**Лабораторная работа № 1**

Приближенное вычисление корней уравнения f(x) = 0 методом дихотомии и методом простой итерации

Задание: Написать программу приближенного вычисления двух ближайших к 0 корней уравнения f(x) = 0 с точностью e = 10-5 методом дихотомии и методом простой итерации с выбором оптимального итерационного параметра.

# Вариант 2

Код программы на python

#вычисления

import math

e = 0.00001

**def BisectionMethod(a, b):**

n = 0

x = (a + b) / 2

while math.fabs(4\*x - 8\*math.sin(x) + 1) >= e:

x = (a + b) / 2

n+=1

a, b = (a, x) if (4\*a - 8\*math.sin(a) + 1) \* (4\*x - 8\*math.sin(x) + 1) < 0 else (x, b)

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

return x

**def SimpleIterationMethod(a, b, t):**

n = 0

x = (a + b) / 2

while math.fabs(4\*x - 8\*math.sin(x) + 1) >= e:

x = x + t\*(4\*x - 8\*math.sin(x) + 1)

n+=1

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

return x

#построение графика

import pylab

import numpy

X = numpy.arange(-4, 4, 0.1)

pylab.plot([x for x in X], [(4\*x - 8\*math.sin(x) + 1) for x in X])

pylab.grid(True)

#вывод результата

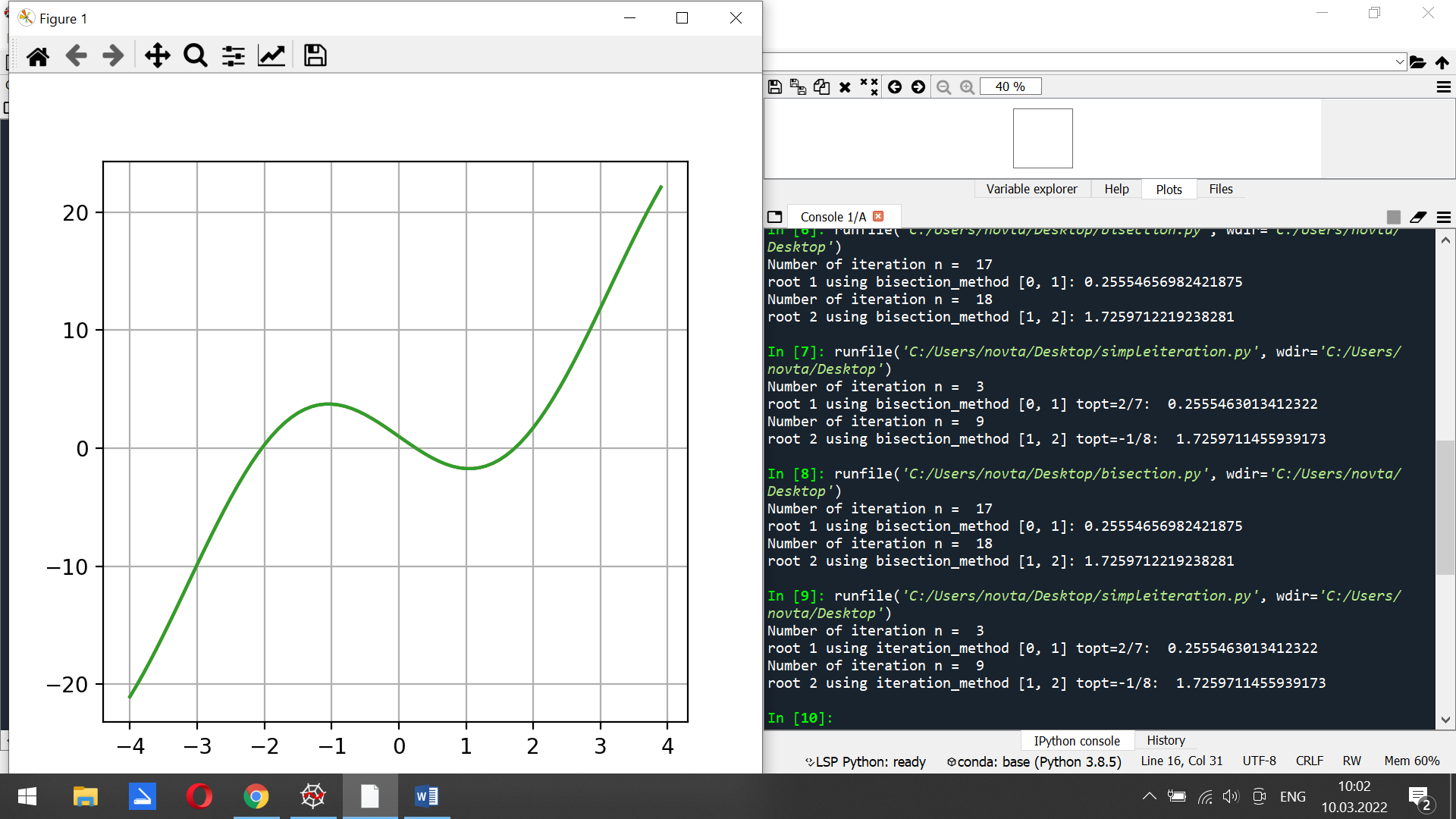
print ('root 1 using bisection\_method [0, 1]:%.6f' % BisectionMethod(0, 1))

print ('root 2 using bisection\_method [1, 2]:%.6f' % BisectionMethod(1, 2))

print ('root 1 using iteration\_method [0, 1] topt=2/7: %.6f' % SimpleIterationMethod(0, 0.5, 2/7))

print ('root 2 using iteration\_method [1, 2] topt=-1/8: %.6f' % SimpleIterationMethod(1.5, 2, -0.125))

График



Результат

