**Лабораторная работа №2**

Приближенное вычисление корней системы нелинейных уравнений методом Ньютона

**Цель работы:** освоение методов приближенного вычисления корней системы нелинейных уравнений **f**(**x**) = **0**. Подробное изучение метода Ньютона. Получение навыков решения задач вычислительной математики на компьютере.

**Задание:** Написать программу вычисления приближенного решения системы нелинейных уравнений **f**(**x**) = **0** с точностью *ε = 10-5* методом Ньютона для случаев:

* частные производные находятся аналитически,
* частные производные находятся численно.

# Вариант 2



Код программы на python

import math

e = 0.00001

h = 0.001

def F(x, y):

return math.sin(x-2.2\*y) - y \* x + 2.0

def a(x, y):

return math.cos(x-2.2\*y)-y

def b(x, y):

return -2.2\*math.cos(x-2.2\*y)-x

def a2(x, y): #FDerivativeBy\_X

return (F(x+h, y)-F(x, y))/h

def b2(x, y):

return (F(x, y+h)-F(x, y))/h

def G(x, y):

return x\*\*2/1.75-y\*\*2-0.75

def c(x, y):

return x\*2.0/1.75

def d(x, y):

return -2.0\*y

def c2(x, y):

return (G(x+h, y)-G(x, y))/h

def d2(x, y):

return (G(x, y+h)-G(x, y))/h

**def NewtonMethod():**

n = 0

x = 1.7

y = 1.0

xn = x - (d(x, y)\*F(x, y)-b(x, y)\*G(x, y))/(a(x, y)\*d(x, y)-b(x, y)\*c(x, y))

yn = y - (a(x, y)\*G(x, y)-c(x, y)\*F(x, y))/(a(x, y)\*d(x, y)-b(x, y)\*c(x, y))

while math.sqrt((xn-x)\*\*2+(yn-y)\*\*2)>= e or math.sqrt(F(xn, yn)\*\*2+G(xn, yn)\*\*2)>= e:

n+=1

x = xn

y = yn

xn = x - (d(x, y)\*F(x, y)-b(x, y)\*G(x, y))/(a(x, y)\*d(x, y)-b(x, y)\*c(x, y))

yn = y - (a(x, y)\*G(x, y)-c(x, y)\*F(x, y))/(a(x, y)\*d(x, y)-b(x, y)\*c(x, y))

print('Number of iteration n = ', n)

print('F(xn+1, yn+1) = ', F(xn, yn))

print('G(xn+1, yn+1) = ', G(xn, yn))

array = [xn, yn]

return array

def NewtonMethod2():

n = 0

x = 1.7

y = 1.0

xn = x - (d2(x, y)\*F(x, y)-b2(x, y)\*G(x, y))/(a2(x, y)\*d2(x, y)-b2(x, y)\*c2(x, y))

yn = y - (a2(x, y)\*G(x, y)-c2(x, y)\*F(x, y))/(a2(x, y)\*d2(x, y)-b2(x, y)\*c2(x, y))

while math.sqrt((xn-x)\*\*2+(yn-y)\*\*2)>= e or math.sqrt(F(xn, yn)\*\*2+G(xn, yn)\*\*2)>= e:

n+=1

x = xn

y = yn

xn = x - (d2(x, y)\*F(x, y)-b2(x, y)\*G(x, y))/(a2(x, y)\*d2(x, y)-b2(x, y)\*c2(x, y))

yn = y - (a2(x, y)\*G(x, y)-c2(x, y)\*F(x, y))/(a2(x, y)\*d2(x, y)-b2(x, y)\*c2(x, y))

print('Number of iteration n = ', n)

print('F(xn+1, yn+1) = ', F(xn, yn))

print('G(xn+1, yn+1) = ', G(xn, yn))

array = [xn, yn]

return array

#вывод результата

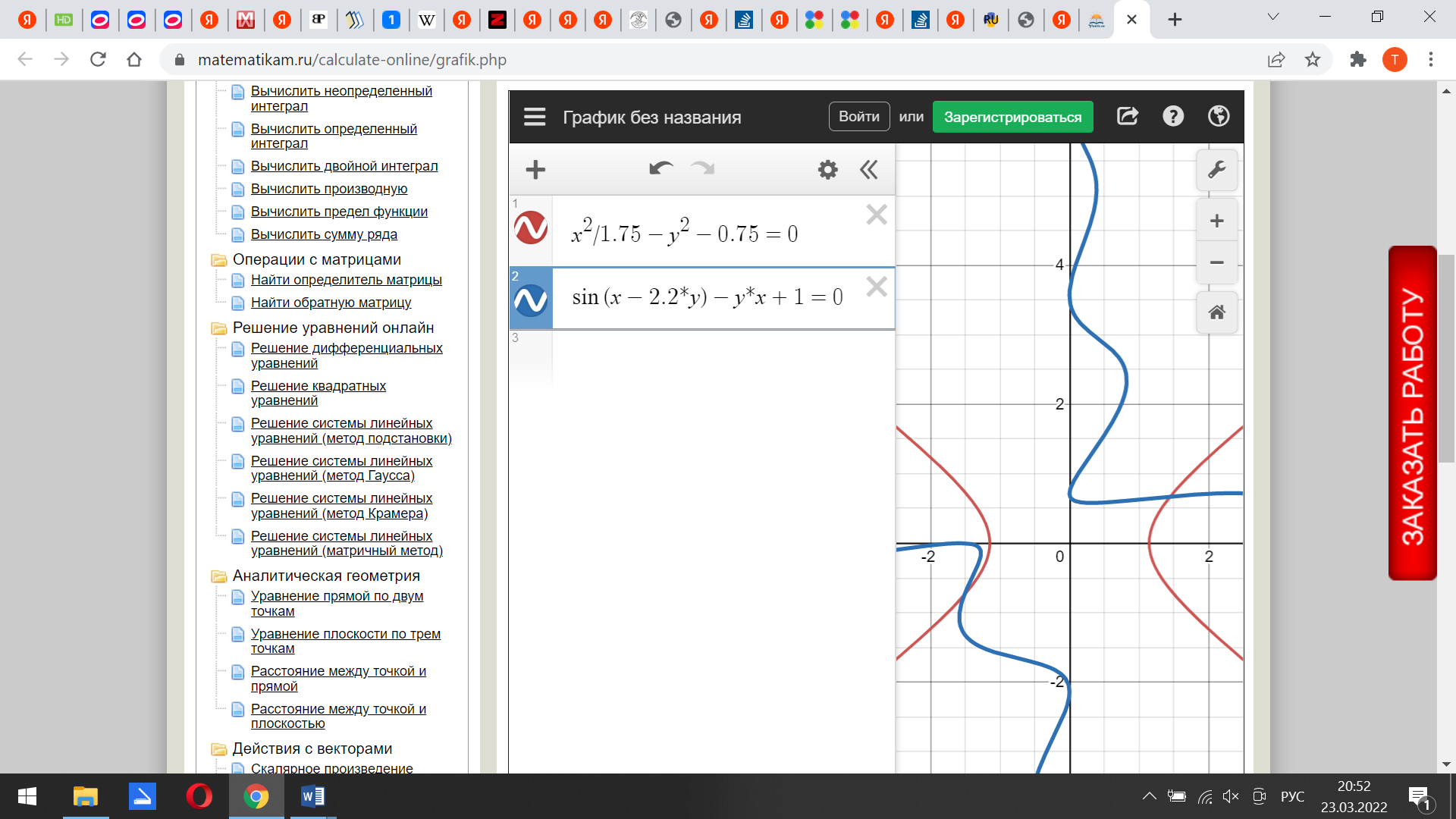
array = NewtonMethod()

print ('root using Newton\_method x0=1.7, y0=1.0: x=%.6f, y=%.6f' % (array[0], array[1]))

array = NewtonMethod2()

print ('root using Newton\_method2 x0=1.7, y0=1.0: x=%.6f, y=%.6f' % (array[0], array[1]))

График



Результат

