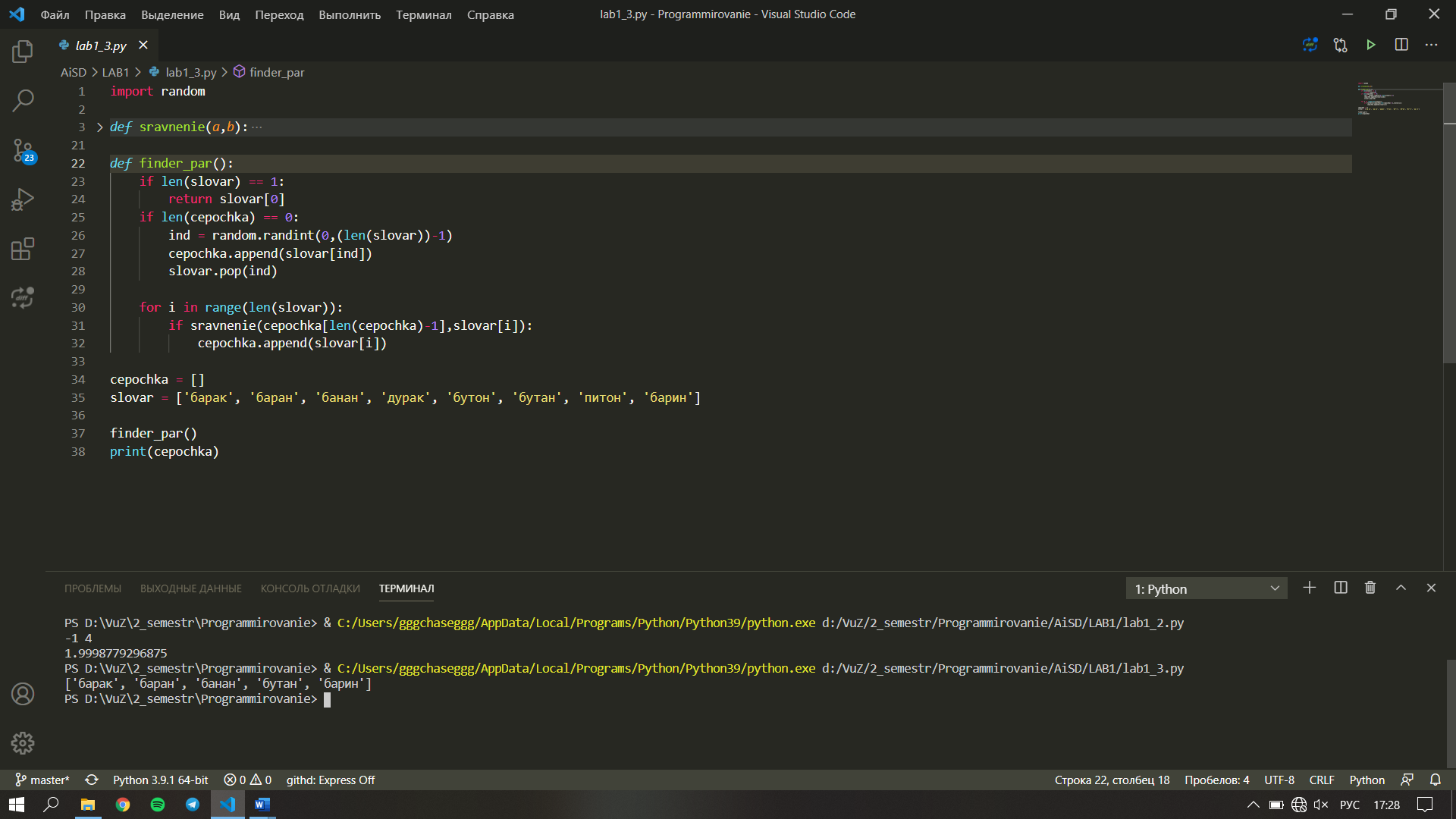


Рисунок 3. Скриншот работы программы

ВЫВОД

В ходе выполнения работы были получены практические навыки по реализации рекурсивных алгоритмов на языке Python.