**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информатика»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

по дисциплине «Методы численного анализа»

на тему: «Решение алгебраических уравнений и систем»

Выполнила: студент гр. ИП-22

Коваленко А.И.

Принял: преподаватель

Процкая М.А,

Гомель 2022

**Цель:** Получить навыки решения уравнений различных видов с использованием численных методов, научиться решать прикладные задачи с решением систем уравнений.

***Задание 1***

Для уравнения по индивидуальному заданию:

-- на заданном глобальном интервале изоляции найти несколько локальных интервалов изоляции;

-- для каждого локального интервала изоляции найти корень уравнения с применением метода половинного деления.

*Таблица 1*

|  |  |  |
| --- | --- | --- |
| 7. | sin2(x-0,65) – 0,2x=0 | a=0 b=6 |

**Листинг задания 1**

import numpy as np

import matplotlib.pyplot as plt

def f(x):

return (np.sin(x-0.65))\*\*2-0.2\*x

a = 0

b = 6

x = a

eps = 0.001

DotA = [0]

DotB = [0]

step = 1

while x <= b - step:

xNext = x + step

if f(x) \* f(xNext) < 0:

DotA += [x]

DotB += [xNext]

x = xNext

list\_dop = []

count = 0

for i in range(1, len(DotA)):

print('Корни на интервале %6.3f ;' %(DotA[i]), '%6.3f' %(DotB[i]))

a = DotA[i]

b = DotB[i]

while b - a > eps:

t = (a + b) / 2

if f(t) \* f(a) < 0:

b = t

else:

a = t

list\_dop.append(t)

print('%8.2f' % list\_dop[count])

count += 1

x = np.arange(0, 6, 0.0001)

plt.plot(x, f(x))

for i in range(0, len(list\_dop)):

plt.plot(list\_dop[i], f(list\_dop[i]), 'ro')

plt.grid()

plt.show()

**Результат выполнения задания 1**

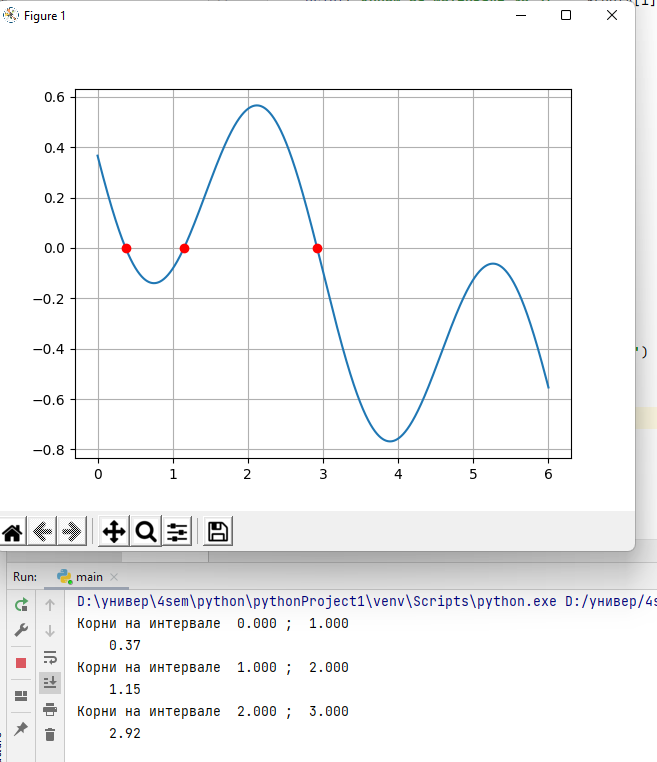
******

Рисунок 1 – Результат выполнения задания 1

***Задание 2***

Решить уравнение методом хорд и методом касательных, сравнить полученные результаты.

Таблица 2

|  |  |
| --- | --- |
| 7. | a= 3 b= 5 |

**Листинг задания 2**

#метод хорд

from math import\*

import numpy as np

import matplotlib.pyplot as plt

a = 3

b = 5

num = [0]

def f(x):

return 0.5\*x\*\*4 - 1.1\*x\*\*3 - 7\*x\*\*2 + 13.7\*x - 6.6

def findRoot(xn, xk):

a = xn

b = xk

eps = 0.00001

while abs(b - a) > eps:

a = b - (b - a) \* f(b) / (f(b) - f(a))

b = a + (a + b) \* f(a) / (f(a) - f(b))

return b

for i in range(1):

print(findRoot(a, b))

num += [findRoot(a, b)]

x = np.arange(3.25, 5.2, 0.0001)

plt.plot(x, f(x), 'y')

for i in range(1, len(num)):

plt.plot(num[i], f(num[i]), 'ro')

plt.grid()

plt.show()

#метод касательных

import math

import numpy as np

import matplotlib.pyplot as plt

from scipy.misc import derivative

def F(x):

return 0.5 \* x \*\* 4 - 1.1 \* x \*\* 3 - 7 \* x \*\* 2 + 13.7 \* x - 6.6

def F1(x):

return derivative(F, x, dx=1e-6)

def Method(a, b):

x0 = (a + b) / 2

xn = F(x0)

xn1 = xn - F(xn) / F1(xn)

while abs(xn1 - xn) > math.pow(10, -5):

xn = xn1

xn1 = xn - F(xn) / F1(xn)

return xn1

a = 0

b = 5

dx = 1

x1 = a

la = [0]

lb = [0]

while x1 <= b - dx:

x2 = x1 + dx

if F(x1) \* F(x2) < 0:

la += [x1]

lb += [x2]

x1 = x2

for i in range(1, len(la)):

print('%8.2f' % la[i], '%8.2f' % lb[i])

print(Method(a, b))

num = (Method(a, b))

**Результат выполнения задания 2**

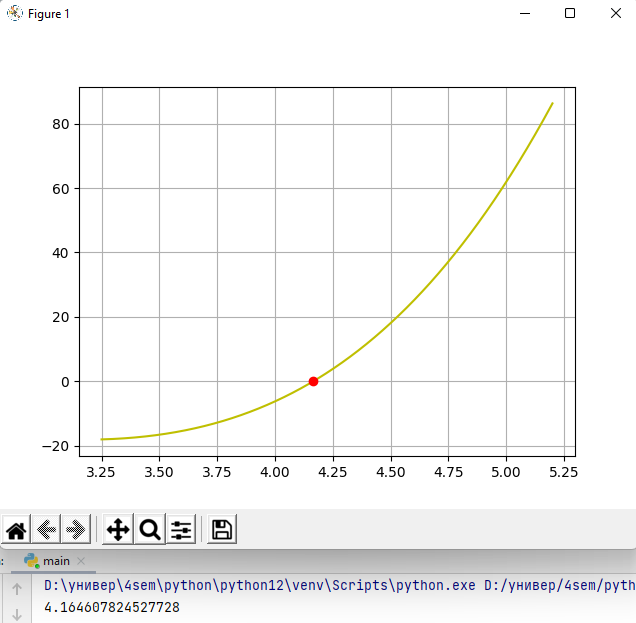
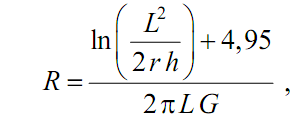


Рисунок 2 – Результат выполнения задания 2

**Задание 3 (Индивидуальное задание 2.2)**

Заземлитель, изготовленный в виде решетки прямоугольной формы из металлических труб, расположен горизонтально в грунте на глубине ***h***. Сопротивление заземлителя рассчитывается по формуле



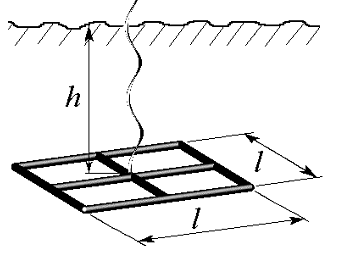
где π = 3,14…,

***L*** = 6×***l*** − суммарная длина труб,

***r***  − радиус труб,

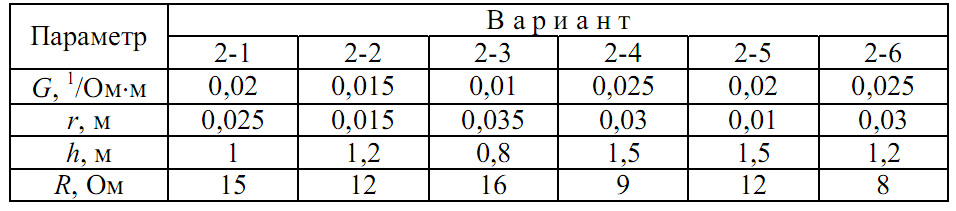
***h*** − глубина,

***G*** − удельная электропроводность грунта.



***Задание***

1. Задавшись параметрами ***r*** , ***h*** (из таблицы), определите размер ***l***, соответствующий требуемому сопротивлению ***R.*** Доказать графически, что значение ***l*** найдено верно.



**Листинг задания 3**

**import** numpy **as** np  
**import** matplotlib.pyplot **as** plt  
**import** math  
  
a = 1.5  
b = 2  
e = 0.00001  
G = 0.015  
r = 0.015  
h = 1.2  
R = 12  
  
  
**def** f(l):  
 **return** (np.log(((6 \* l) \*\* 2) / (2 \* r \* h)) + 4.95) / (math.pi \* 2 \* 6 \* l \* G) - R  
  
  
y1 = f(a)  
y2 = f(b)  
  
**if** y1 \* y2 >= 0:  
 print(**'Корней нет'**)  
 c = 1  
**else**:  
 n = 1  
 c = (a + b) / 2  
 y3 = f(c)  
 **while** (abs(y3) > e):  
 c = (a + b) / 2  
 y3 = f(c)  
 **if** y1 \* y3 < 0:  
 b = c  
 **else**:  
 a = c  
 n += 1  
  
x = np.arange(1.7, 2.5, 0.01)  
plt.plot(x, f(x), **'y'**)  
plt.plot(c, f(c), **'ro'**)  
print(**'c = %0.2f '** % c)  
print(c)  
plt.grid()  
plt.show()

**Результат выполнения задания 3**

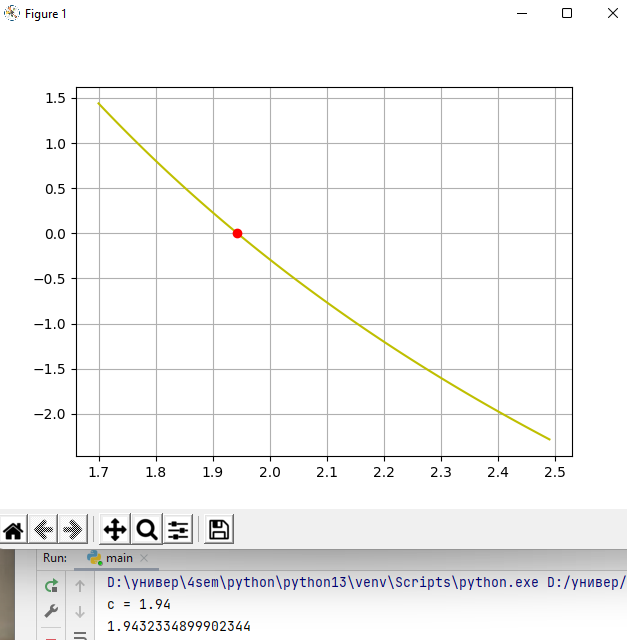


Рисунок 3 – Результат выполнения задания 3

**Вывод:**

В ходе данной лабораторной работы освоила навыки решения уравнений различных видов с использованием численных методов, научилась решать прикладные задачи с решением уравнений при помощи языка программирования Python.