**MINISTERUL EDUCAŢIEI, CULTURII ȘI CERCETĂRII AL REPUBLICII MOLDOVA**

**UNIVERSITATEA TEHNICĂ A MOLDOVEI**

**Facultatea Calculatoare, Informatică și Microelectronică**

**Departamentul Ingineria Software și Automatică**

**RAPORT**

**la lucrarea de laborator nr.1**

**Tema: „Rezolvarea numerică a ecuațiilor algebrice și transcendente”**

**Disciplina: „Metode Numerice ”**

**A elaborat *st. gr. SI-211, Vozian Vladimir***

**A verificat *lect.univ V.Struna***

**Chișinău 2022**

**Scopul lucrării :**

* Să se separe toate rădăcinile reale ale ecuaţiei .
* Să se determine o rădăcină reală a ecuaţiei aplicând metodele : înjumătăţirii intervalului , aproximațiilor succesive , tangentelor ( Newton ) .
* Să se aprecieze rădăcina obţinută cu exactitatea .
* Să se compare rezultatele luând în consideraţie numărul de iteraţii , evaluările pentru funcţii şi derivată .

***Varianta 23***

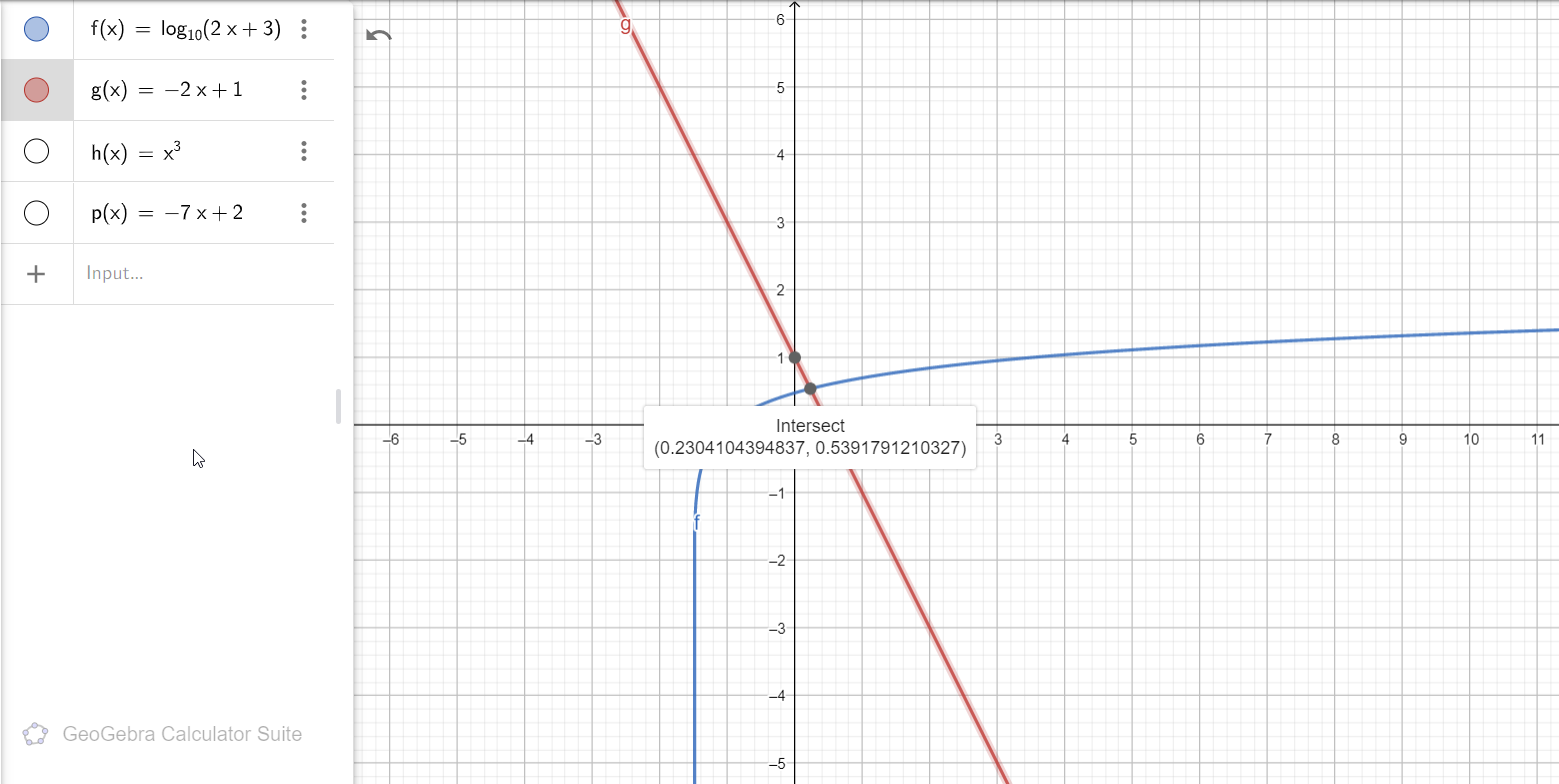
**Sarcina problemei :**

Să se găsească rădăcinile reale ale ecuaţiei :

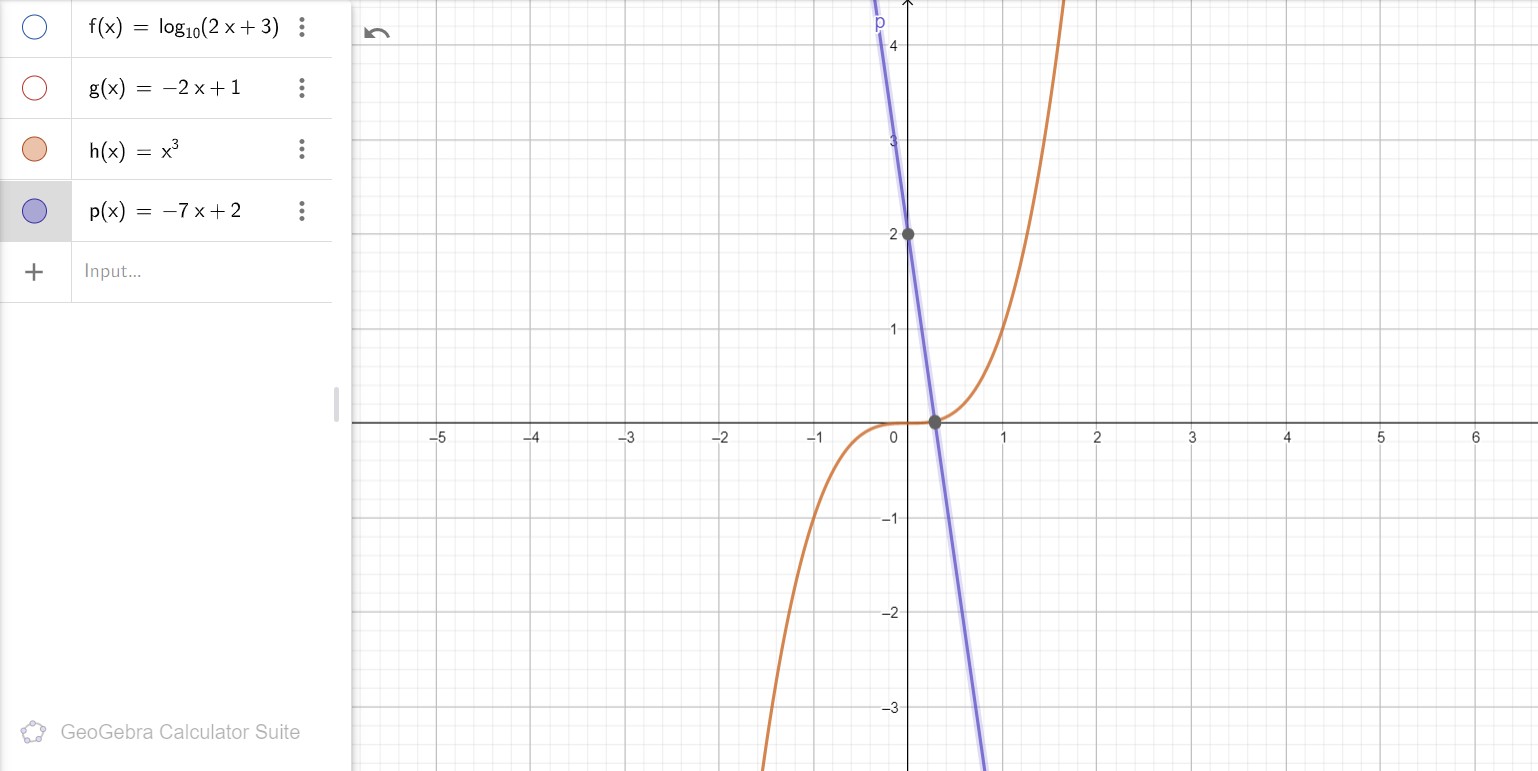
și

1. **Metoda graficelor**

Pentru f1(x):



pentru f2(x):



1. **Metoda analitica**

pentru f1(x)

f 1(x) =

df1(x) = =0

df1(x) =>

|  |  |  |  |
| --- | --- | --- | --- |
| **x** | **-1.717** | **0** | **1.717** |
| **y** | **-4.7** | **-0.52** | **3.24** |

Semnul se schimba pe intervalul [0,1.717] => radacina in intervalul [0,1.717]

pentru f2(x)

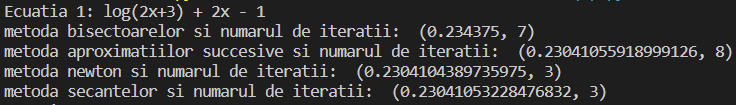
f2(x) = x3 +7x-2

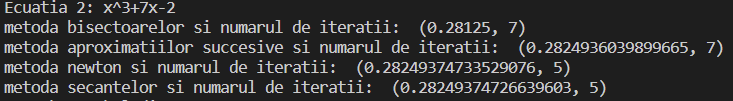
df2(x) = 3x2 -7

df2(x)=>imaginar

1. **Codul**
2. import math
3. #log(2x+3) + 2x - 1
4. def f1(x):
5. return math.log10(2\*x+3) + 2\*x - 1
6. # fi (pentru medota aproximarilor)
7. def f1fi(x):
8. return (1 - math.log10(2\*x+3))/2
9. #f1 derivat si dublu derivat pentru metoda newton
10. def f1d(x):
11. return 2/(math.log(10)\*(2\*x+3)) + 2
12. def f1dd(x):
13. return -4/(math.log(10)\*(2\*x+3)\*\*2)
14. #x^3 + 7x - 2
15. def f2(x):
16. return x\*\*3 + 7\*x -2
17. # fi (pentru medota aproximarilor)
18. def f2fi(x):
19. return (x\*\*3-2)/(-7)
20. #f2 derivat si dublu derivat pentru metoda newton
21. def f2d(x):
22. return 3\*x\*\*2 + 7
23. def f2dd(x):
24. return 6\*x
25. #metoda injumatatirii intervalelor (bisectoarelor)
26. def bisect(func,a,b):
27. x = a
28. y = func(x)
29. count=0
30. if(func(a) \* func(b) < 0):
31. while(abs(y) > 10\*\*(-2)):
32. count+=1
33. x = (a+b)/2
34. y = func(x)
35. if(func(a)\*y < 0):
36. b = x
37. else:
38. a = x
39. return x, count
40. #metoda aproximatiilor succesive
41. def aprox(func, a):
42. count=0
43. while True:
44. count+=1
45. x = a
46. a = func(x)
47. if(abs(a - x) < 10\*\*(-6)):
48. return x, count
49. #metoda newton
50. def newton(func, dFunc, ddFunc, a, b):
51. count=0
52. if(func(a) \* ddFunc(a) > 0):
53. x = x1 = a
54. else:
55. x = x1 = b
56. while True:
57. count+=1
58. x = x1
59. x1 = x - func(x) / dFunc(x)
60. if(abs(x1-x) < 10\*\*(-6)):
61. return x1, count
62. #metoda secantelor
63. def secant(func, a, b):
64. count = 0
65. x1 = 0
66. x2 = 0
67. if (func(a) \* func(b) < 0):
68. while True:
69. count = count + 1
70. x1 = (a \* func(b) - b \* func(a)) / (func(b) - func(a))
71. if (func(a) \* func(x1) == 0):
72. break
73. a = b
74. b = x1
75. x2 = (a \* func(b) - b \* func(a)) / (func(b) - func(a))
76. if (abs(x2 - x1) < 10\*\*(-6)):
77. return x1, count
78. print("Ecuatia 1: log(2x+3) + 2x - 1")
79. print("metoda bisectoarelor si numarul de iteratii: ", bisect(f1, 0, 2))
80. print("metoda aproximatiilor succesive si numarul de iteratii: ", aprox(f1fi, 0))
81. print("metoda newton si numarul de iteratii: ", newton(f1, f1d, f1dd, 0, 2))
82. print("metoda secantelor si numarul de iteratii: ", secant(f1, 0, 2))
83. print("Ecuatia 2: x^3+7x-2")
84. print("metoda bisectoarelor si numarul de iteratii: ", bisect(f2, -1, 1))
85. print("metoda aproximatiilor succesive si numarul de iteratii: ", aprox(f2fi, -2))
86. print("metoda newton si numarul de iteratii: ", newton(f2, f2d, f2dd, -1, 1))
87. print("metoda secantelor si numarul de iteratii: ", secant(f2, -1, 1)

**Rezultatele**





**Compararea rezultatelor**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Metoda | Rădăcina () | | Nr. de iterații | |  | Eroarea ε | |
|  |  |  |  |
| Bisecției | 0.234375 | 0.28125 | 7 | 7 |  | |
| Apr. succesive | 0.23041055... | 0.2824936... | 8 | 7 |  | |
| Newton | 0.23041043... | 0.28249374... | 3 | 5 |  | |
| Secantei | 0.23041053... | 0.282493747... | 3 | 5 |  | |

**Concluzie :**

În urma efectuării acestei lucrări de laborator am realizat calculul rădăcinilor reale a unei ecuații prin prisma a 4 metode. Eficacitatea metodelor utilizate poate fi determinată în baza tabelului cu rezultatele calculelor. Din tabel observăm că metoda secantelor oferă un rezultat mult mai precis (decât metoda înjumătățirii intervalelor) și necesită efectuarea a unui număr mai redus de iterații decât celelalte metode, în pofida acestui fapt metoda tangentelor este mai puțin practică, deoarece presupune calculul funcției derivate, ceea ce nu întotdeauna poate fi realizat cu o bună precizie.