**Ministerul Educației, Culturii și Cercetării**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică și Microelectronică**

**Departamentul Ingineria Software și Automatică**

**Raport**

Lucrarea de laborator nr.1

Disciplina: Metode și modele de calcul

Tema: Rezolvarea numerica a ecuatiilor algebrice si transcendenta.Separarea radacinilor.

**Efectuat**: st.gr.TI-207 Bunescu Gabriel.

**Verificat**: conf. univ. dr. Dohotaru Leonid

Chișinău 2021

**Scopul lucrării:**

1. Rezolvarea numerică a ecuației algebrice și transcendente

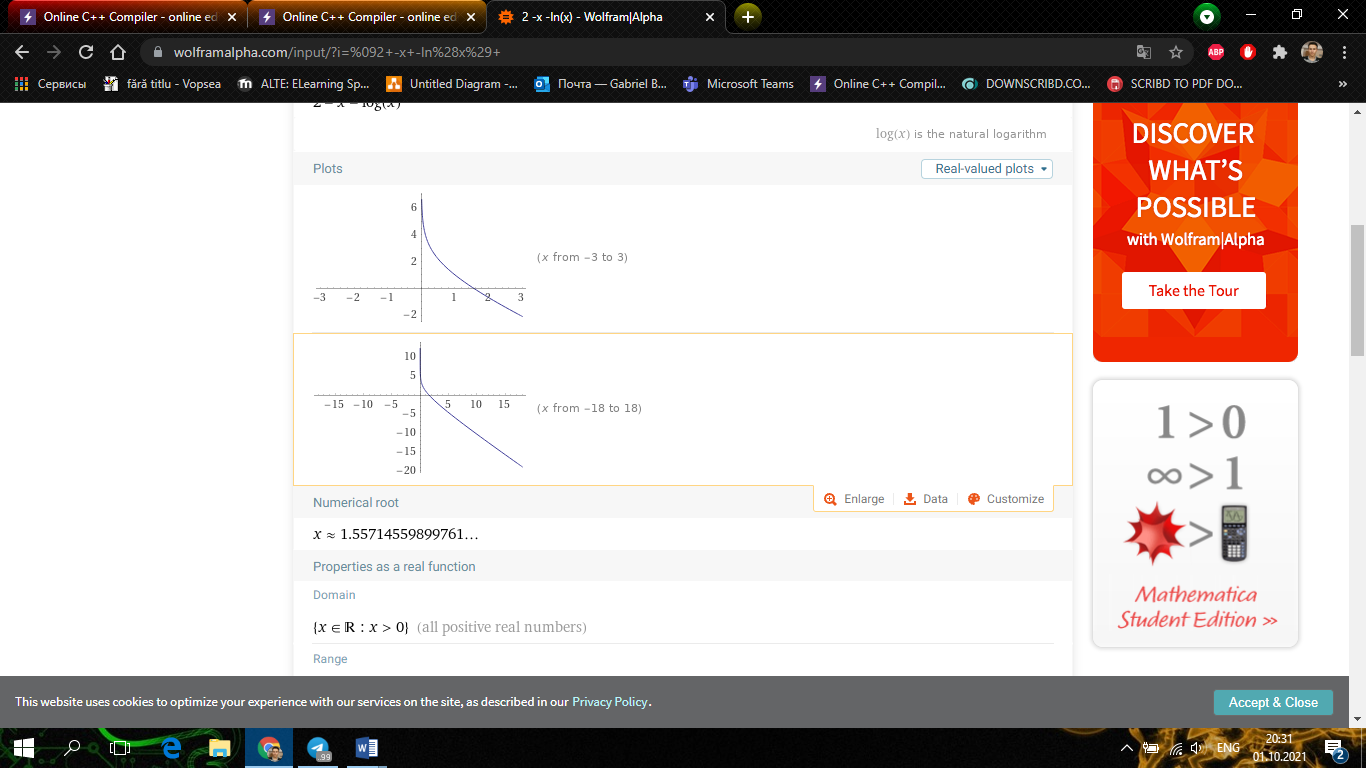
**Obiectivele lucrări:**

1. Să se separe toate rădăcinile ecuației f(x) = 0 unde y = f(x) de variabilă reală.
2. Să se determine o rădăcină reală a ecuației date cu ajutorul metodelor înjumătățiri intervalului cu o eroare mai mică decât ε = 10^-2.
3. Să se precizeze rădăcina obținută cu exactitatea ε = 10^-6 cu ajutorul metodelor:
4. Metoda aproximațiilor succesive.
5. Metoda secantelor.
6. Metoda tangentelor (Newton).
7. Să se compare rezultatele luând în considerație numărul de iterații pentru funcție și derivată.

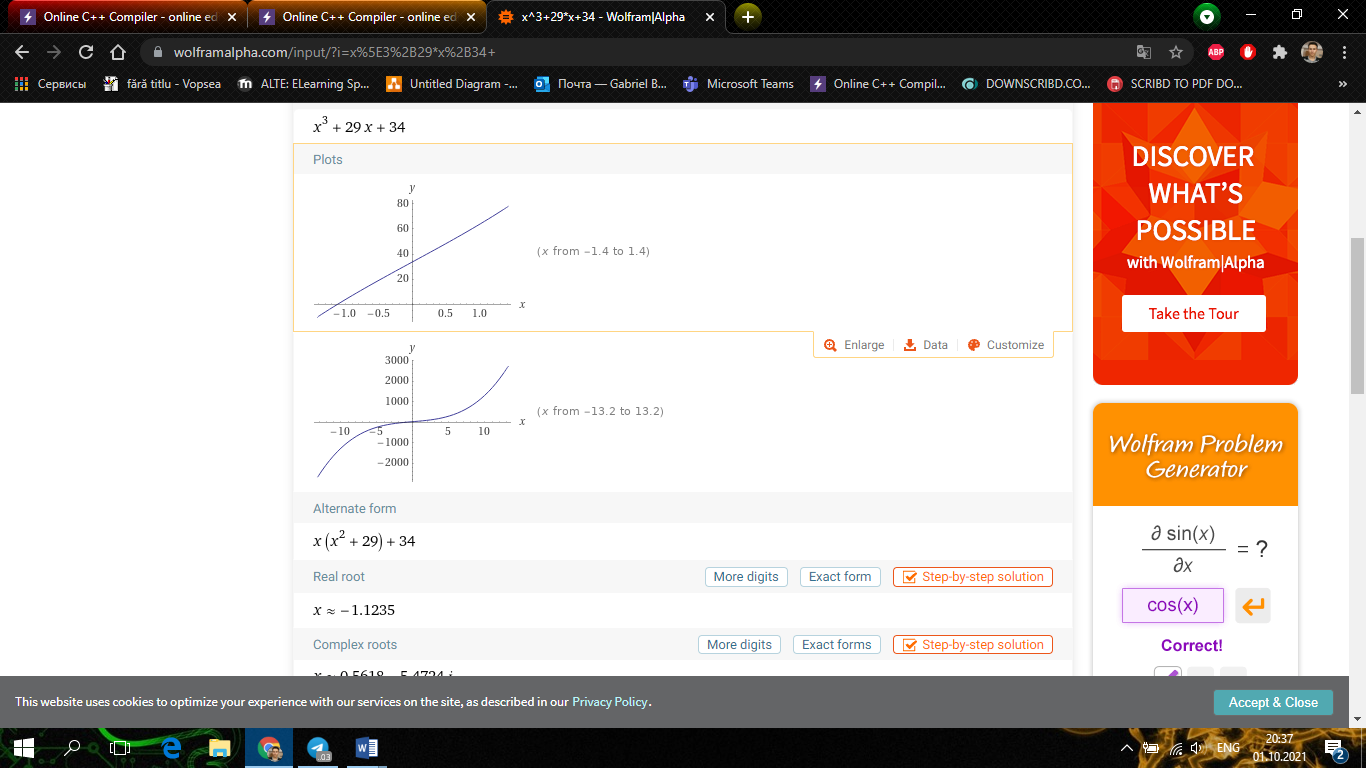
**Problemă dată spre rezolvare:**

Varianta 5:

1. 2 -x -ln(x)
2. X3+29x+34



b)



**Calculul aproximativ al radacinilor**

Metoda injumatatirii intervalului

Consideram functia 𝑓 ( 𝑥 ) = 0, unde functia 𝑓 ( 𝑥 ) este continua pe intervalul [a,b], are o singura radacina reala si 𝑓 ( 𝑎 ) ∗ 𝑓 ( 𝑏 ) < 0.

b−a Se calculeaza c = a + a , daca c=0, atunci c este radacina cautata, in caz contrar :

daca 𝑓 ( 𝑎 ) ∗ 𝑓 ( 𝑏 ) < 0 => 𝑎 = 𝑎; 𝑏 = 𝑐 ;

daca 𝑓 ( 𝑎 ) ∗ 𝑓 ( 𝑏 ) > 0 => 𝑎 = 𝑐 ; 𝑏 = 𝑏 ;

Iteratiile se repeta pina cind este respectata conditia (b-a)<ε unde ε este eroarea

Metoda aproximatiilor succesive

Pentru a folosi Metoda aproximatiilor succesive, e necesar de pus ecuatia sub forma 𝑥 = 𝜑(𝑥). Pentru ca metoda sa convearga catre radacina e necesar de a se respecta conditia de convergenta :

Functia 𝜑(𝑥) e derivabila si derivata sa 𝜑′(𝑥) satisface inegalitatea |𝜑′ ( 𝑥 ) | ≤ 𝛼 < 1, oricare ar fi 𝑥 ∈ [ 𝑎, 𝑏 ] .

Metoda lui Newton(tangentelor)

Presupunem ecuatia 𝑓 ( 𝑥 ) = 0, care admite o singura solutie reala pe intervalul [a,b], iar derivatele 𝑓 ′ (𝑥) si 𝑓 ′′ ( 𝑥 ) pastreaza un semn constant pe intervalul [a,b].

Formula pentru metoda tangentelor este :

𝑓(𝑥 𝑘 ) 𝑘+1 = 𝑥 𝑘 –x

, 𝑘 = 0,1,2, …𝑓′(𝑥 𝑘 )

Pentru a asigura convergenta procesului iterational e necesar ca din valorile a si b sa fie aleasa in calitate de solutie initiala 𝑥 0 acea valoare pentru care are loc inegalitatea 𝑓 ( 𝑥 ) ∗ 𝑓 ′′ ( 𝑥 ) > 0

Metoda secantelor

Metoda secantelor poate fi dedusa din metoda lui Newton. Formula iteativa pentru calcularea aproximativa a solutiilor prin metoda secantelor este :

𝑥 𝑘 − 𝑥 𝑘−1 𝑥 𝑘+1 = 𝑥 𝑘 − 𝑓 ( 𝑥 𝑘 ) ∗ , 𝑘 = 0,1, … 𝑓 ( 𝑥 𝑘 ) − 𝑓(𝑥 𝑘−1 )

La fel ca pentru Metoda lui Newton in calitate de solutie initiala 𝑥 0 vom alege acea valoare pentru care are loc inegalitatea :

𝑓 ( 𝑥 ) ∗ 𝑓 ′′ ( 𝑥 ) > 0

Finisarea procesului iterational are loc atunci cind aleasa in calitate de solutie initiala 𝑥 0 acea valoare pentru care are loc inegalitatea |𝑥 𝑘+1 − 𝑥 𝑘 | ≤ 𝜀, unde 𝜀 este eroarea.

**Listningul programului:**

#include <iostream>

#include <math.h>

using namespace std;

double(\*f)(double), (\*fn)(double), (\*fd)(double);

double f1(double x) {

return pow(x, 3) + 29 \* x + 34;

}

double f2(double x) {

return 2-x-log(x);

}

double fd1(double x)

{

return 3 \* pow(x, 2) + 29;

}

double fd2(double x)

{

return - (1/x) - 1;

}

double f3(double x) {

return (pow(x, 3) + 3) / 29;

}

double f4(double x) {

return (log(x) - 2) / -x;

}

void Met\_aproximatiei()

{

int k = 0;

double x0, x1, eps = 0.000001;

cout << "Introduceti valoarea initiala x0" << endl;

cout << "x0= ";

cin >> x0;

while (1)

{

x1 = fn(x0);

k++;

if (abs(x1 - x0) < eps) {

cout << "Radacina este: " << x0 << " Numarul de iteratii este " <<

k << endl;

break;

}

x0 = x1;

}

}

void Met\_injumatatirii() {

int k = 0;

double a, b, c = 0;

double eps = 0.01;

cout << "Introduceti intervalul a si b : " << endl;

cout << "a=";

cin >> a;

cout << "b=";

cin >> b;

while ((b - a) > eps)

{

k++;

c = (a + b) / 2;

if (f(c) == 0)

break;

if (f(a) \* f(c) < 0)

b = c;

else

a = c;

}

cout << "Radacina este: " << c << endl;

cout << "Numarul de iteratii este : " << k;

}

void Met\_Tangentelor() {

int k = 0;

double x0, x1, eps = 0.000001;

cout << "Introduceti valoarea initiala x0" << endl;

cout << "x0=";

cin >> x0;

while (1) {

x1 = x0 - f(x0) / fd(x0);

k++;

if (abs(x1 - x0) < eps) {

cout << "Radacina este: " << x0 << endl << "numarul de iteratii "

<< k << endl; break;

}

x0 = x1;

}

}

void Met\_Secantelor() {

double x2, x1, x3 = 0, y, eps = 0.000001; int n = 0;

cout << "Introduceti intervalul" << endl;

cout << "a=";

cin >> x1;

cout << "b=";

cin >> x2;

do {

n++;

y = x3;

x3 = x2 - (f(x2) \* (x2 - x1) / (f(x2) - f(x1)));

x1 = x2;

x2 = x3;

} while (fabs(y - x3) >= eps);

cout << "Radacina este: " << x3 << endl;

cout << "Numarul de iteratii: " << n << endl;

}

void selectFunction() {

cout << "\n1. Functia f1(x) = 2-x-log(x)" << endl;

cout << "\n2. Functia f2(x) = pow(x, 3) + 29 \* x + 34" << endl;

int opt;

do {

opt = getchar();

} while (opt < '1' || opt < '2');

switch (opt) {

case'1': {

f = f1;

fn = f3;

fd = fd1;

break;

}

case'2': {

f = f2;

fn = f4;

fd = fd2;

break;

}

}

}

int meniu()

{

if (f == f1)

cout << "\nFunctia f1(x)=2-x-log(x)" << endl << endl;

else cout << "\nFunctia f2(x)=pow(x, 3) + 29 \* x + 34" << endl << endl;

cout << "1.Selecatarea functiei " << endl;

cout << "2.Metoda aproximatiei succesive - executia 10^(-6)" << endl;

cout << "3.Metoda injumatatirii - executia 10^(-2)" << endl;

cout << "4.Metoda Tangentelor - executia 10^(-6)" << endl;

cout << "5.Metoda Secantelor - executia 10^(-2)" << endl; cout << "6.Iesire" << endl;

int opt;

do {

opt = getchar();

} while (opt < '1' || opt>'6'); return opt - '0';

}

int main()

{

int opt;

f = f1;

fn = f3;

fd = fd1;

do {

switch (opt = meniu()) {

case 1: {

selectFunction();

break;

}

case 2: {

Met\_aproximatiei();

break;

}

case 3: {

Met\_injumatatirii();

break;

}

case 4: {

Met\_Tangentelor();

break;

}

case 5: {

Met\_Secantelor();

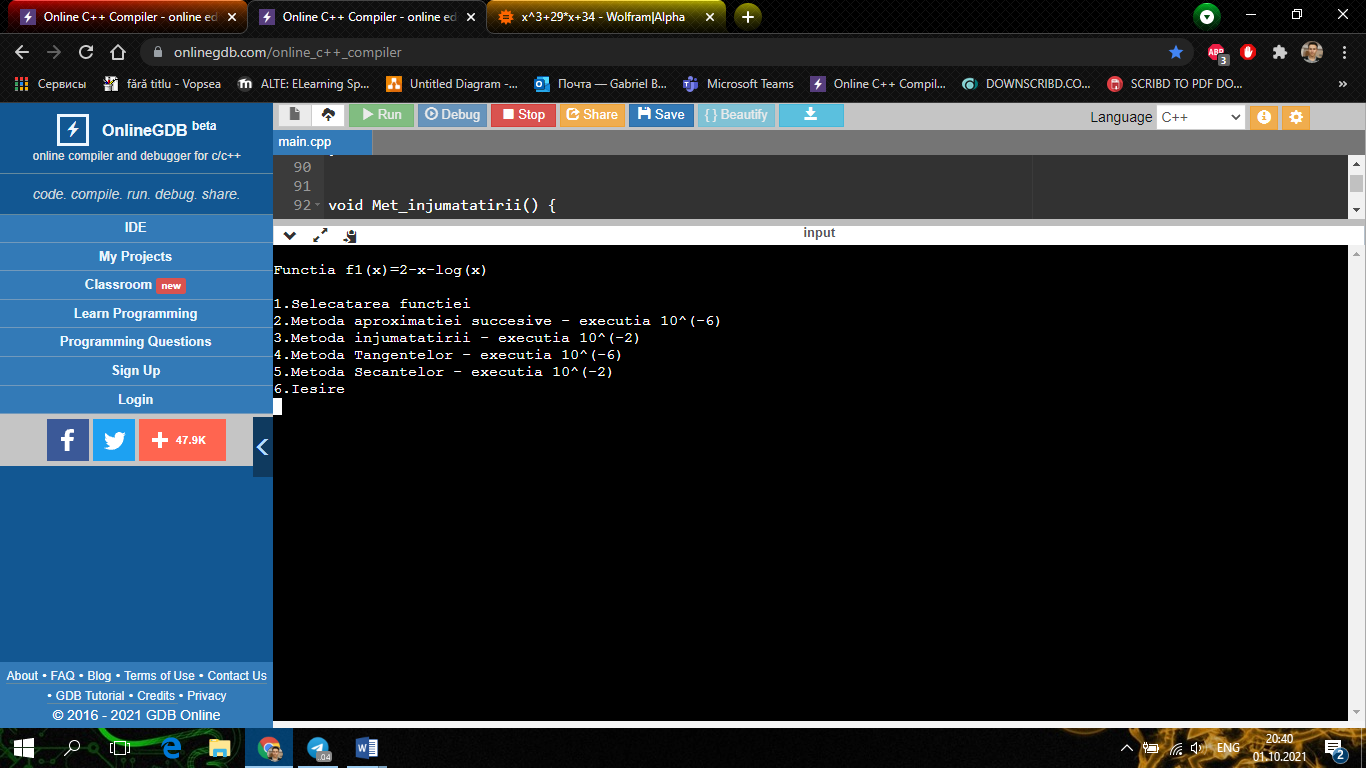
break;

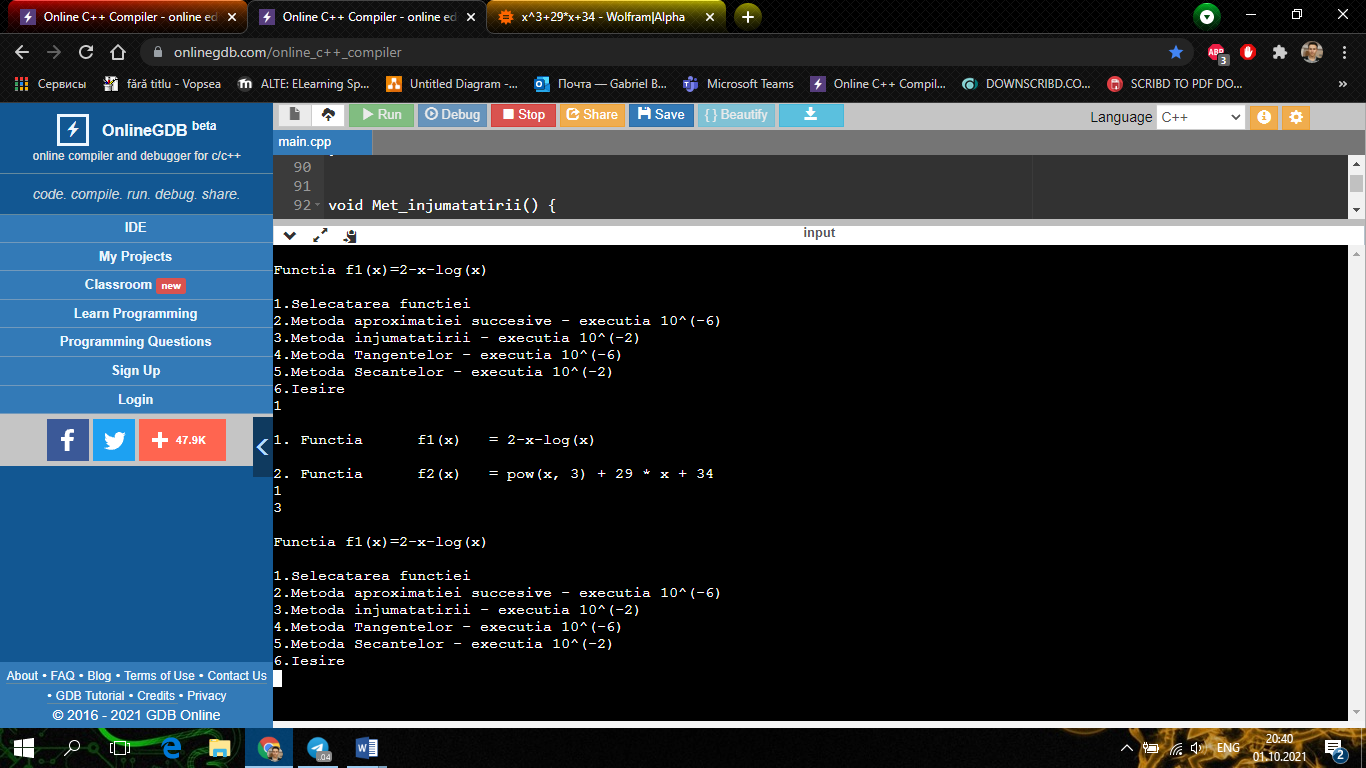
}

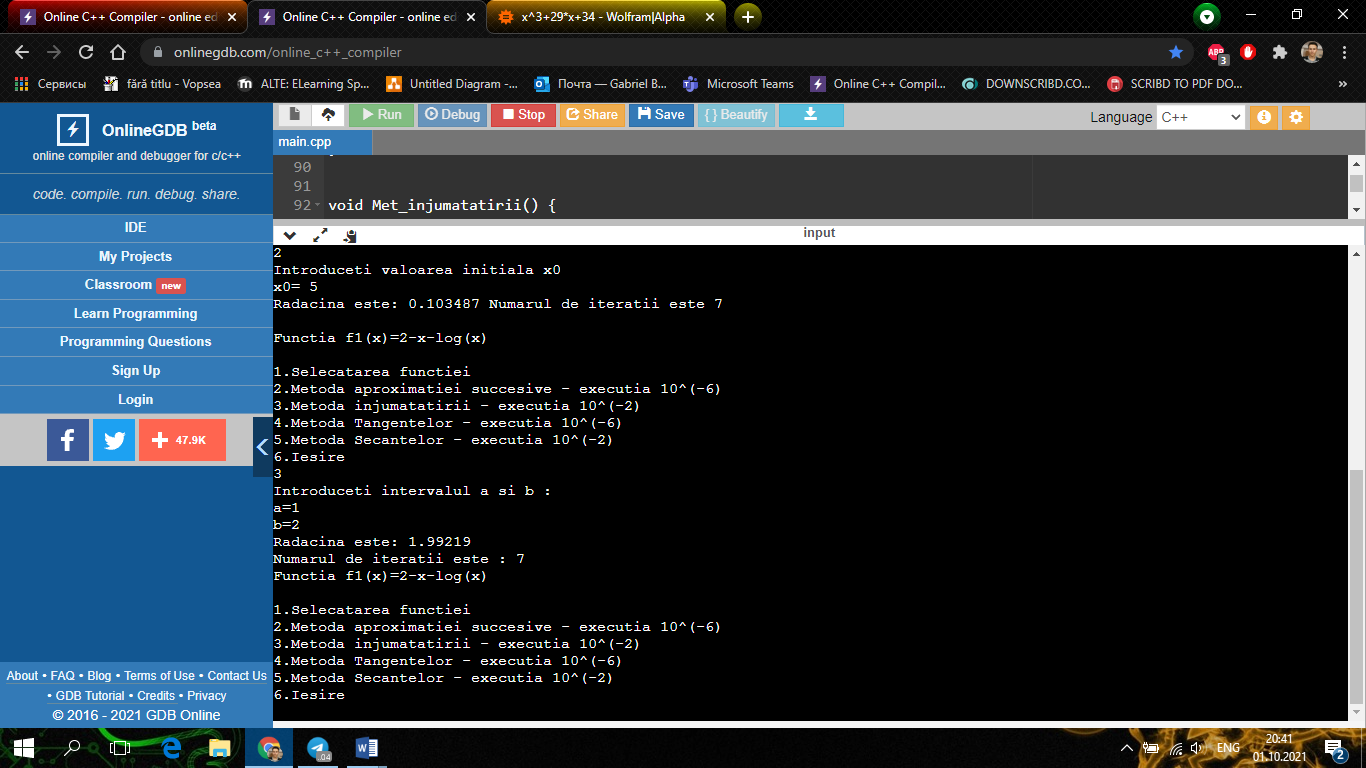
}

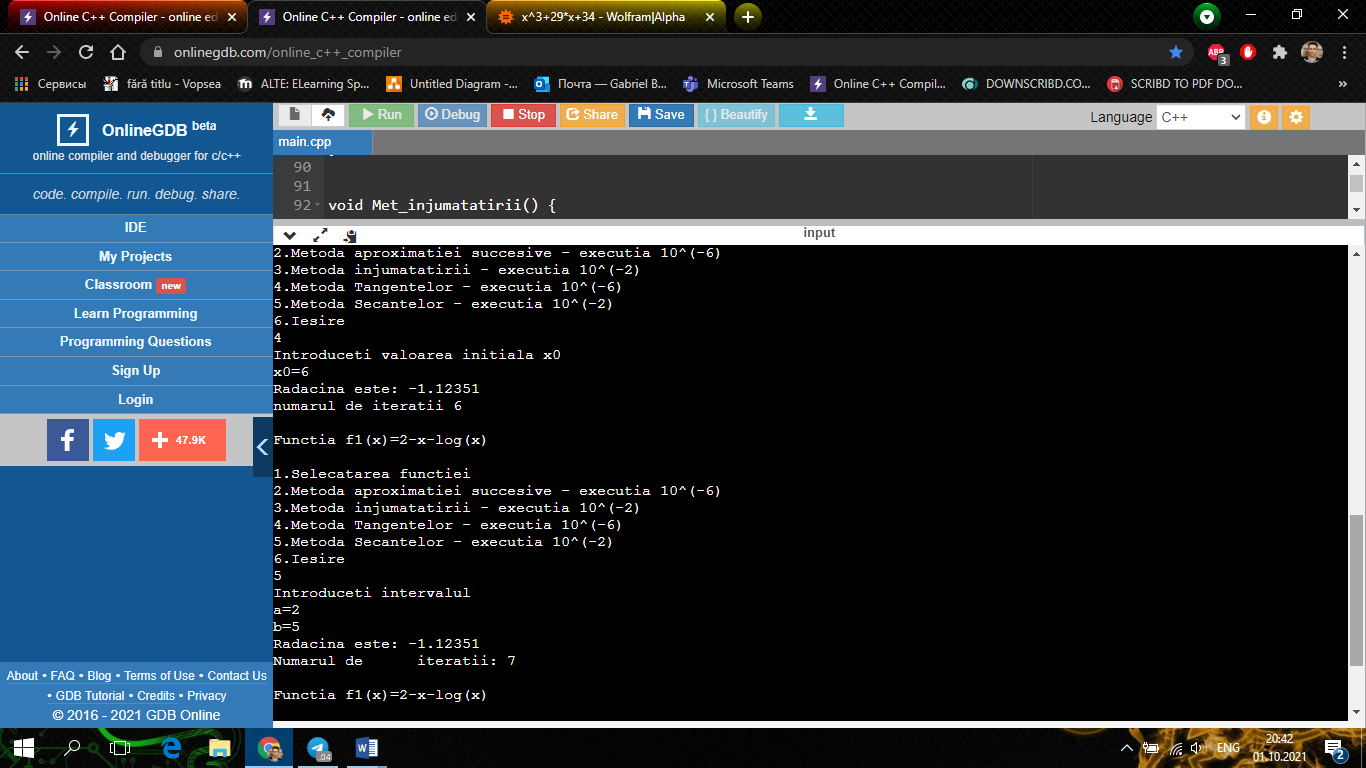
} while (opt != 6);

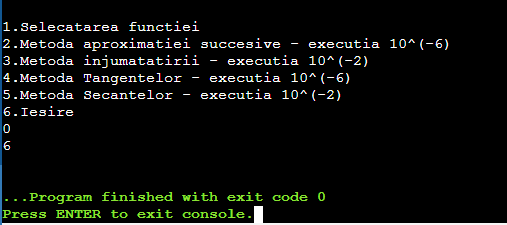
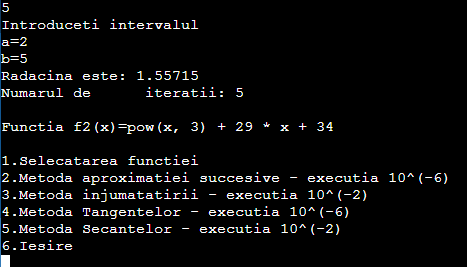
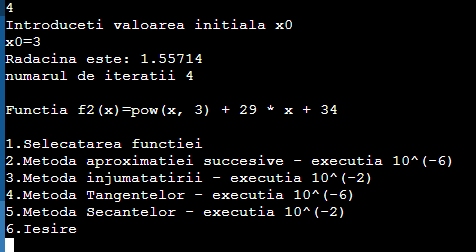
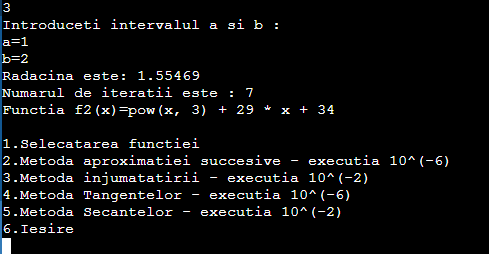
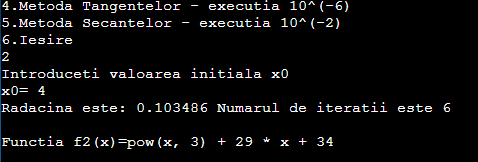
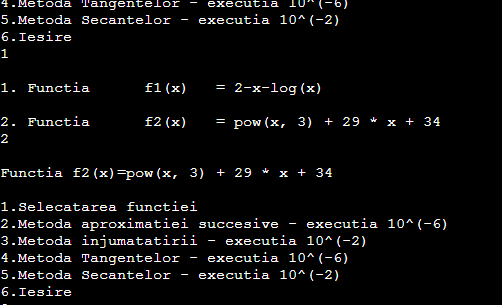
}









****

**Concluzie:**

Efectuind aceasta lucrare de laborator am insusit metodele de rezolvare a ecuatiilor algebrice si transcendente,obtinind aproximativ aceeasi radacina. Totusi cea mai simpla mi s-a parut metoda grafica deoarece nu necesita calcule si usor se poate determina intervalul in care se gaseste solutia,dar cea mai eficienta pare a fi metoda Newton,luind in consideratie ca a facut cele mai putine iteratii pina a fost gasita solutia.