**MINISTERUL EDUCAŢIEI ȘI CERCETĂRII AL REPUBLICII MOLDOVA**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică şi Microelectronică**

**Departamentul Ingineria Software și Automatică**

**Programul de studii: Tehnologia informației**

RAPORT

# LUCRARE DE LABORATOR NR. 1

# la Prelucrarea Semnalelor

**Tema: Introducere în MatLab***.*

A efectuat:

st. gr. TI-211 Popa Cătălin

A verificat: Potlog Mihail

UTM, Chișinău 2023

**Ex1.1**

%E 1.1.Exerciţiu:

%Fie vectorii linie

%a = [0, 0.1, 0.2, … , 2] şi coloană b = [1⋮1]

%a) Ce lungime trebuie să aibă b astfel, încât să aibă sens înmulţirea (în sens matricial) a\*b? Iniţializaţi

%în MATLAB cei doi vectori şi efectuați înmulţirea.

%b) Efectuați înmulţirea b\*a.

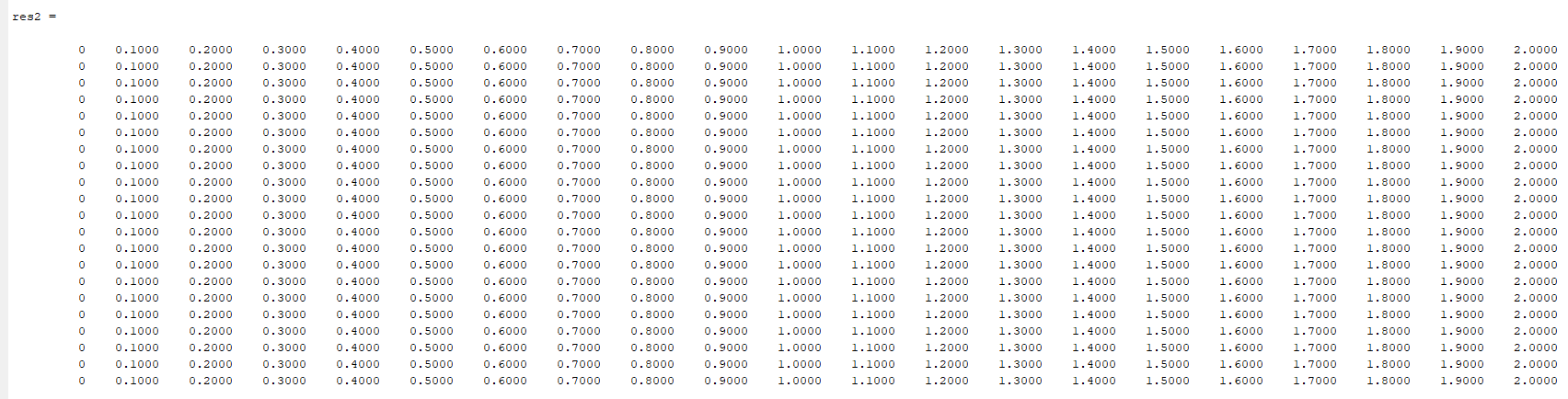
a = 0:0.1:2;

b = ones(length(a),1)

res = a \* b

res2 = b \* a





**Ex1.2**

%E 1.2. Exerciţiu:

%Se creează un fişier nou care trebuie salvat în directorul d:/student/pns/nrgrupa.

%Folosind sintaxele şi indicaţiile din secţiunile 1.2.6 şi 1.2.7, elaborați un program MATLAB care să genereze un vector cu elemente aleatoare cu distribuţie normală

% (gaussiană) şi să afişeze elementele negative ale acestui vector.

% Setarea directorului de lucru

path = 'd:/UTM/ANUL 3/SEM 2/PS/Lab1/lab1';

if ~exist(path, 'dir')

mkdir(path);

end

% Generarea vectorului cu elemente aleatoare cu distribuție normală

num = 100; % Schimbați numărul de elemente după necesități

m = 0; % Medie

deviatia\_standart = 1; % Deviație standard

random\_vector = m + deviatia\_standart \* randn(1, num);

% Afișarea elementelor negative

negative\_elements = random\_vector(random\_vector < 0);

disp('Elemente negative:');

disp(negative\_elements);

% Salvarea elementelor negative într-un fișier

file\_path = fullfile(path, 'elemente\_negative.txt');

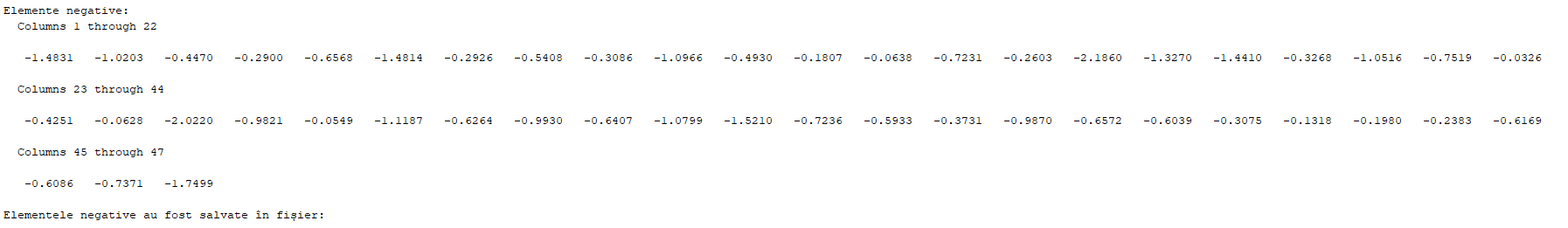
fid = fopen(file\_path, 'w');

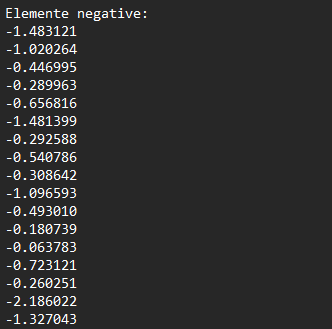
fprintf(fid, 'Elemente negative:\n');

fprintf(fid, '%f\n', negative\_elements);

fclose(fid);

disp(['Elementele negative au fost salvate în fișier: ']);





**Ex 1.3**

%Elaborați un program MATLAB care generază un vector cu elemente complexe. Elaborați (un alt fişier) o funcţie MATLAB care,

%având drept parametru de intrare vectorul cu valori complexe, returnează ca parametri de ieşire:

%- media aritmetică a părţilor reale ale elementelor vectorului;

%- un vector ce conţine elementele vectorului iniţial ridicate la pătrat;

%- o matrice obţinută din înmulţirea vectorului iniţial cu transpusul său.

%Atenţie: pentru a nu se afişa rezultate intermediare din funcţie sau elementele unor variabile se va folosi ; la sfârşitul liniei respective de program.

% Generarea unui vector cu elemente complexe

num\_elements = 5; % Schimbați numărul de elemente după necesități

complex\_vector = randn(1, num\_elements) + 1i \* randn(1, num\_elements);

% Afisarea vectorului generat

disp('Vectorul cu elemente complexe:');

disp(complex\_vector);

% Apelarea funcției și afișarea rezultatelor

addpath('d:/UTM/ANUL 3/SEM 2/PS/Lab1/lab1');

[media\_partilor\_reale, vector\_patrat, matrice\_inmultire] = functie\_complexa(complex\_vector);

disp('Rezultatele functiei:');

disp(['Media partilor reale: ' num2str(media\_partilor\_reale)]);

disp('Vectorul cu elementele ridicate la patrat:');

disp(vector\_patrat);

disp('Matricea obtinuta din inmultirea vectorului cu transpusul sau:');

disp(matrice\_inmultire);

Functia

function [media\_partilor\_reale, vector\_patrat, matrice\_inmultire] = functie\_complexa(complex\_vector)

% Calcularea mediei partilor reale

media\_partilor\_reale = mean(real(complex\_vector));

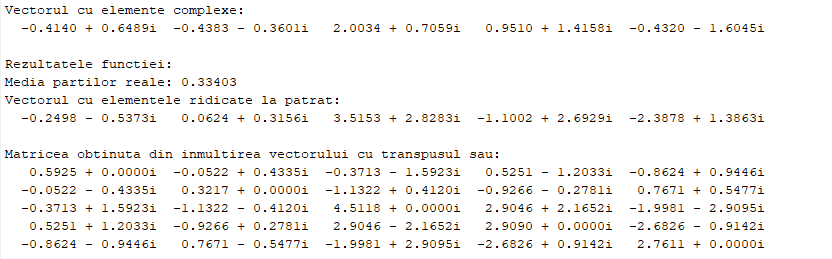
% Calcularea vectorului cu elementele ridicate la patrat

vector\_patrat = complex\_vector.^2;

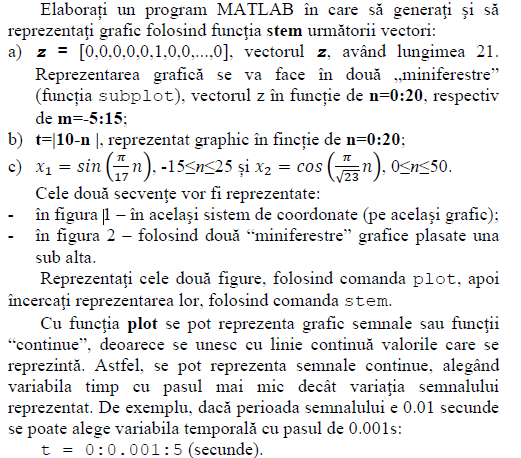
% Calcularea matricei obtinute din inmultirea vectorului cu transpusul sau

matrice\_inmultire = complex\_vector' \* complex\_vector;

end



**Ex 1.4**



z = zeros(1,21);

z(6) = 1;

n = 0:20;

m = -5:15;

% Reprezentarea grafică a lui z în funcție de n=0:20

subplot(2,1,1);

stem(n, z);

% Reprezentarea grafică a lui z în funcție de m=-5:15

subplot(2,1,2);

stem(m, z);

n = 0:20;

t = abs(10 - n);

stem(n, t);

n1 = -15:25;

n2 = 0:50;

x1 = sin((pi/17)\*n1);

x2 = cos((pi/sqrt(23))\*n2);

plot(n1, x1);

hold

plot(n2, x2);

title('x1 si x2')

uiwait(gcf);

subplot(2,1,1);

plot(n1, x1);

title('x1')

subplot(2,1,2);

plot(n2, x2);

title('x2')

uiwait(gcf);

stem(n1, x1);

hold

stem(n2, x2);

title('x1 si x2')

uiwait(gcf);

subplot(2,1,1);

stem(n1, x1);

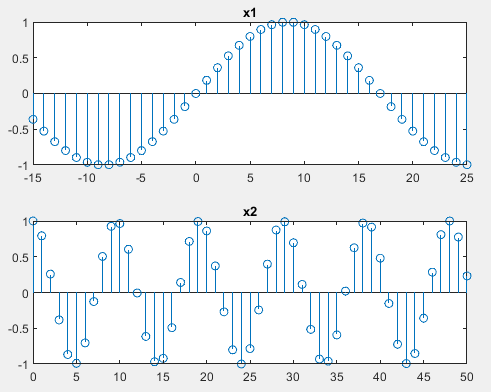
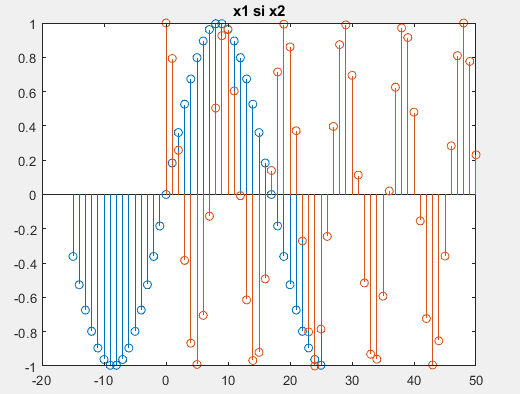
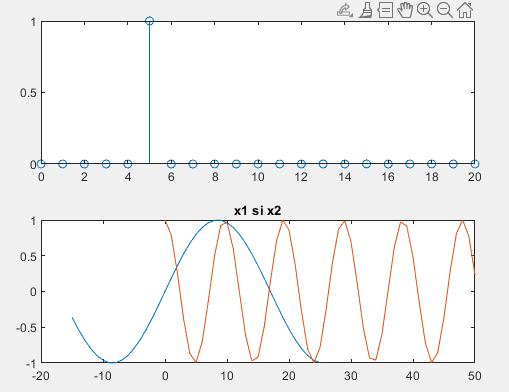
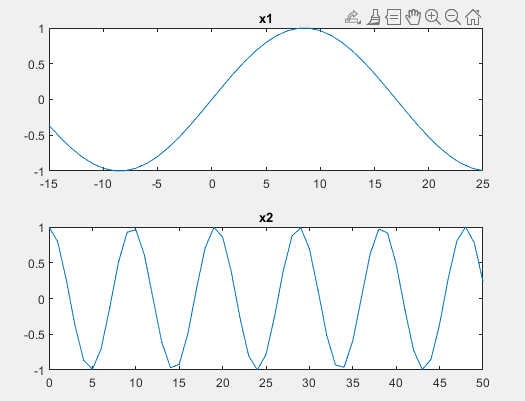
title('x1')

subplot(2,1,2);

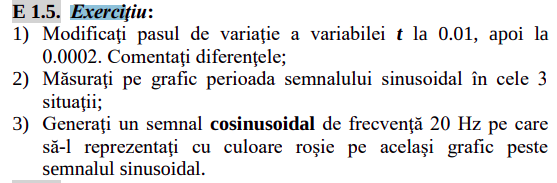
stem(n2, x2);

title('x2')

uiwait(gfc)



**Ex 1.5**



F = 50;

figure(1)

t1 = 0:0.001:0.2;

s1 = 2\*sin(2\*pi\*F\*t1);

plot(t1,s1,'.-'), xlabel('Timp [s]'), grid, title('t1 = 0.001')

figure(2)

t2 = 0:0.01:0.2;

s2 = 2\*sin(2\*pi\*F\*t2);

plot(t2,s2,'.-'), xlabel('Timp [s]'), grid, title('t2 = 0.01')

figure(3)

t3 = 0:0.0002:0.2;

s3 = 2\*sin(2\*pi\*F\*t3);

plot(t3,s3,'.-'), xlabel('Timp [s]'), grid, title('t3 = 0.0002')

figure(4)

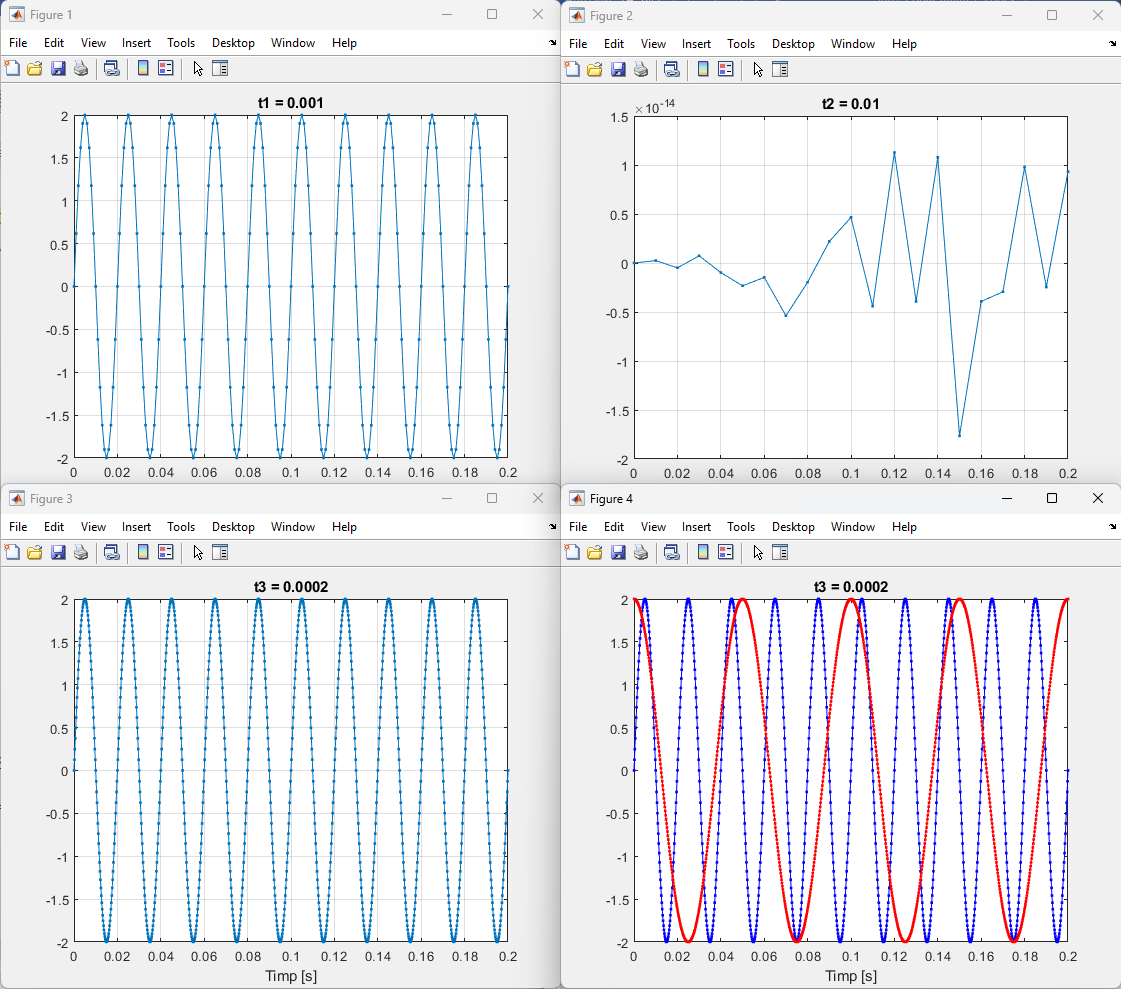
F2 = 20;

c = 2\*cos(2\*pi\*F2\*t3);

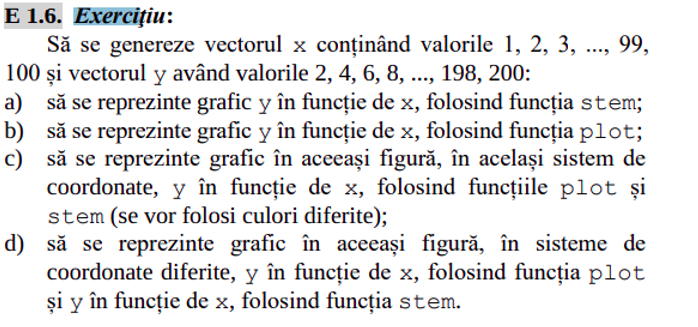
plot(t3,s3,'b.-'), xlabel('Timp [s]'), grid, title('t3 = 0.0002')

hold on

plot(t3,c,'r.-'), xlabel('Timp [s]'), grid, title('t3 = 0.0002')



**Ex 1.6**



x = 1:1:100;

y = 2:2:200;

figure(1)

% a)

stem(x,y), xlabel('x'), ylabel('y')

figure(2)

%b)

plot(x,y), xlabel('x'), ylabel('y')

figure(3)

%c)

stem(x,y)

hold on

plot(x,y, 'r'), xlabel('x'), ylabel('y')

figure(4)

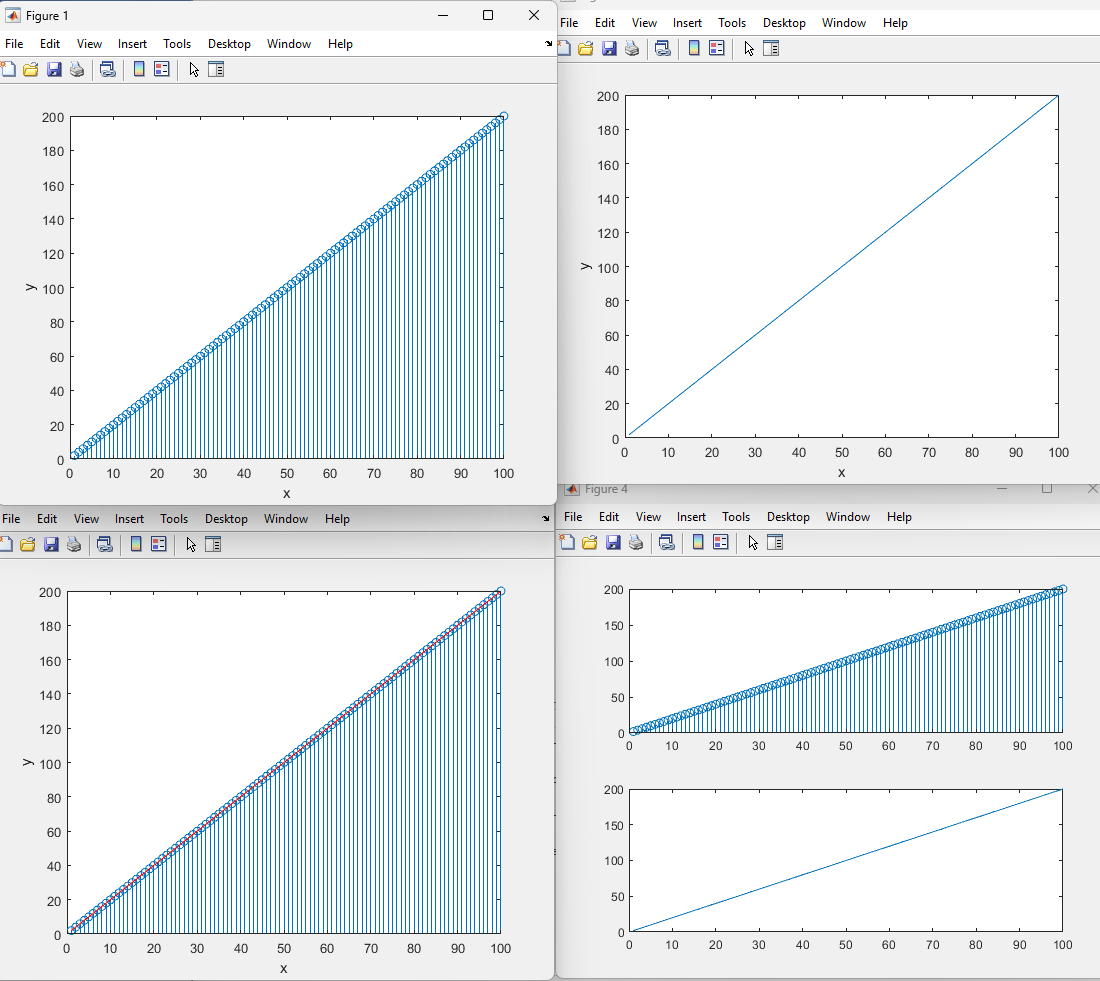
%d)

subplot(2,1,1)

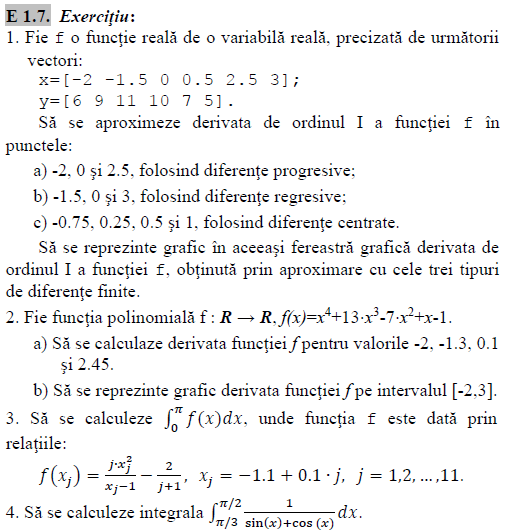
stem(x,y)

subplot(2,1,2)

plot(x,y)



**EX 1.7**



x=[-2 -1.5 0 0.5 2.5 3];

y=[6 9 11 10 7 5];

%a) difrente progresive

prog=diff(y)./diff(x);

vx=[-2 0 2.5];

for i=1:length(vx)

k=find(x==vx(i));

disp(['f''(' num2str(vx(i)) ') = ' num2str(prog(k))])

end

%b) diferente regresive

regr=diff(y)./diff(x);

vx=[-1.5 0 3];

for i=1:length(vx)

k=find(x==vx(i));

disp(['f''(' num2str(vx(i)) ') = ' num2str(regr(k-1))])

end

%c) diferente centrate

dx = diff(x);

dy = diff(y);

k = length(dx);

centr=(dy(1:k-1)+dy(2:k))./(dx(1:k-1)+dx(2:k));

vx=[-0.75 0.25 0.5 1];

for i=1:length(vx)

k=find(x==vx(i));

disp(['f''(' num2str(vx(i)) ') = ' num2str(centr(k))])

end

figure(1)

plot(prog)

figure(2)

plot(regr)

figure(3)

plot(centr)

c=[1 13 -7 1 -1];

dc=polyder(c);

vx=[-2 -1.3 0.1 2.45];

format short g

der=polyval(dc,vx)

x=-2:0.1:3; df=polyval(dc,x);

plot(x,df)

for j=1:11

x(j)= -1.1 + 0.1 \* j;

y(j)=( (j\*(x(j))^2) / (x(j)-1) ) - 2/(j+1);

end

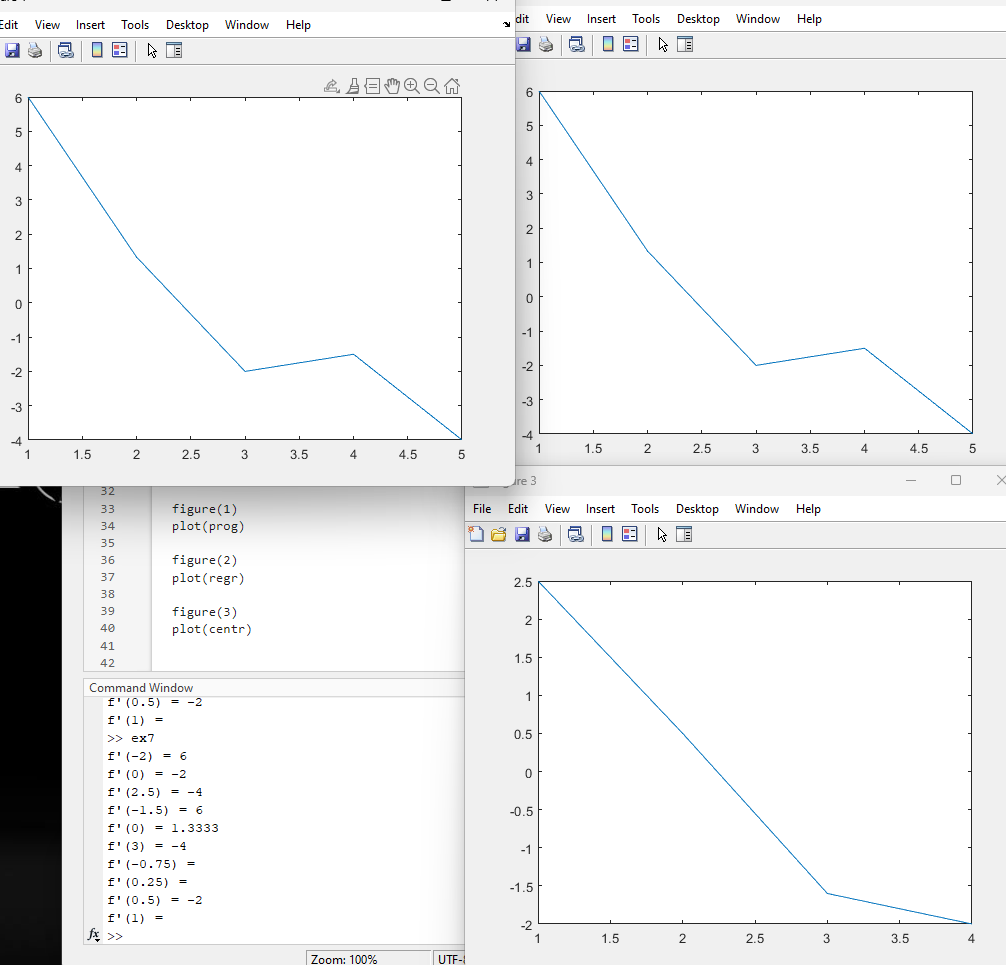
I=trapz(x,y)

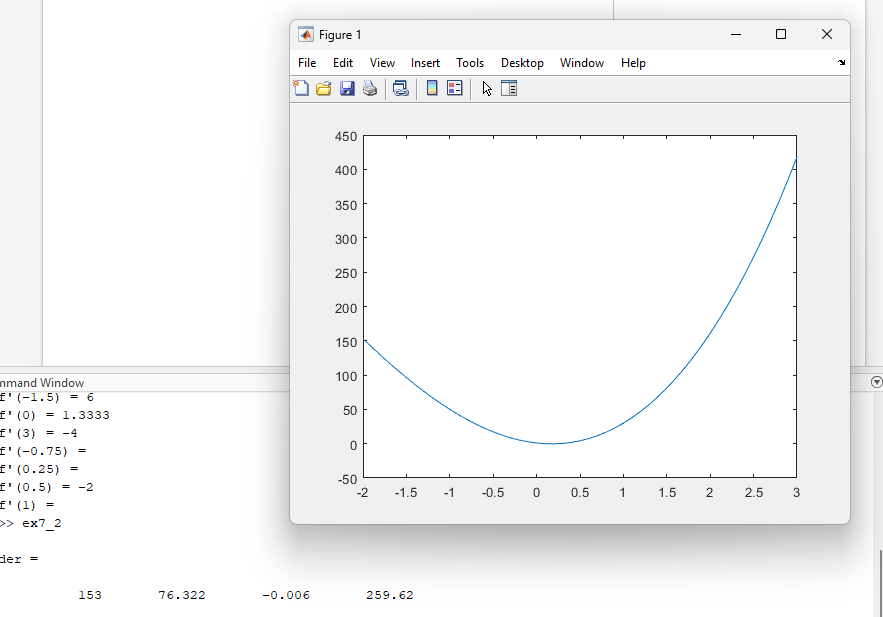
function y=ex\_7\_4\_f(x)

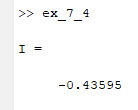
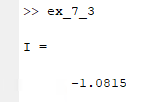
y=(1./((sin(x))+(cos(x))));

end

%I=quad('ex\_7\_4\_f',(pi/2),(pi/3))







**Concluzie**

În cadrul acestui experiment de laborator, am asimilat principiile fundamentale ale MATLAB, inclusiv definirea și manipularea matricelor, gestionarea tipurilor de date și variabilelor, structurile de control, scripturile, precum și reprezentarea grafică și implementarea funcțiilor. În plus, am acumulat cunoștințe despre crearea funcțiilor MATLAB, rezolvarea derivatelor polinomiale și aplicarea practică a metodelor de calcul integral, inclusiv metoda trapezelor și metoda Simpson adaptivă-recursivă. Astfel, după absorbirea informațiilor necesare, am reușit să rezolvăm cu succes sarcinile propuse, iar rezultatele obținute au fost conforme, demonstrând înțelegerea corectă a materialului.