Университет ИТМО

Факультет ФПИ и КТ

### Отчёт

### по лабораторной работе 2

«**Численное решение нелинейных уравнений и систем**»

Вариант 10

Студент:

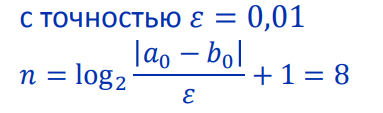
Преподаватель:  
Малышева Татьяна Алексеевна

**1 -3.125 -3.5 2.458**

**Вычислительная реализация задачи:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № шага | a | b | x | f(a) | f(b) | f(x) | |a-b| |
| 1 | -2 | -1 | -1.5 | -11.042 | 1.833 | -2.698 | 1 |
| 2 | -1.5 | -1 | -1.25 | -2.698 | 1.833 | -0.00294 | 0.5 |
| 3 | -1.25 | -1 | -1.125 | -0.00294 | 1.833 | 1.016594 | 0.25 |
| 4 | -1.25 | -1.125 | -1.1875 | -0.00294 | 1.016594 | 0.532951 | 0.125 |
| 5 | -1.25 | -1.1875 | -1.21875 | -0.00294 | 0.532951 | 0.271629 | 0.0625 |
| 6 | -1.25 | -1.21875 | -1.234375 | -0.00294 | 0.271629 | 0.1360128 | 0.03125 |
| 7 | -1.25 | -1.23437 | -1.2421875 | -0.00294 | 0.1360128 | 0.066955 | 0.015625 |
| 8 | -1.25 | -1.24218 | ***-1.2460937*** | -0.00294 | 0.066955 | 0.032114353 | 0.0078125 |

1)Метод половинного деления(Крайний левый корень):



1. Метод простой итерации(Центральный корень):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № шага |  |  |  | │xk+1 − xk│ |
| 1 | 0 | 0.702285714 | -1.194895343 | 0.702285714 |
| 2 | 0.702285714 | 0.360887045 | 0.834898767 | 0.341398669 |
| 3 | 0.360887045 | 0.59942955 | -0.547480753 | 0.238542505 |
| 4 | 0.59942955 | 0.443006477 | 0.38112339 | 0.156423072 |
| 5 | 0.443006477 | 0.551898875 | 0.257393025 | 0.108892397 |
| 6 | 0.551898875 | 0.47835801 | 0.178125443 | 0.073540864 |
| 7 | 0.47835801 | 0.529250994 | -0.121464945 | 0.050892984 |
| 8 | 0.529250994 | 0.494546724 | 0.08373951 | 0.03470427 |
| 9 | 0.494546724 | 0.518472298 | -0.057322941 | 0.023925574 |
| 10 | 0.518472298 | 0.502094315 | 0.03943878 | 0.016377983 |
| 11 | 0.502094315 | 0.513362538 | -0.027042681 | 0.011268223 |
| 12 | 0.513362538 | **0.505636058** | 0.018586721 | 0.00772648 |

3)Метод Ньютона(Крайний правый корень):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № шага | xk | f(xk ) | F’(xk ) |  | │xk+1 − xk│ |
| 1 | 4 | 2.458 | 19.5 | 3.873948718 | 0.126051282 |
| 2 | 3.873948718 | 0.139011396 | 17,30125652 | 3.865918138 | 0.00803058 |
| 3 | 3.865918138 | 0.000547446 | 17.17398079 | **3.865886262** | 0.0000318764 |

Program：

def chord(equation, interval):  
 graph.draw\_graph(equation, -5, 5)  
 check\_interval(equation, interval)  
 epsilon = parse\_float\_array(input("Enter epsilon\n").replace(",", ".").split())[0]  
 equation = list(equation)  
 x\_prev2 = interval[0]  
 x\_next2 = interval[1]  
 while abs(x\_next2 - x\_prev2) > epsilon:  
 t = ((x\_next2 \* function(equation, x\_prev2)) - (x\_prev2 \* function(equation, x\_next2))) / (  
 function(equation, x\_prev2) - function(equation, x\_next2))  
 if abs(x\_next2 - t) > abs(x\_prev2 - t):  
 x\_next2 = t  
 else:  
 x\_prev2 = t  
 return x\_next2  
  
  
def newton(equation, interval):  
 graph.draw\_graph(equation, -5, 5)  
 check\_interval(equation, interval)  
 epsilon = parse\_float\_array(input("Enter epsilon\n").replace(",", ".").split())[0]  
 equation = list(equation)  
 x\_prev = interval[0]  
 x\_next = interval[1]  
 derivativee = derivative(equation)  
 while abs(x\_next - x\_prev) > epsilon:  
 tmp = x\_next  
 x\_next = x\_next - (function(equation, x\_next) / function(derivativee, x\_next))  
 x\_prev = tmp  
 return x\_next  
  
  
def simple(equation, interval):  
 graph.draw\_graph(equation, -5, 5)  
 check\_interval(equation, interval)  
 epsilon = parse\_float\_array(input("Enter epsilon\n").replace(",", ".").split())[0]  
 equation = list(equation)  
 a = interval[0]  
 b = interval[1]  
 derivative\_eq = derivative(equation)  
 if (function(derivative\_eq, a)) > function(derivative\_eq, b):  
 max\_derivative = function(derivative\_eq, a)  
 xi = a  
 else:  
 max\_derivative = function(derivative\_eq, b)  
 xi = b  
 lambda\_var = - 1 / max\_derivative  
 phi = equation[:]  
 for i in range(len(equation)):  
 phi[i] \*= lambda\_var  
 if len(phi) < 2:  
 phi.append(1)  
 else:  
 phi[1] += 1  
  
 while abs(function(phi, xi) - xi) > epsilon:  
 xi = function(phi, xi)  
  
 return function(phi, xi)