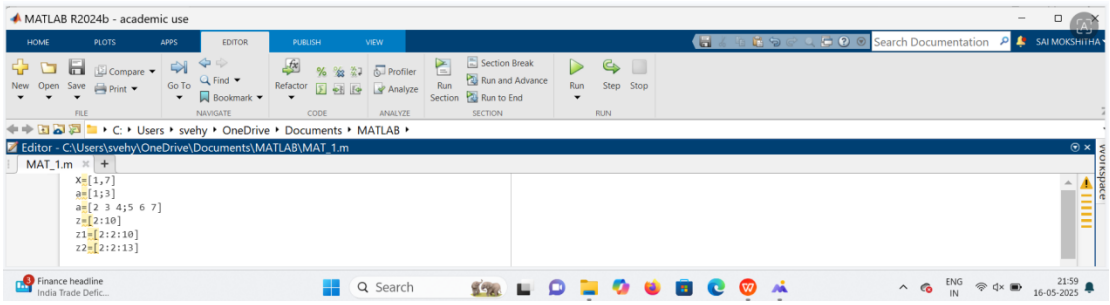
LINEAR ALGEBRA(23MAT117)
LAB MANUAL

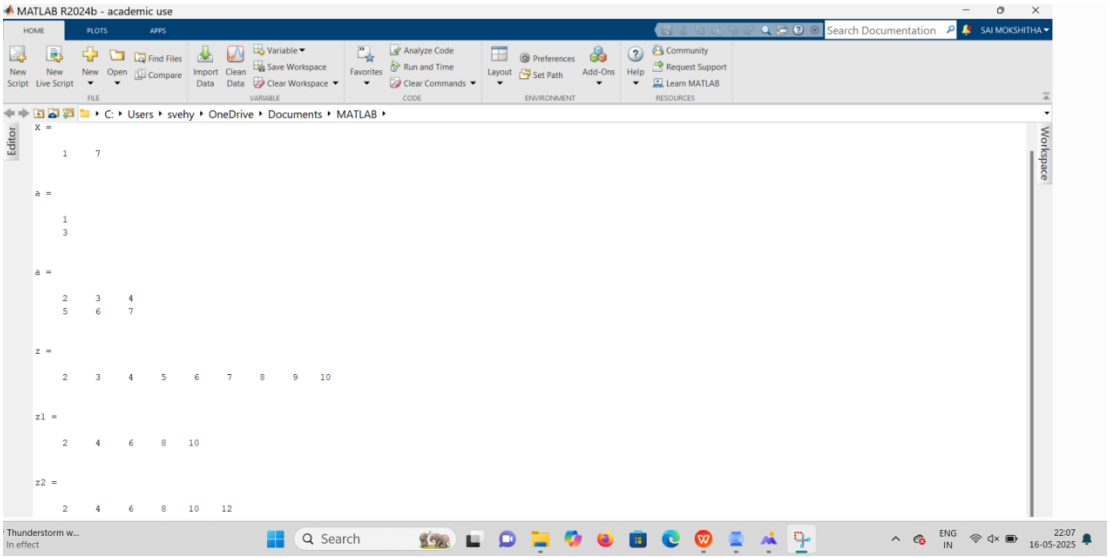
AMRITA SCHOOL OF COMPUTING
AMRITA VISWAVIDYAPEETAM,AMARAVATI CAMPUS

NAME:M.SAI MOKSHITHA
SECTION:CSE-B
ROLL NO:AV.SC.U4CSE24151
SEMESTER:2nd Semester

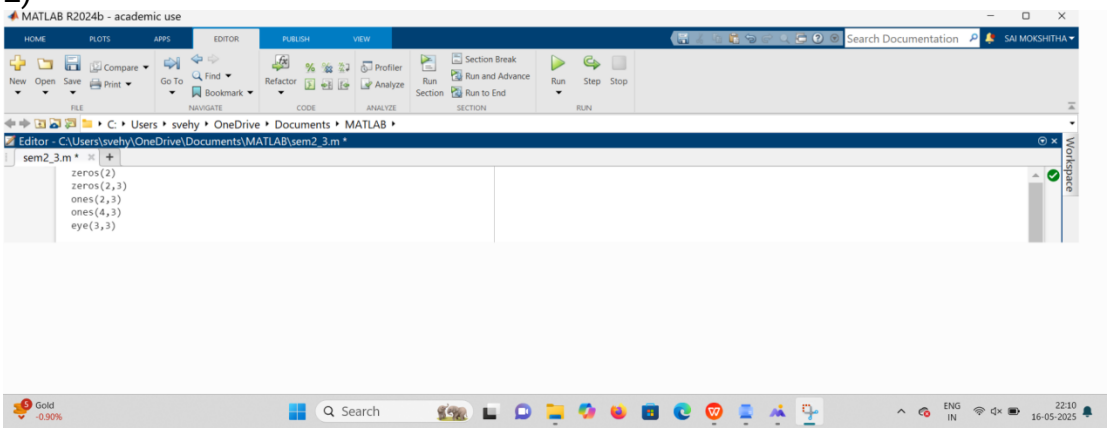
MATRIX OPERATORS:
1)



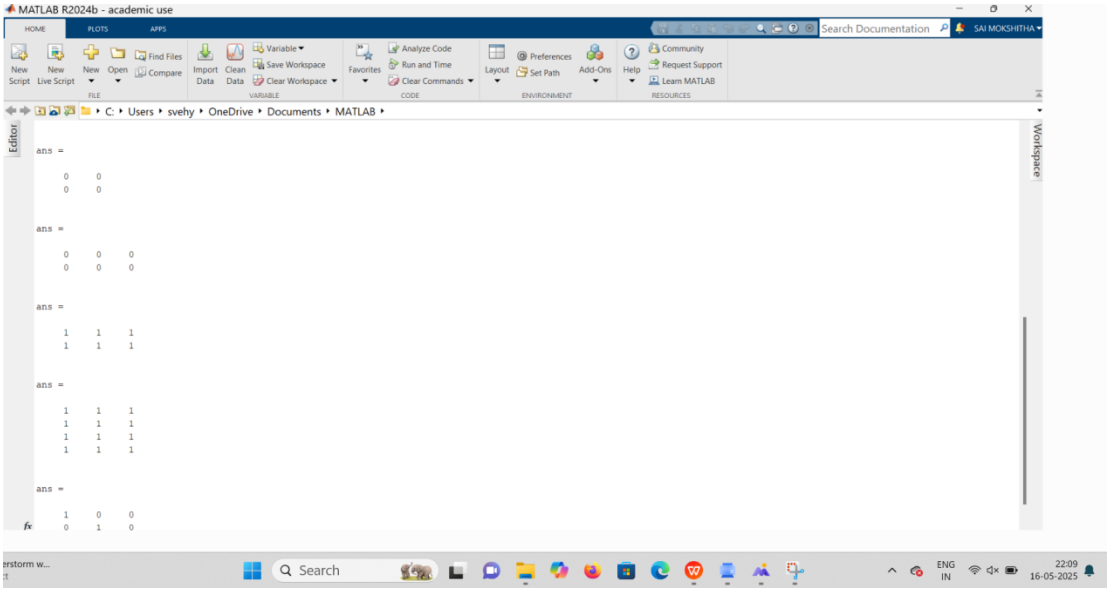
OUTPUT:



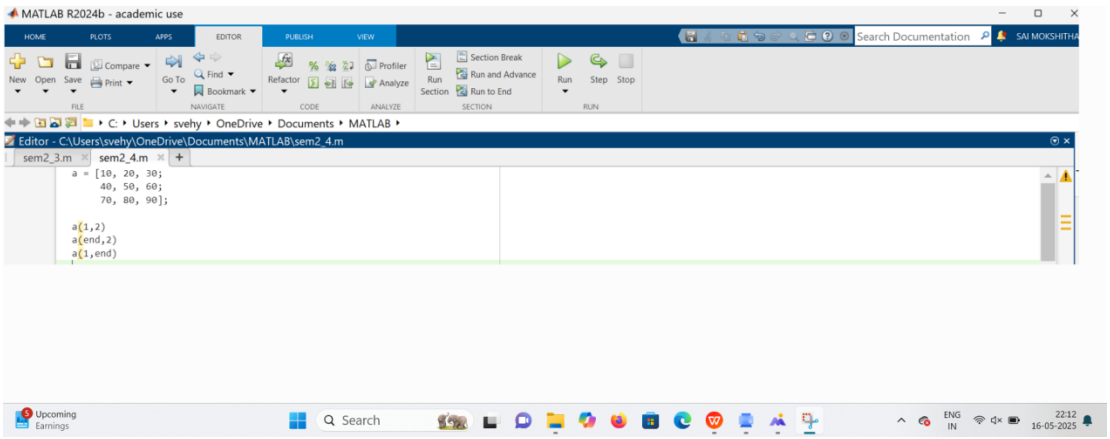
2)



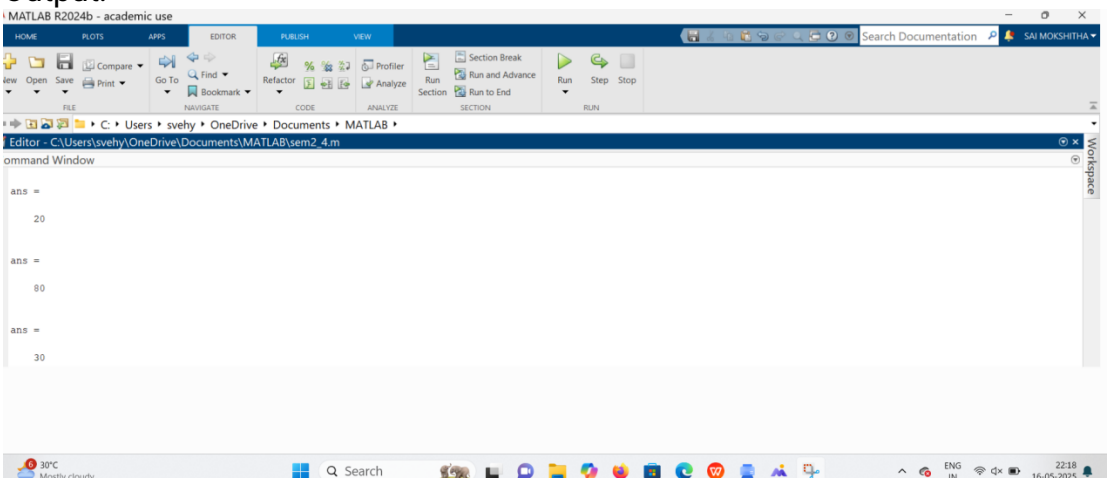
OUTPUT:



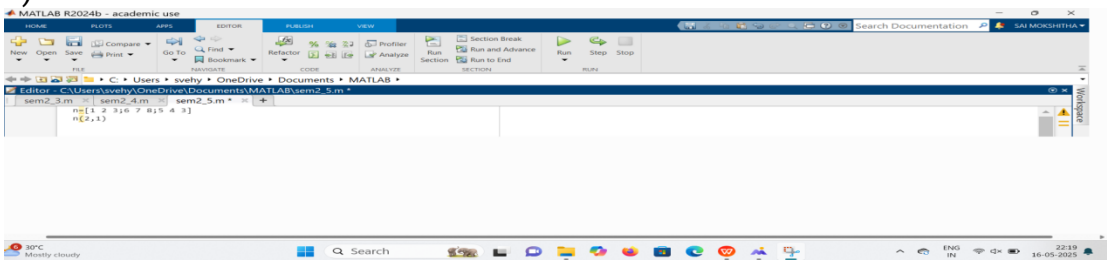
3)



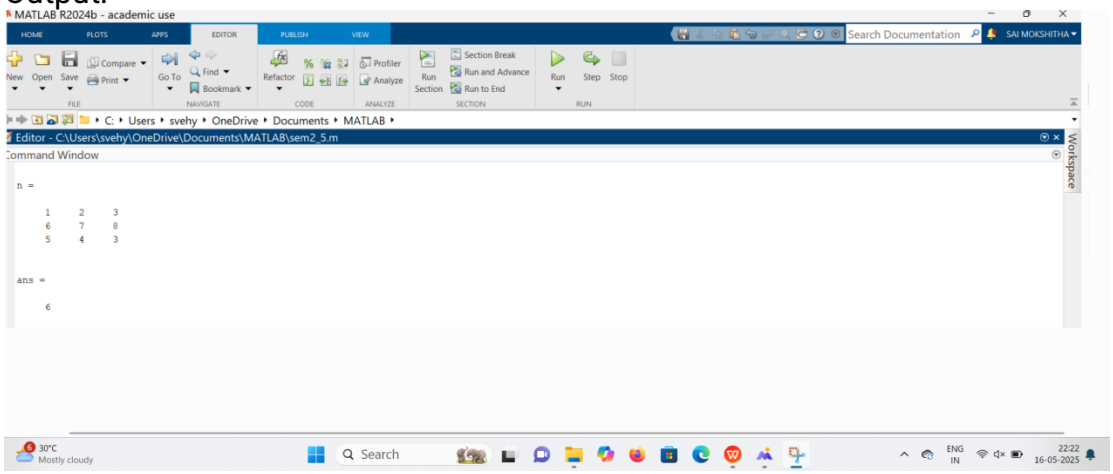
Output:



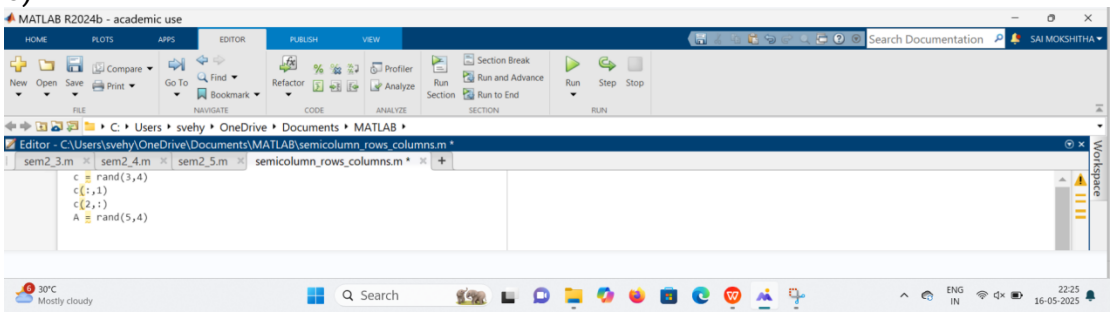
4)



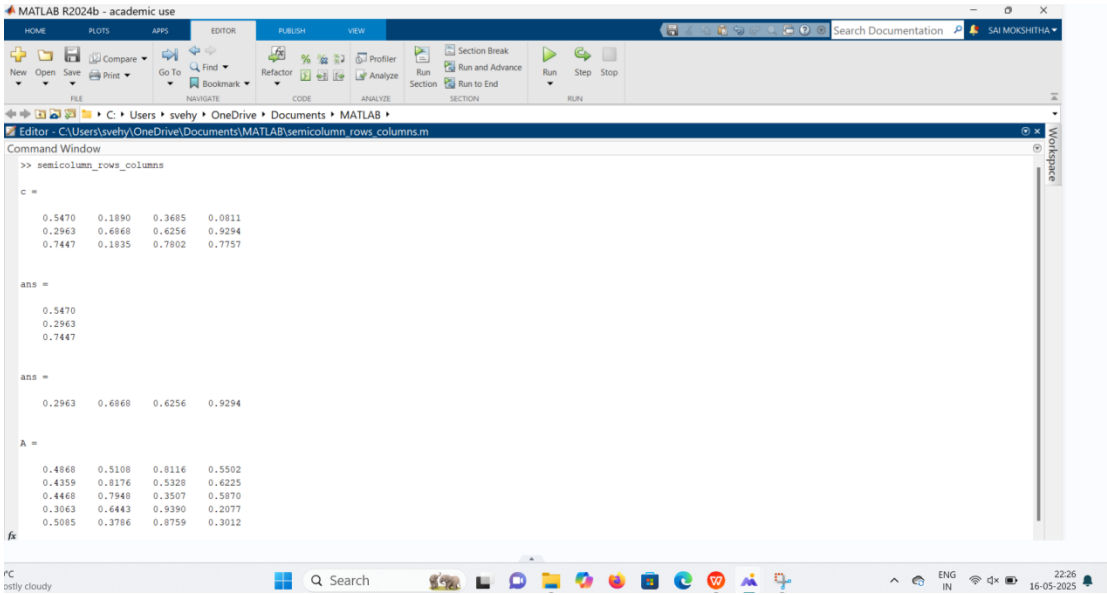
Output:



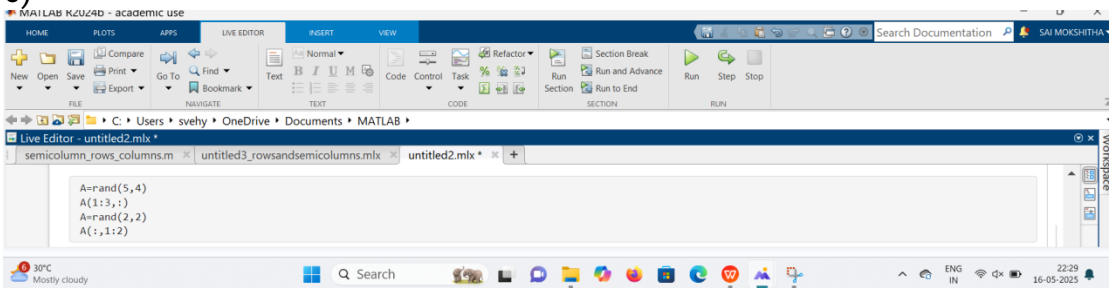
5)



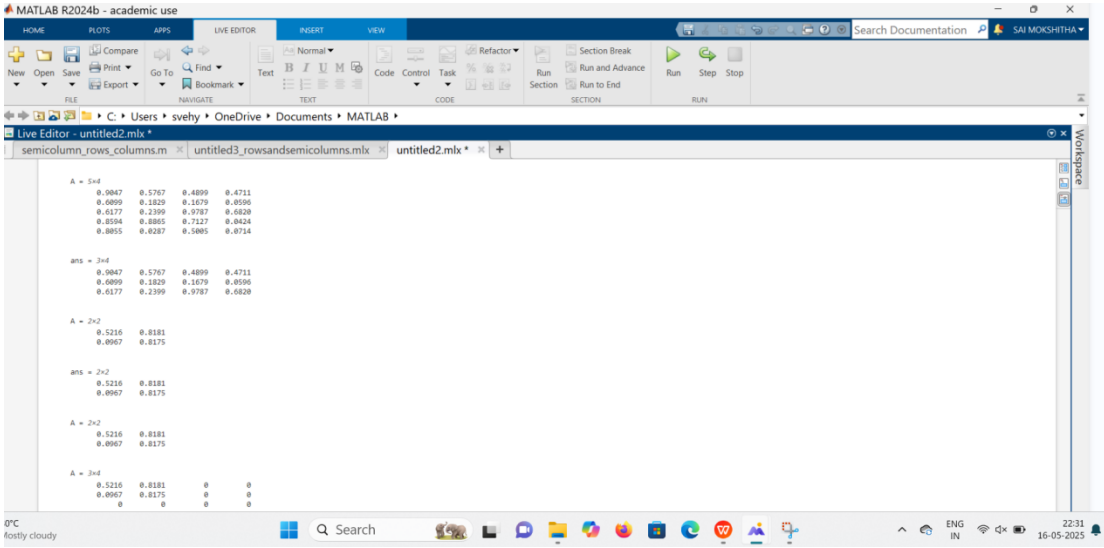
OUTPUT:



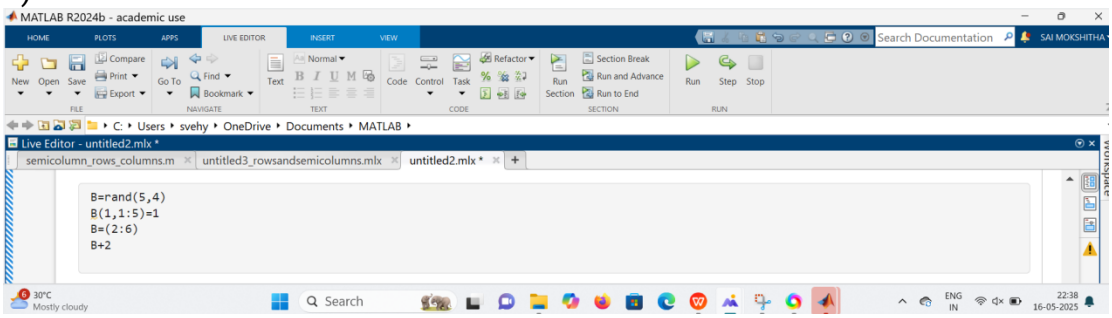
6)



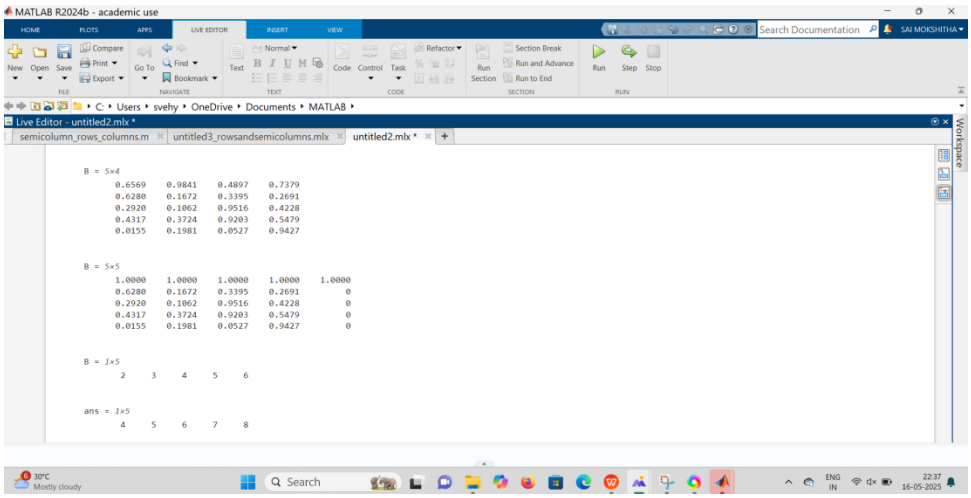
OUTPUT:



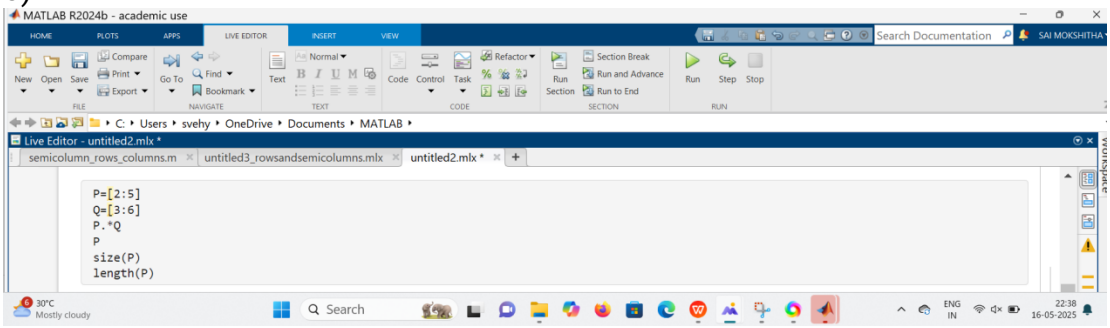
7)



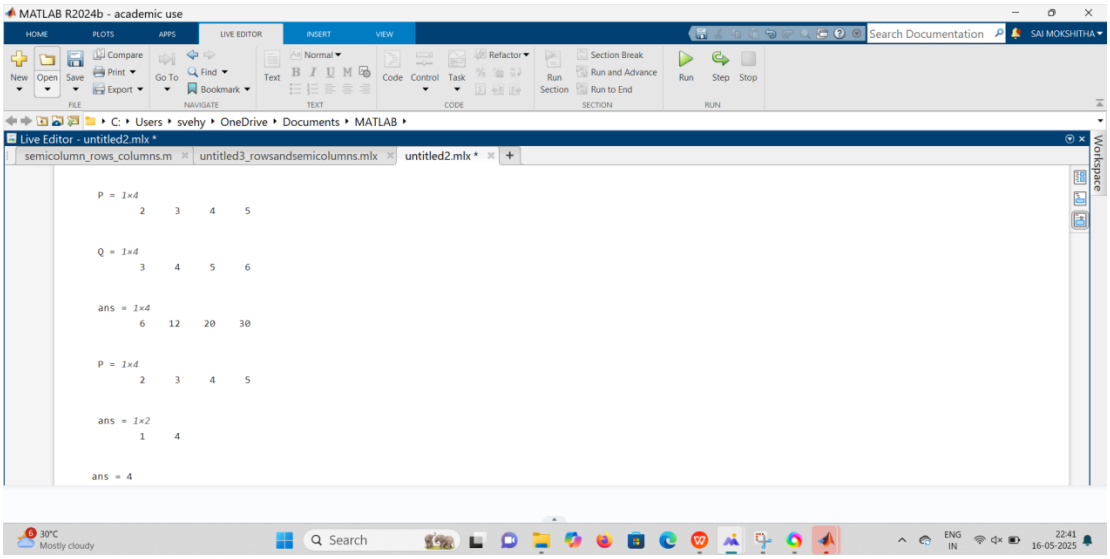
OUTPUT:



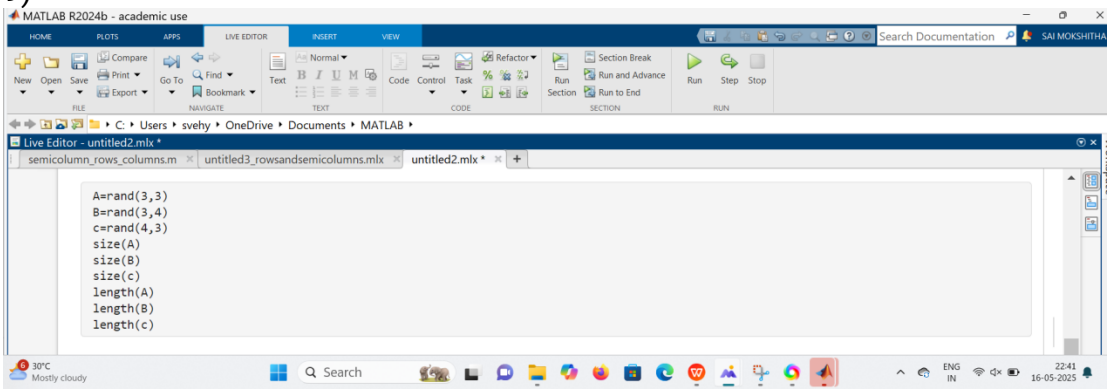
8)



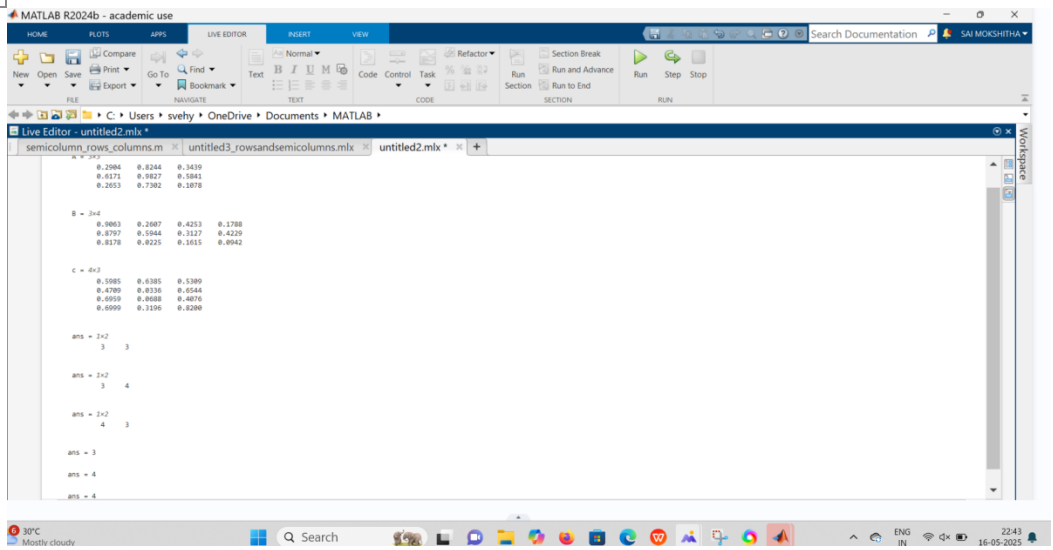
OUTPUT:



9)

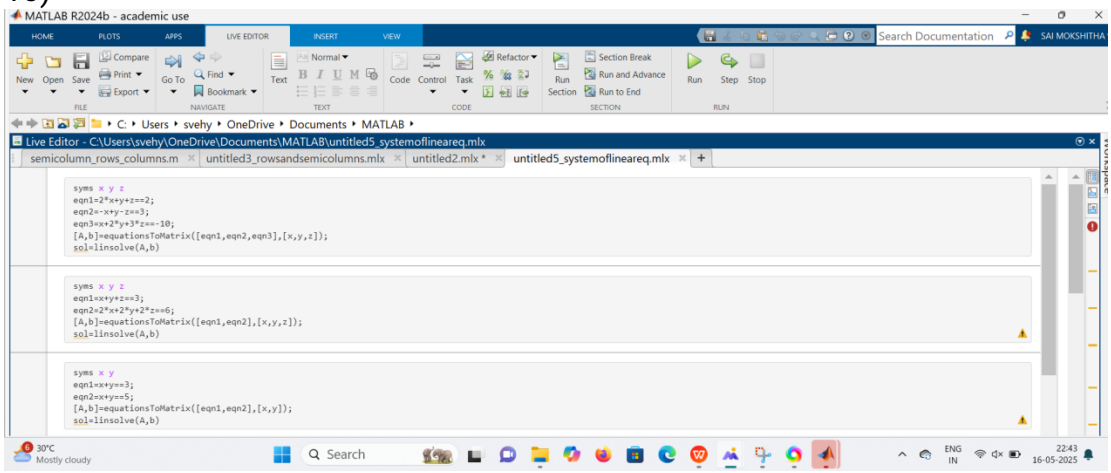


OUTPUT:

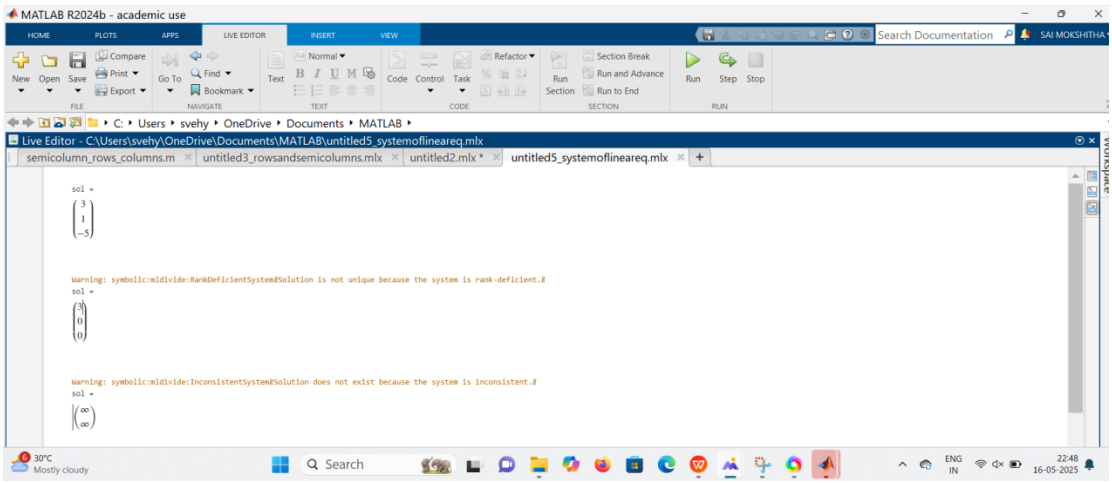


SYSTEM OF LINEAR EQUATIONS:-

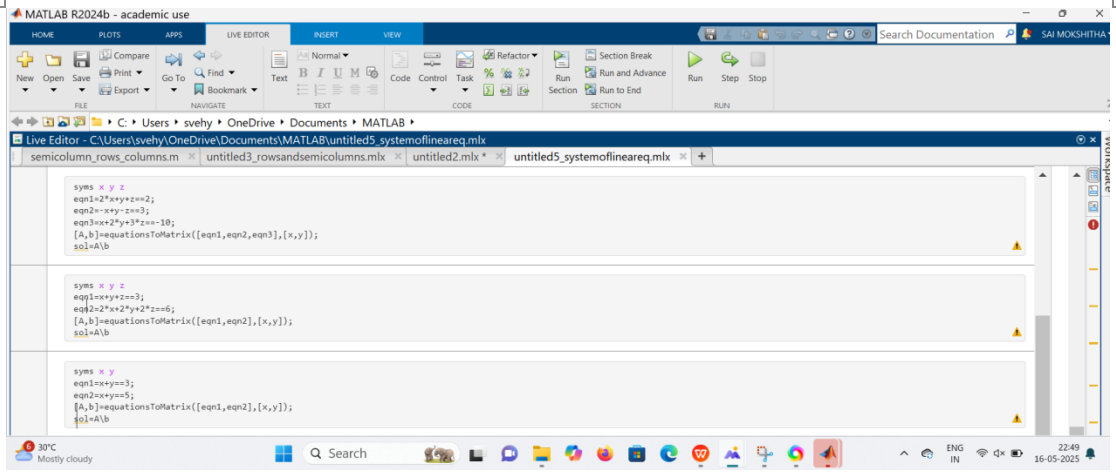
10)



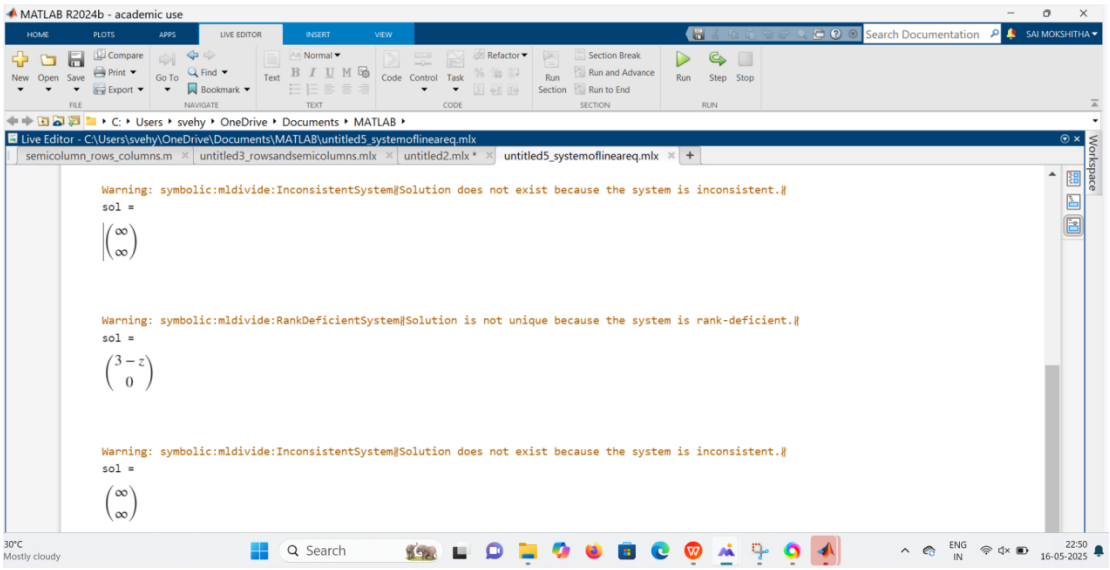
OUTPUT:



11)

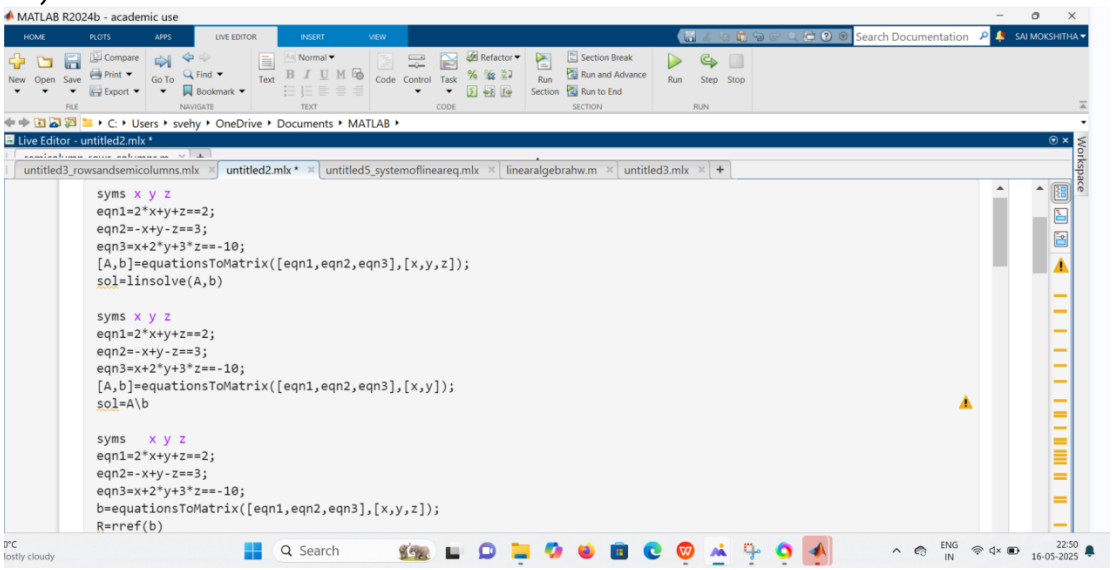


OUTPUT:

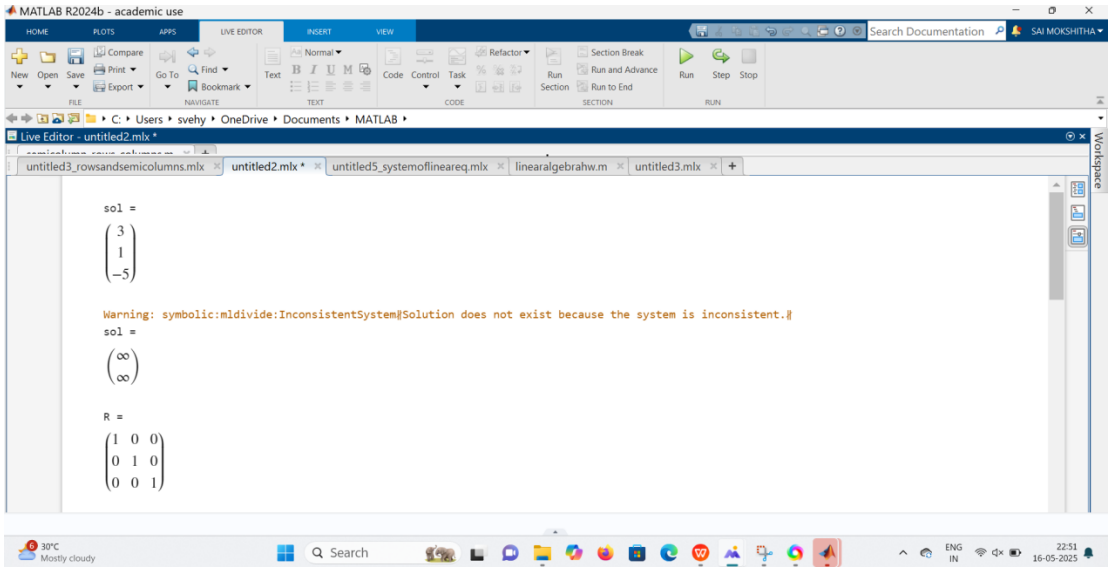


HOMEWORK PROBLEMS:

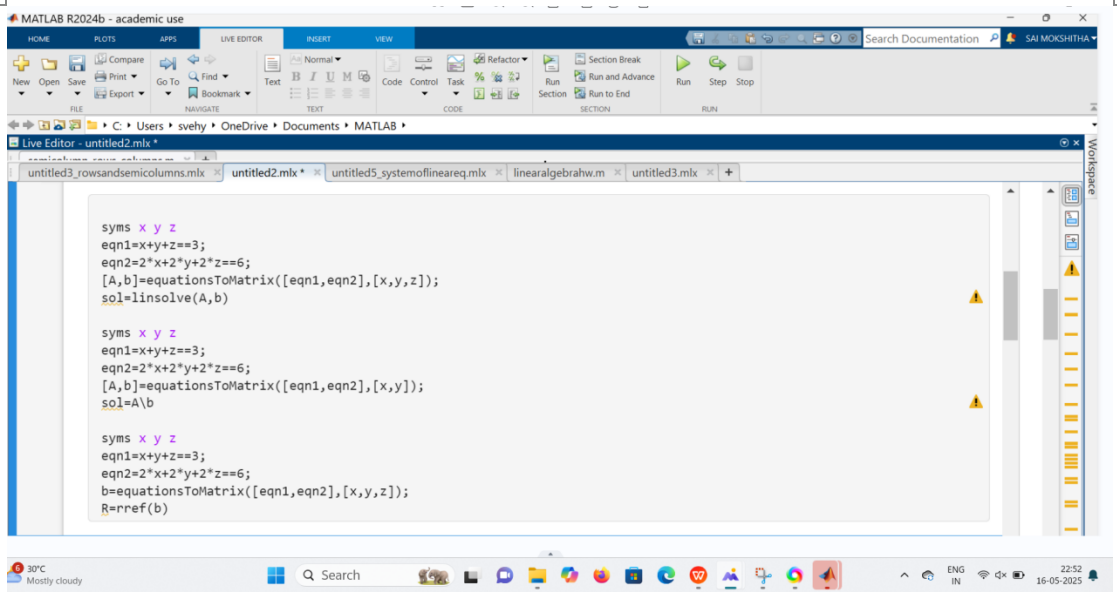
12)


MATLAB R2024b - academic use
Live Editor - untitled2.mlx *
untitled3_rowsandsemicolumns.mlx
untitled2.mlx *
untitled5_systemoflineareq.mlx
linearalgebrahw.m
untitled3.mlx
syms x y z
eqn1=2*x+y+z==2;
eqn2=-x+y-z==3;
eqn3=x+2*y+3*z==-10;
[A,b]=equationsToMatrix([eqn1,eqn2,eqn3],[x,y,z]);
sol=linsolve(A,b)
syms x y z
eqn1=2*x+y+z==2;
eqn2=-x+y-z==3;
eqn3=x+2*y+3*z==-10;
[A,b]=equationsToMatrix([eqn1,eqn2,eqn3],[x,y]);
sol=A\b
syms x y z
eqn1=2*x+y+z==2;
eqn2=-x+y-z==3;
eqn3=x+2*y+3*z==-10;
b=equationsToMatrix([eqn1,eqn2,eqn3],[x,y,z]);
R=rref(b)

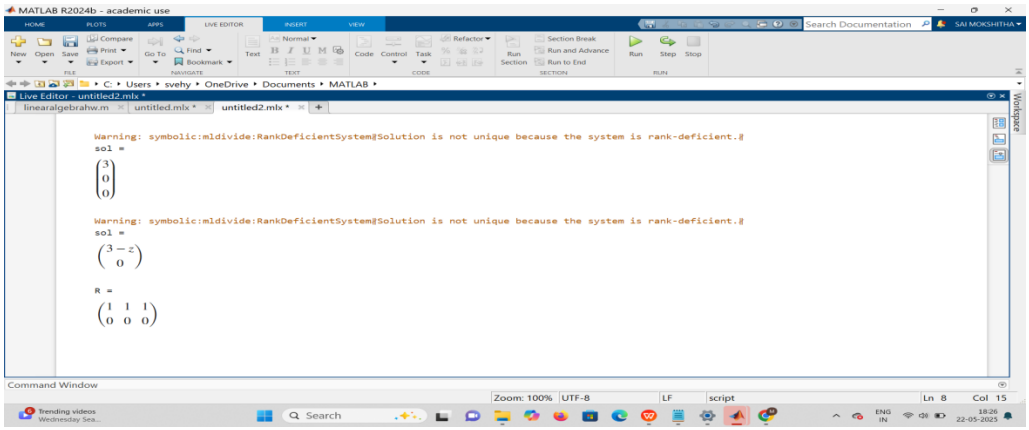
OUTPUT:


MATLAB R2024b - academic use
Live Editor - untitled2.mlx *
sol =
3
1
−5
Warning: symbolic:mldivide:InconsistentSystem Solution does not exist because the system is inconsistent.
sol =
∞
∞
R =
1 0 0
0 1 0
0 0 1

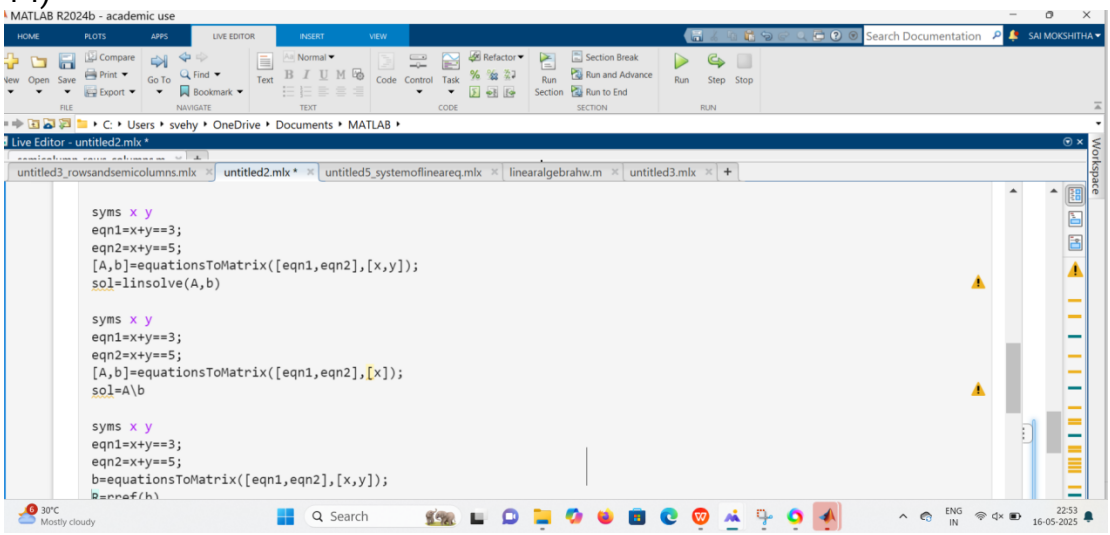
13)



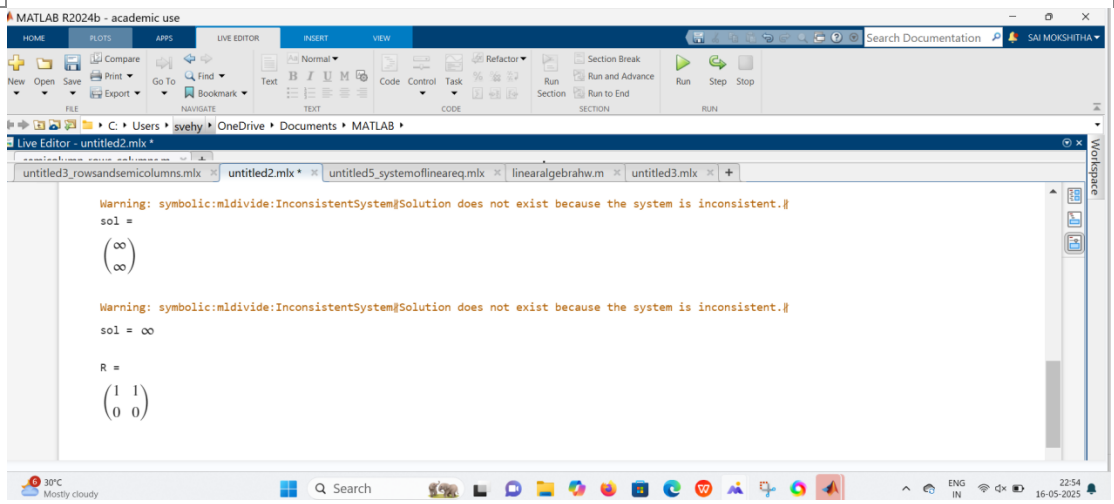
OUTPUT:



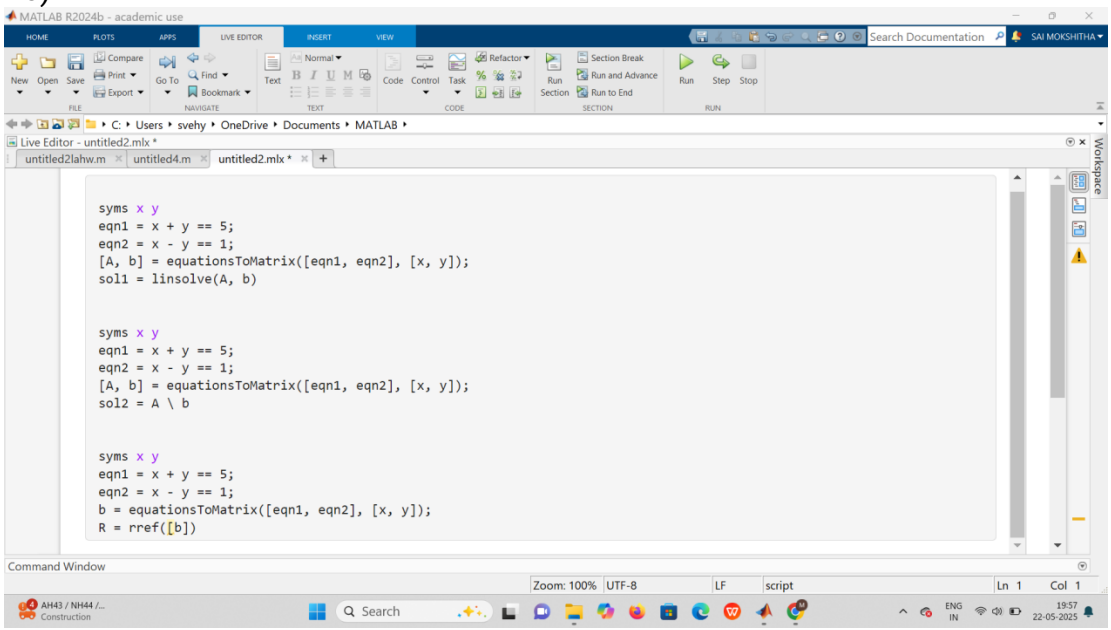
14)



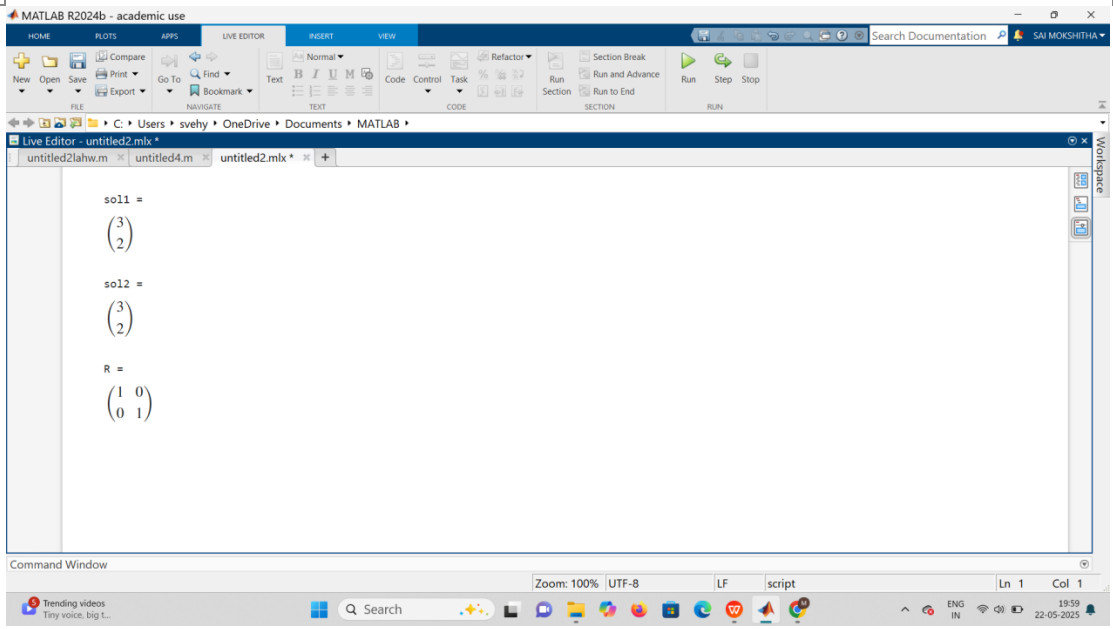
Output:


MATLAB R2024b - academic use
Live Editor - untitled2.mlx *
untitled3_rowsandsemicolumns.mlx
untitled2.mlx *
untitled5_systemoflineareq.mlx
linearalgebrahw.m
untitled3.mlx
Warning: symbolic:mldivide:InconsistentSystem#Solution does not exist because the system is inconsistent.#
sol =
Warning: symbolic:mldivide:InconsistentSystem#Solution does not exist because the system is inconsistent.#
sol = ∞
R =

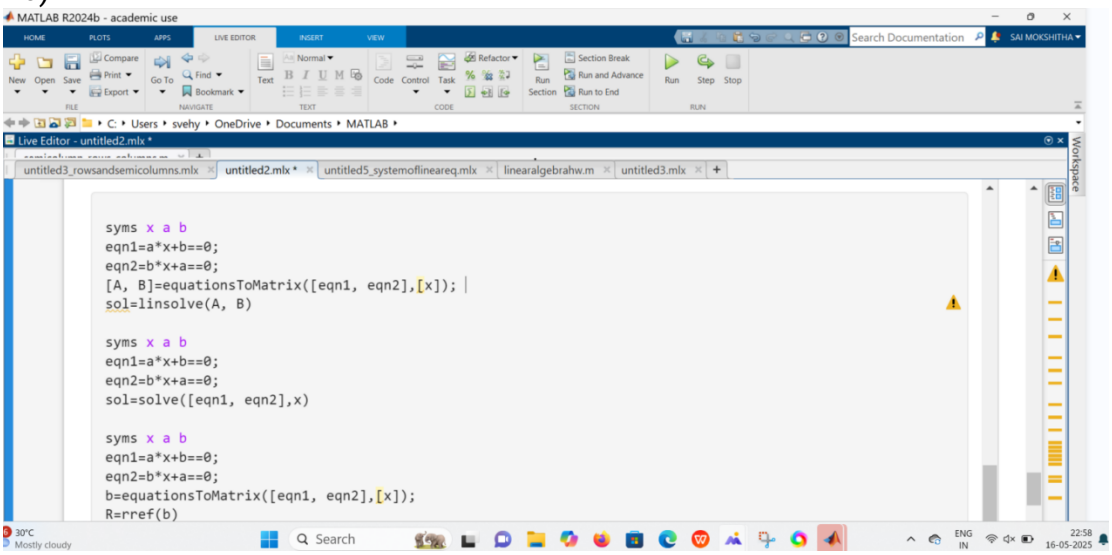
15)


MATLAB R2024b - academic use
Live Editor - untitled2.mlx *
untitled2lahw.m
untitled4.m
untitled2.mlx *
syms x y
eqn1 = x + y == 5;
eqn2 = x - y == 1;
[A, b] = equationsToMatrix([eqn1, eqn2], [x, y]);
sol1 = linsolve(A, b)
syms x y
eqn1 = x + y == 5;
eqn2 = x - y == 1;
[A, b] = equationsToMatrix([eqn1, eqn2], [x, y]);
sol2 = A \ b
syms x y
eqn1 = x + y == 5;
eqn2 = x - y == 1;
b = equationsToMatrix([eqn1, eqn2], [x, y]);
R = rref([b])
Command Window

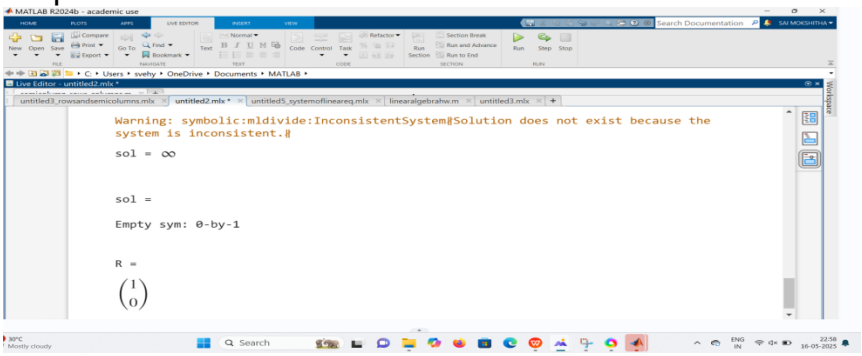
Output:



16)

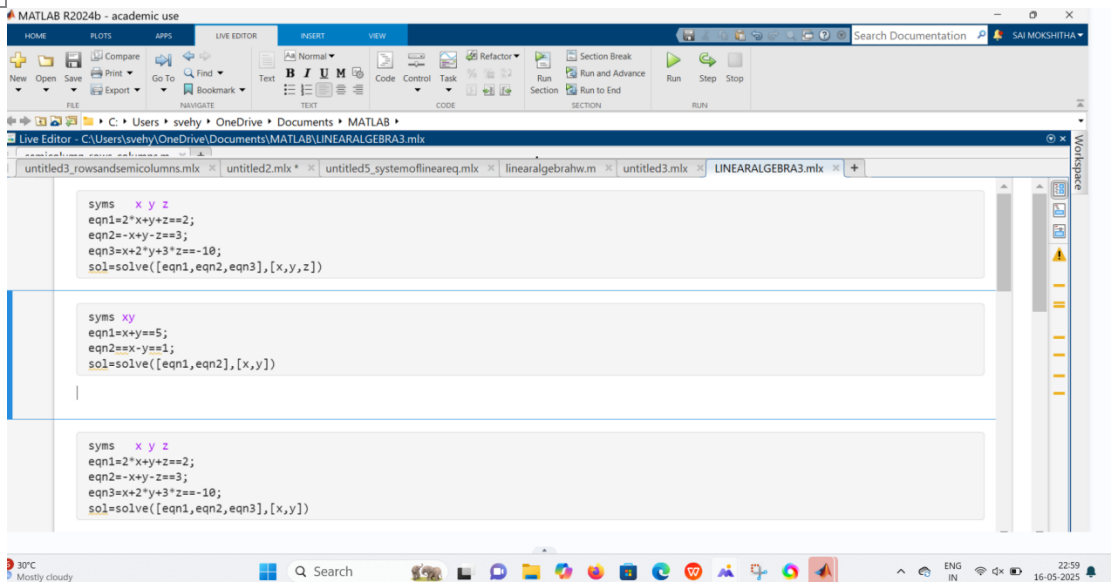


Output:

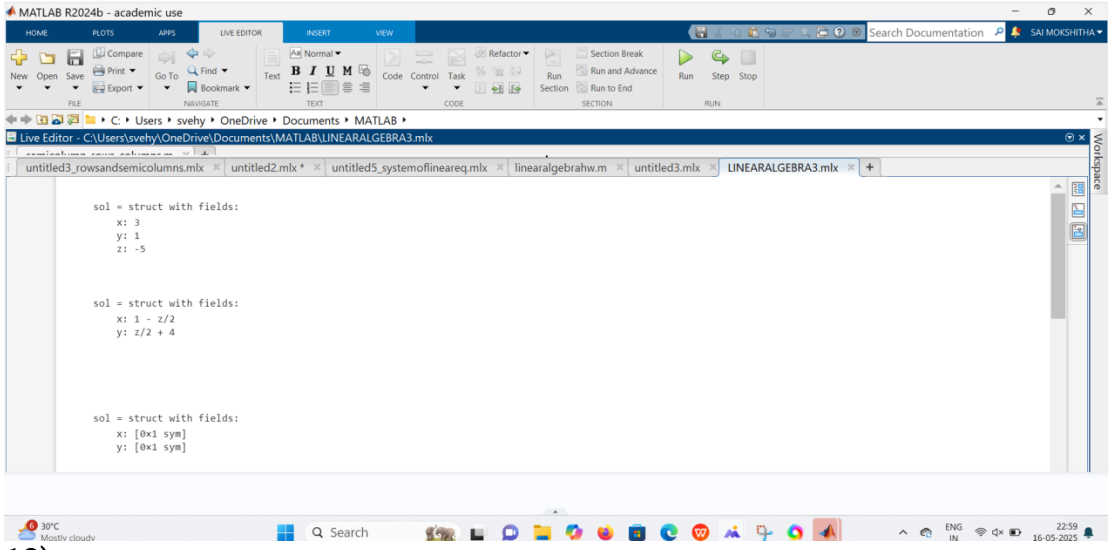


Solving three equations by three variables:

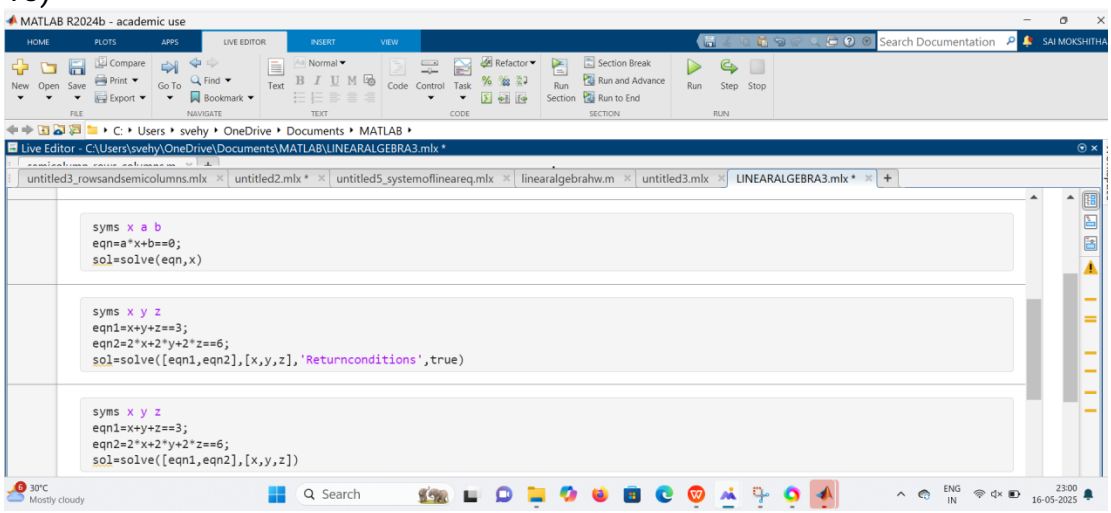
17)



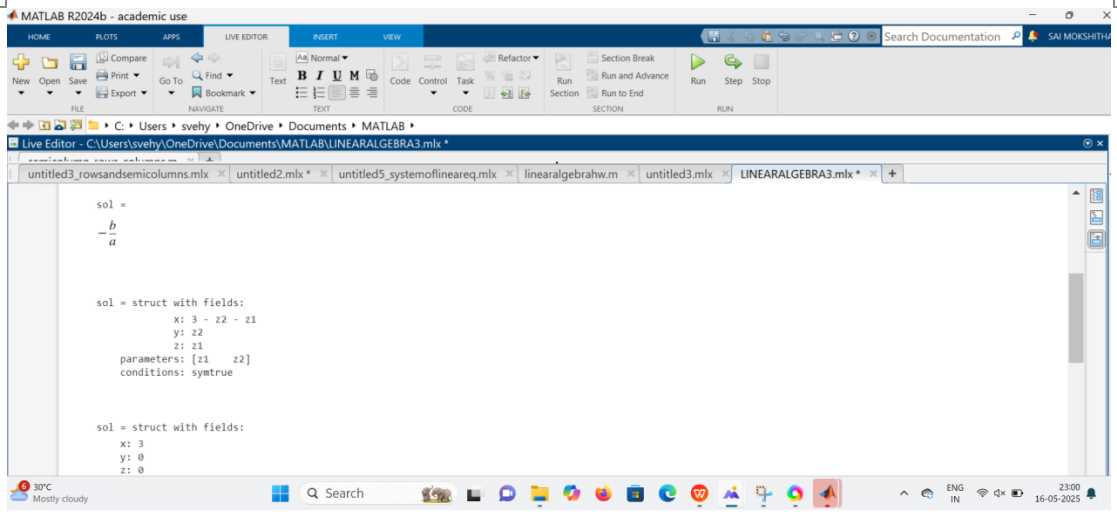
Output:



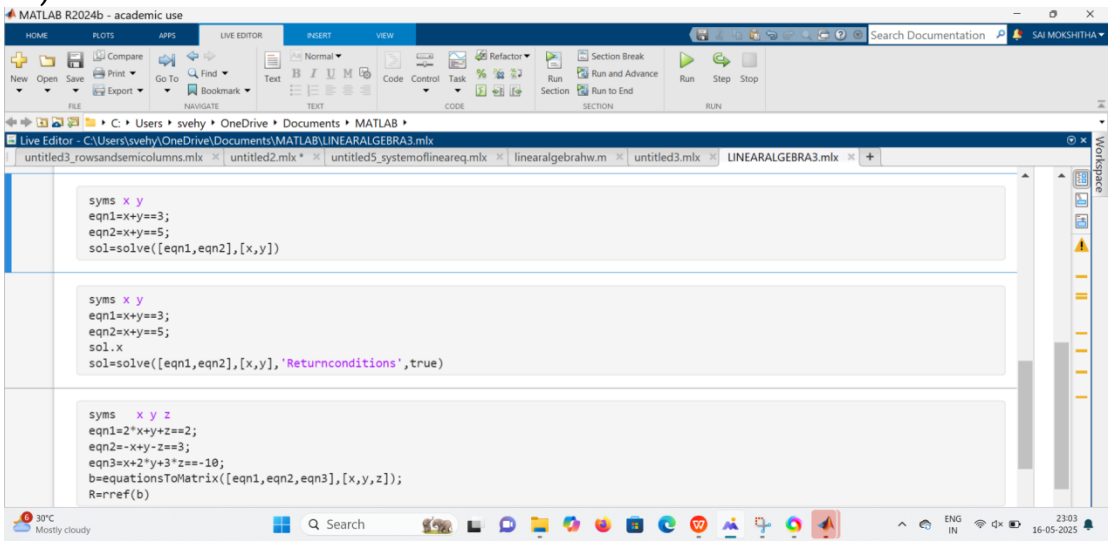
18)



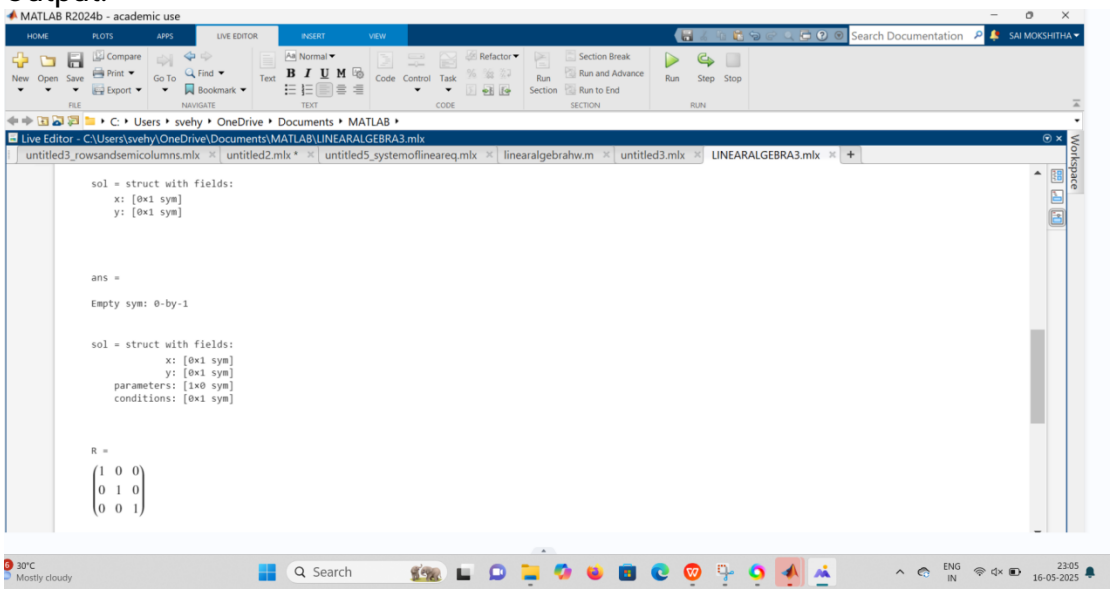
Output:



19)

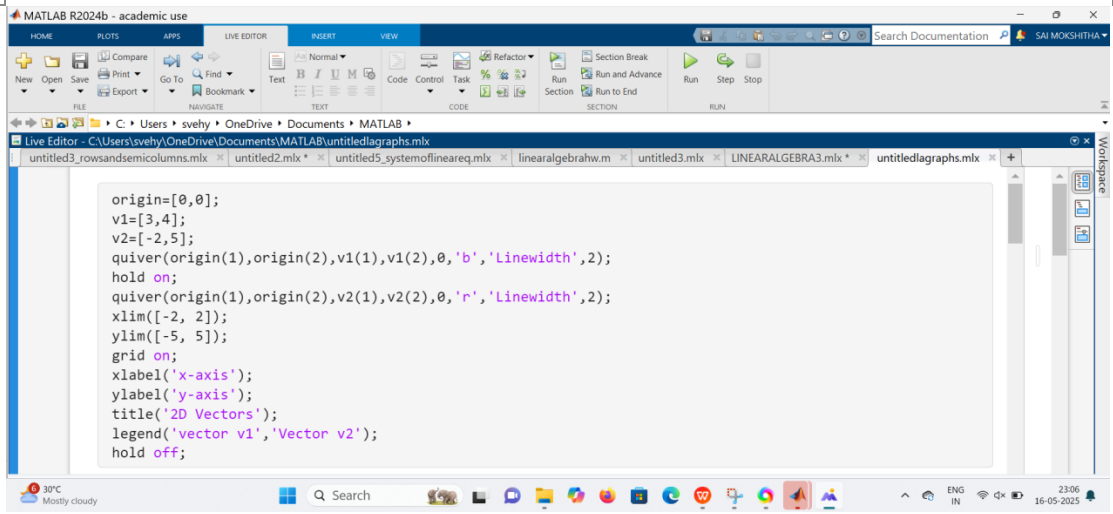


Output:

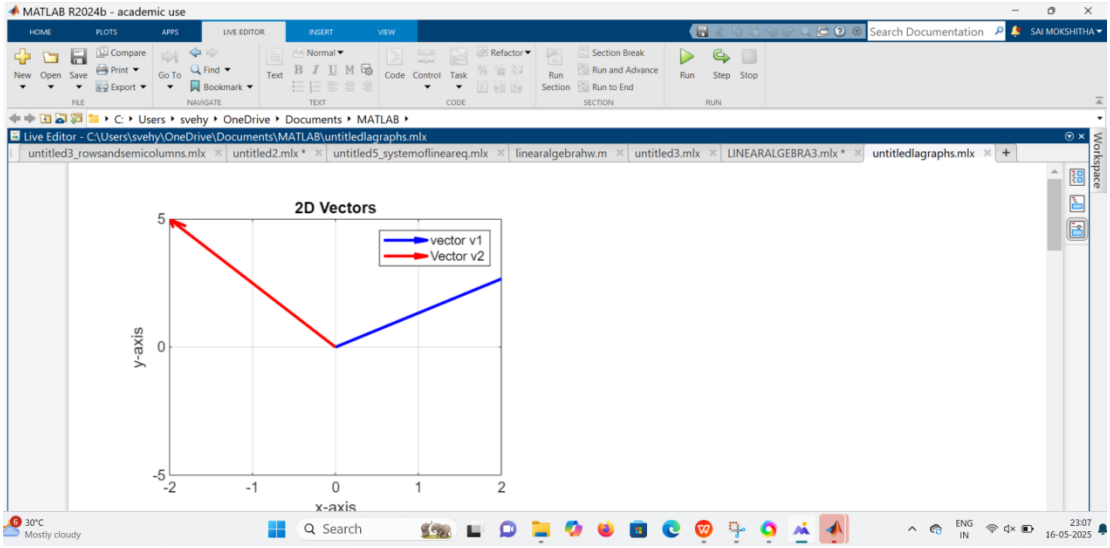


Graphs:

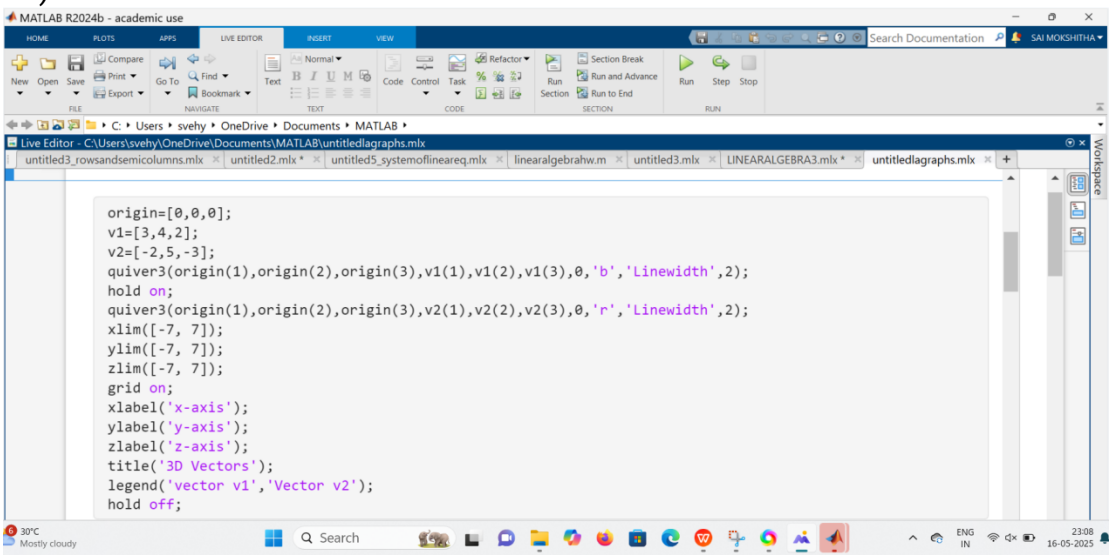
20)



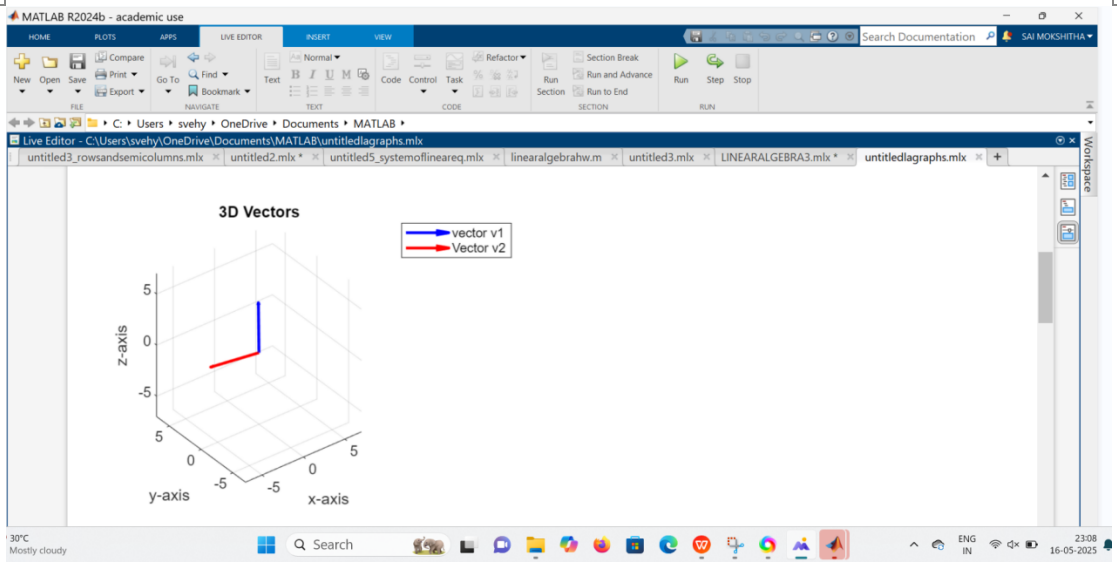
OUTPUT:



21)



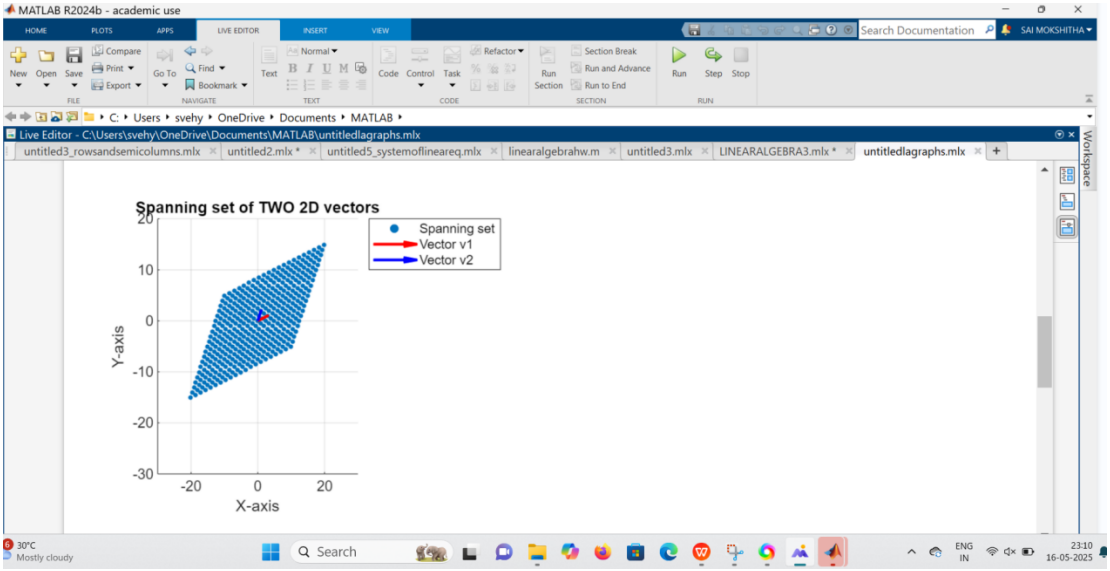
OUTPUT:



22)

```
v1 = [3,1];
v2 = [1,2];
origin = [0,0];
[s,t] = meshgrid(-5:0.5:5);
X=s*v1(1) + t * v2(1);
Y=s*v1(2) + t * v2(2);
scatter(X(:),Y(:),10,'filled','MarkerFaceAlpha',1);
hold on;
quiver(origin(1),origin(2), v1(1),v1(2),0,'r','LineWidth',2);
quiver(origin(1),origin(2), v2(1),v2(2),0,'b','LineWidth',2);
xlim([-30,30]);
ylim([-30,20]);
grid on;
xlabel('X-axis');
ylabel('Y-axis');
title('Spanning set of TWO 2D vectors');
legend('Spanning set','Vector v1','Vector v2');
hold off;
```

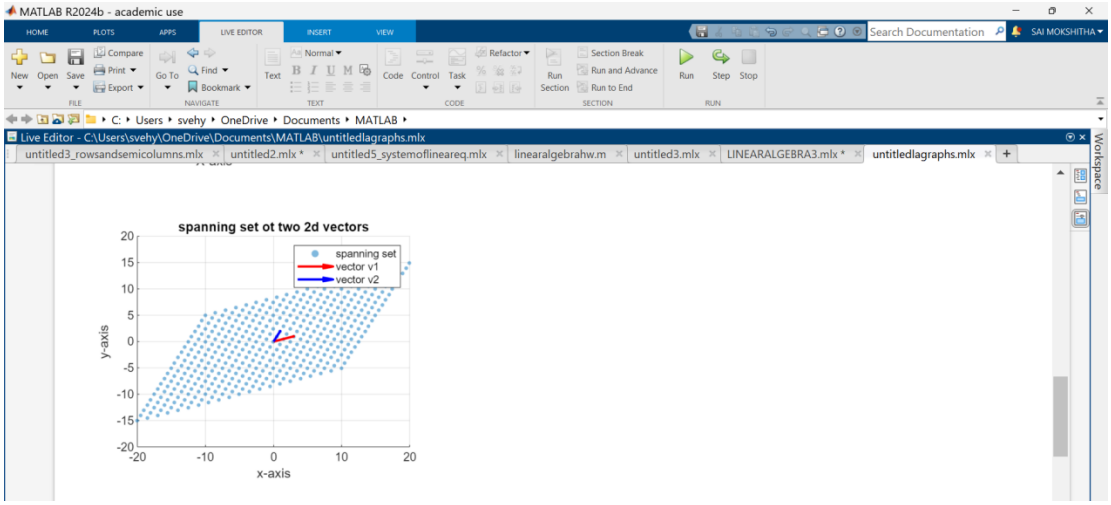
OUTPUT:



23)

```
%span(k1v1+k2v2=v)
%define vectors
v1=[3,1];
v2=[1,2];
origin=[0,0];
%grid of scalar coefficients for linear combination
[s,t]=meshgrid(-5:0.5:5);
%spanning set
X=s*v1(1)+t*v2(1);
Y=s*v1(2)+t*v2(2);
%plot the spanning set
%figure
scatter(X(:),Y(:),10,'filled','MarkerFaceAlpha',0.5);
hold on;
%plot original vector
quiver(origin(1),origin(2),v1(1),v1(2),0,'r','LineWidth',2);
quiver(origin(1),origin(2),v2(1),v2(2),0,'b','LineWidth',2);
%set axis limits
xlim([-20,20]);
ylim([-20,20]);
%labeling
grid on;
xlabel('x-axis');
ylabel('y-axis');
title('spanning set ot two 2d vectors');
legend('spanning set','vector v1','vector v2');
hold off;
```

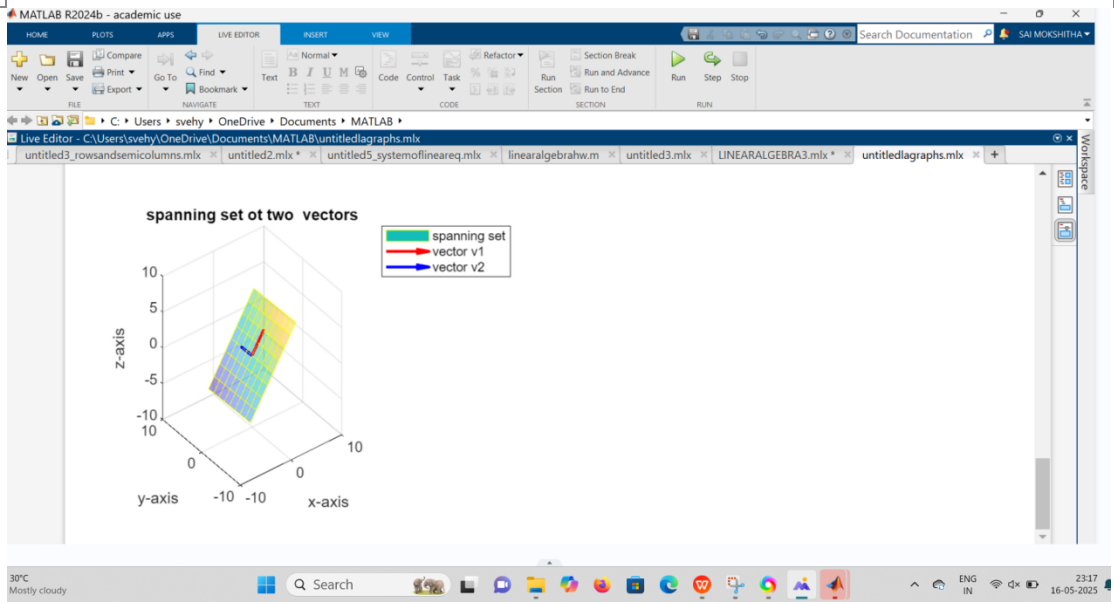
OUTPUT:



24)

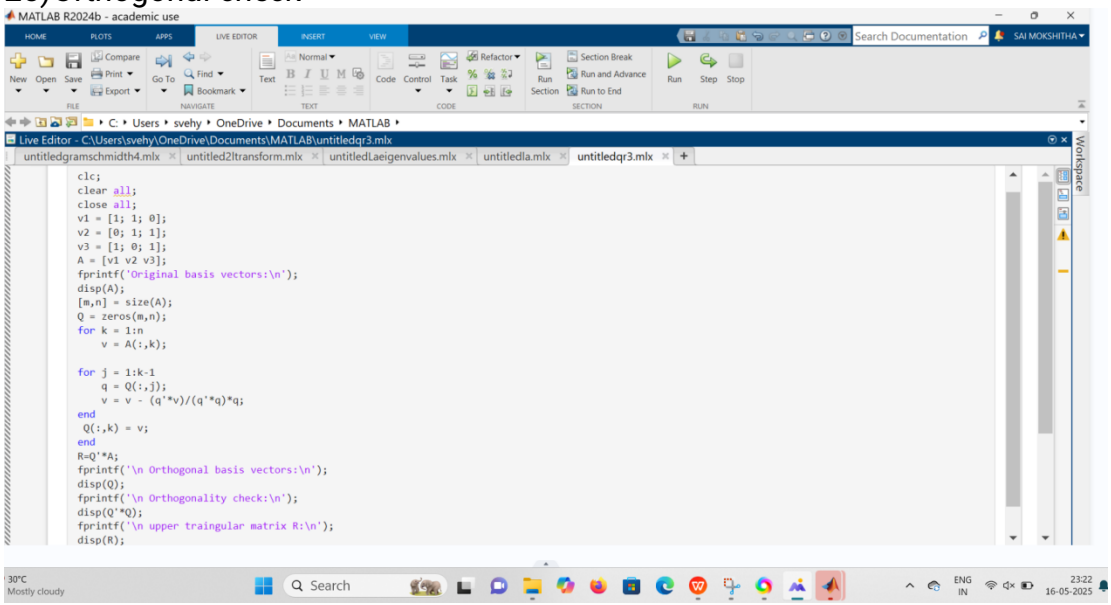
```
v1=[3,1,2];
v2=[1,4,-1];
origin=[0,0,0];
[s,t]=meshgrid(-2:0.5:2);
%spanning set
X=s*v1(1)+t*v2(1);
Y=s*v1(2)+t*v2(2);
Z=s*v1(3)+t*v2(3);
%plot the spanning set
figure;
surf(X,Y,Z,'FaceAlpha',0.5,'Edgecolor','yellow');
hold on;
%plot original vector
quiver3(origin(1),origin(2),origin(3),v1(1),v1(2),v1(3),0,'r','LineWidth',2);
quiver3(origin(1),origin(2),origin(3),v2(1),v2(2),v2(3),0,'b','LineWidth',2);

%set axis limits
xlim([-10,10]);
ylim([-10,10]);
zlim([-10,10]);
%labeling
grid on;
xlabel('x-axis');
ylabel('y-axis');
zlabel('z-axis');
title('spanning set ot two  vectors');
legend('spanning set','vector v1','vector v2');
hold off;
```

OUTPUT:

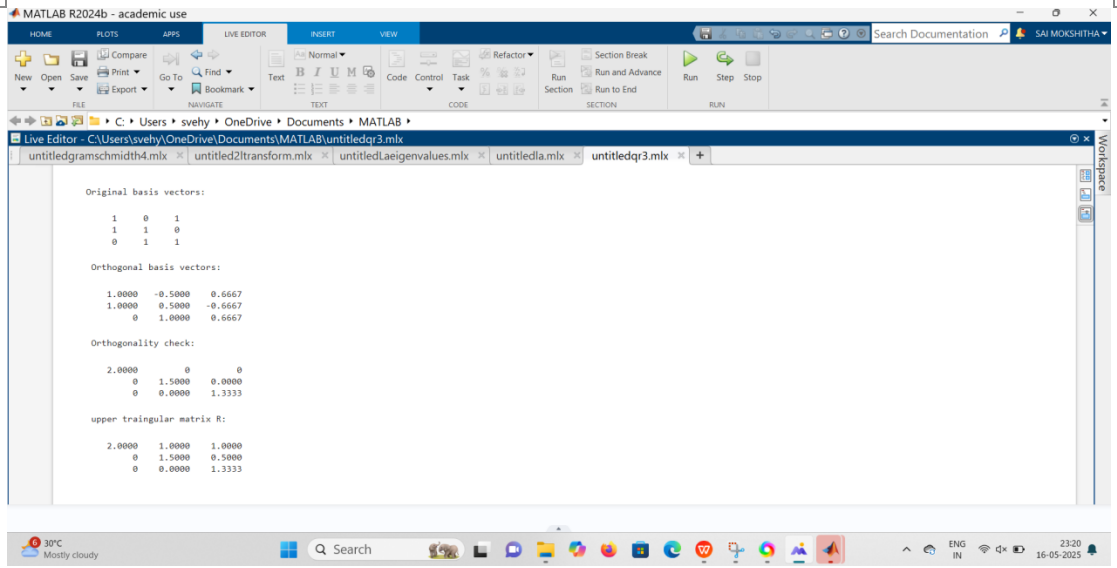


Gram-Schmidt Problems:

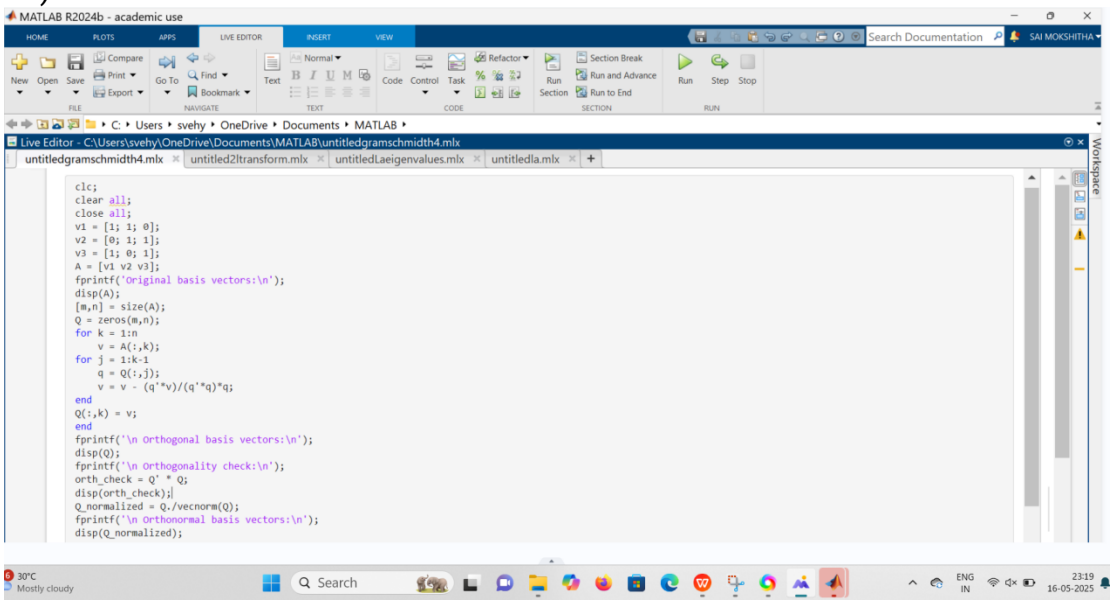
25)Orthogonal check



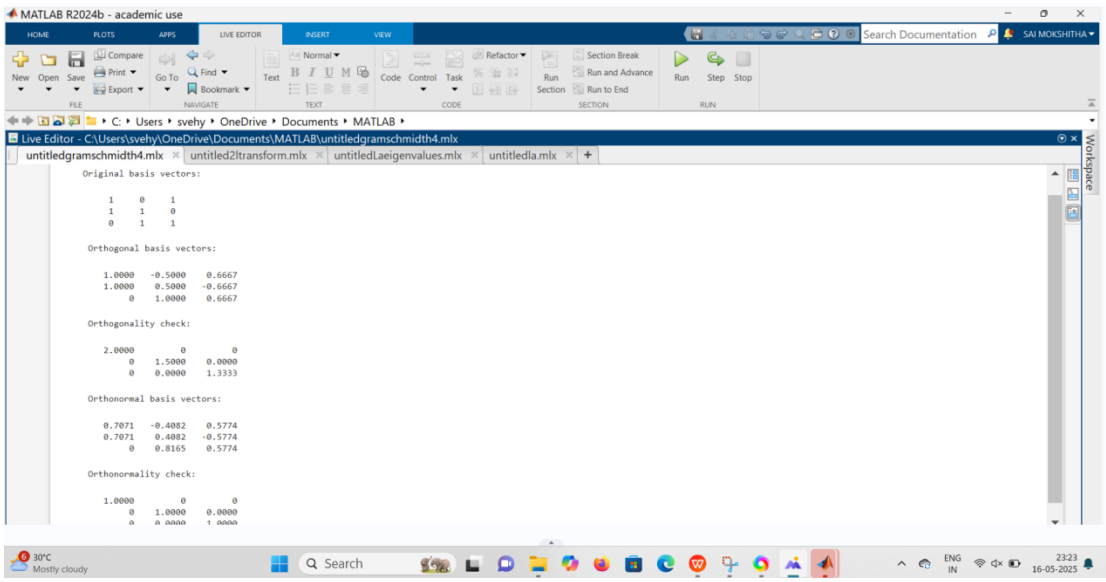
Output:



26)Orthonormal check



OUTPUT:

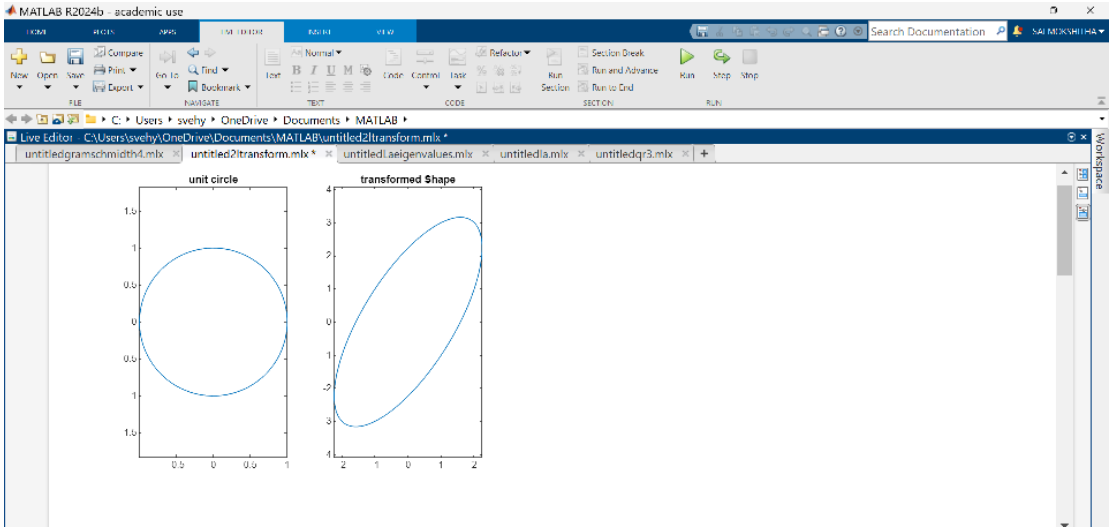


Transformation

27)

```
A=[1 2;3 1];
theta=linspace(0,2*pi,100);
x=cos(theta);
y=sin(theta);
circle=[x;y];
transformed=A*circle;
figure;
subplot(1,2,1);
plot(circle(1,:),circle(2,:));
title('unit circle');
axis equal;
subplot(1,2,2);
plot(transformed(1,:),transformed(2,:));
title('transformed Shape');
axis equal;
```

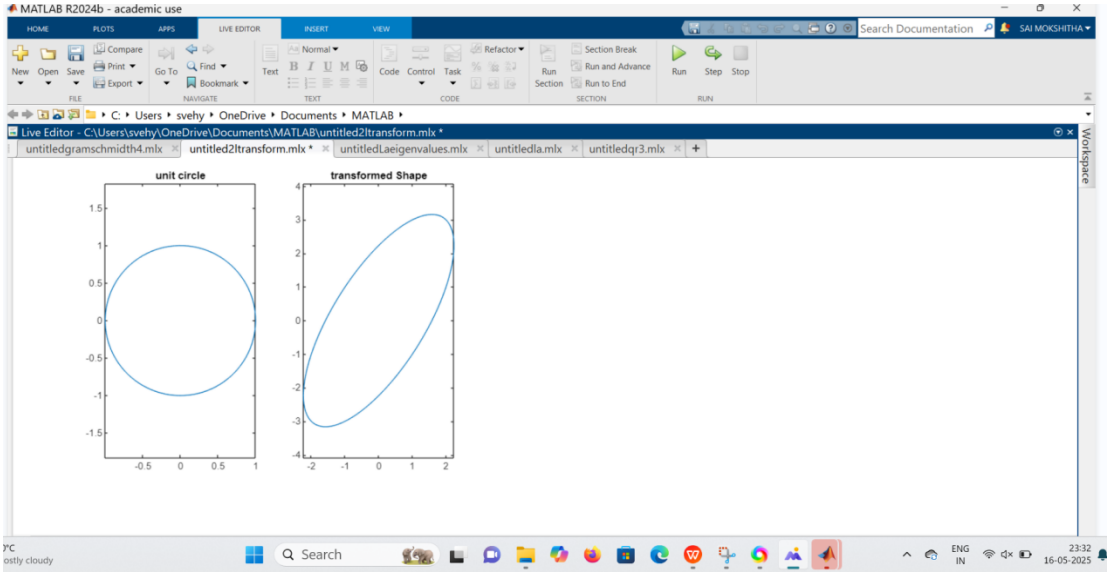
OUTPUT:



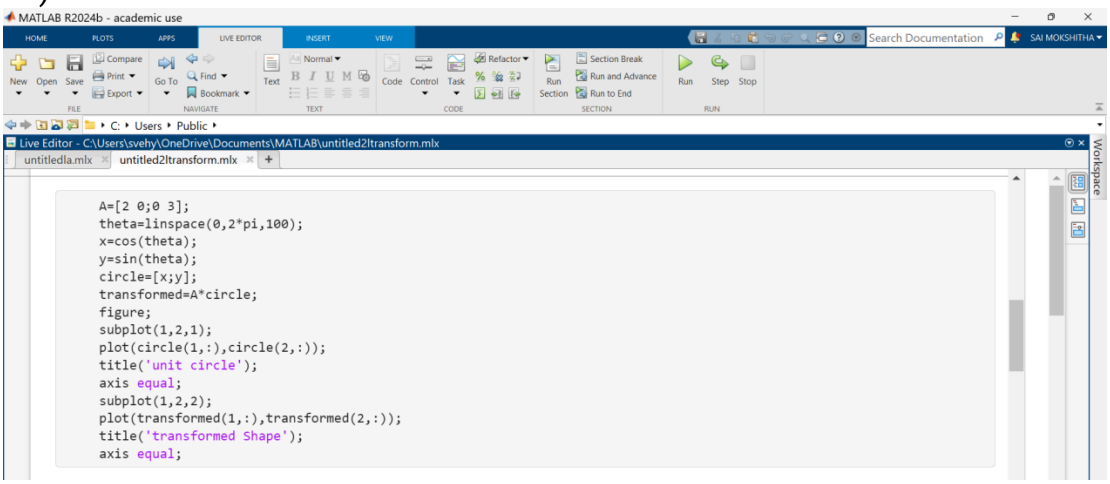
28)

```
A=[2 0;0 2];
theta=linspace(0,2*pi,100);
x=cos(theta);
y=sin(theta);
circle=[x;y];
transformed=A*circle;
figure;
subplot(1,2,1);
plot(circle(1,:),circle(2,:));
title('unit circle');
axis equal;
subplot(1,2,2);
plot(transformed(1,:),transformed(2,:));
title('transformed Shape');
axis equal;
```

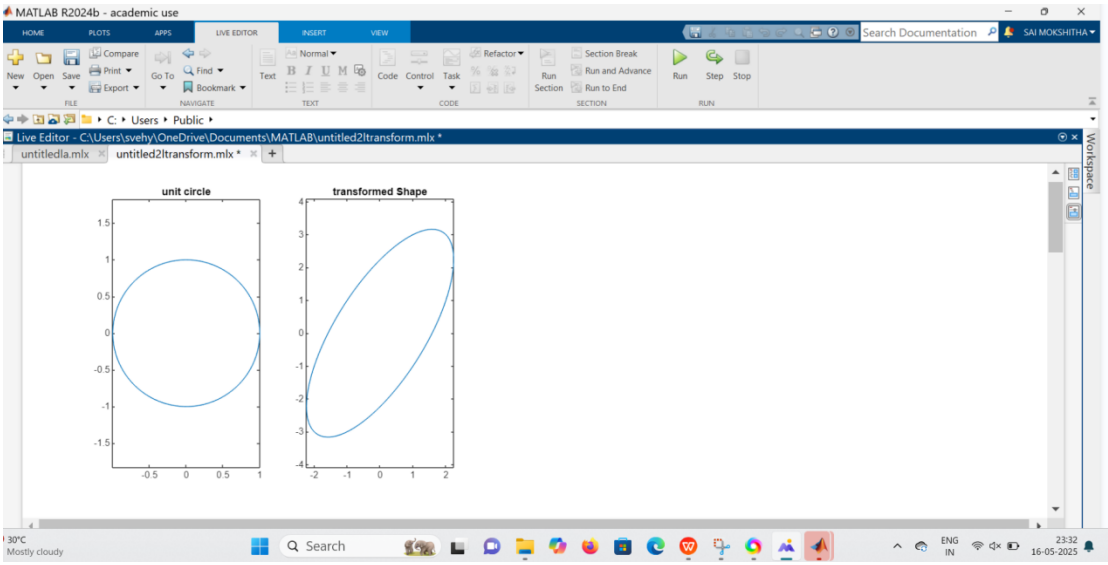
OUTPUT:


MATLAB R2024b - academic use
unit circle
transformed Shape

29)


MATLAB R2024b - academic use
A=[2 0;0 3];
theta=linspace(0,2*pi,100);
x=cos(theta);
y=sin(theta);
circle=[x;y];
transformed=A*circle;
figure;
subplot(1,2,1);
plot(circle(1,:),circle(2,:));
title('unit circle');
axis equal;
subplot(1,2,2);
plot(transformed(1,:),transformed(2,:));
title('transformed Shape');
axis equal;

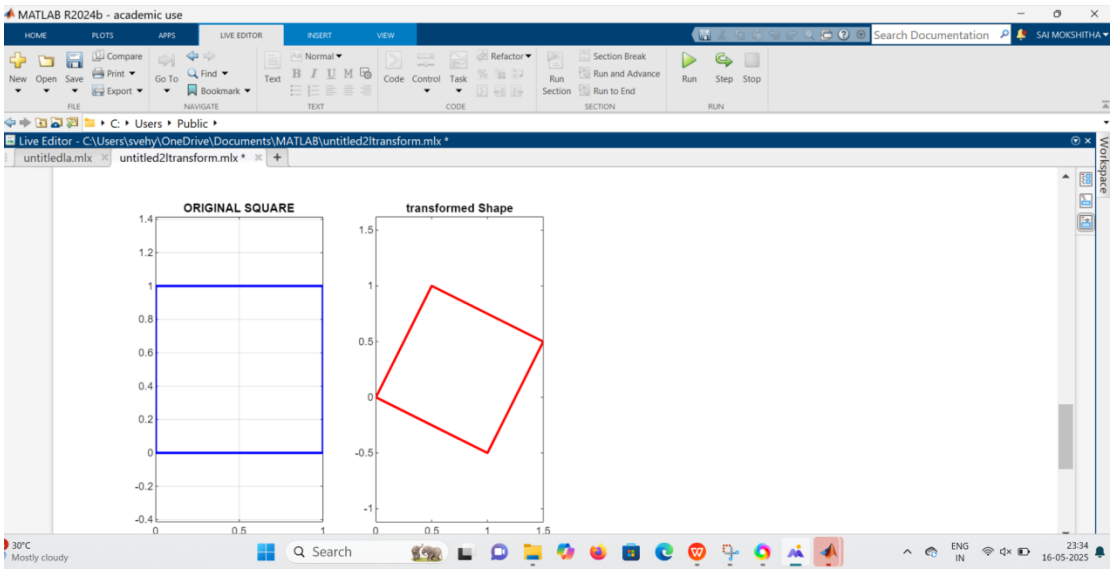
OUTPUT:


MATLAB R2024b - academic use
unit circle
transformed Shape

30)

```
A=[1 0.5;-0.5 1];
    square=[0 1 1 0 0;0 0 1 1 0];
    transformed=A*square;
    figure;
    subplot(1,2,1);
    plot(square(1,:),square(2,:),'b-','Linewidth',2);
    title('ORIGINAL SQUARE');
    axis equal;
    grid on;

    subplot(1,2,2);
    plot(transformed(1,:),transformed(2,:),'r-','linewidth',2);
    title('transformed Shape');
    axis equal;
```

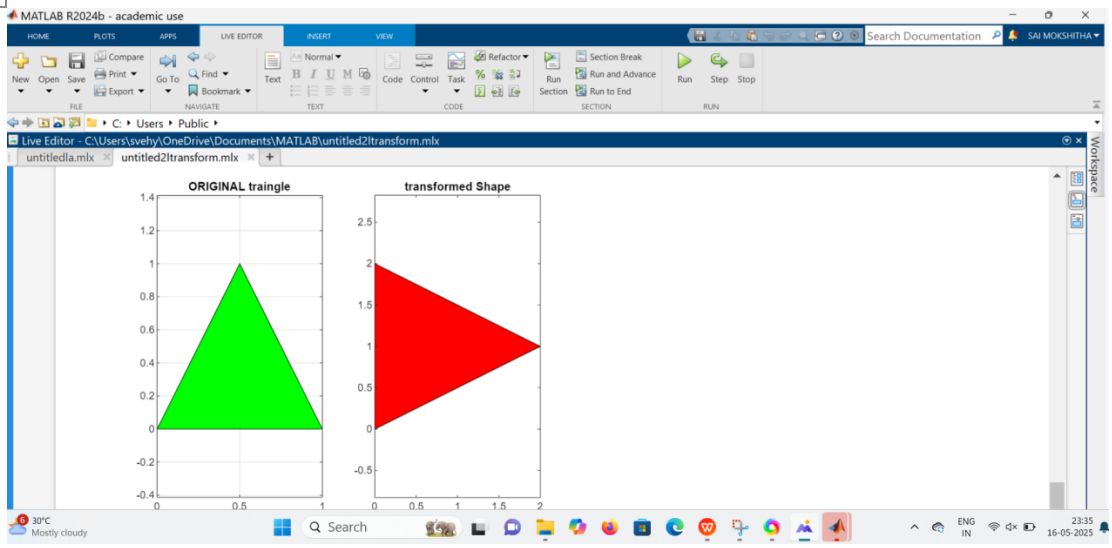
OUTPUT:



31)

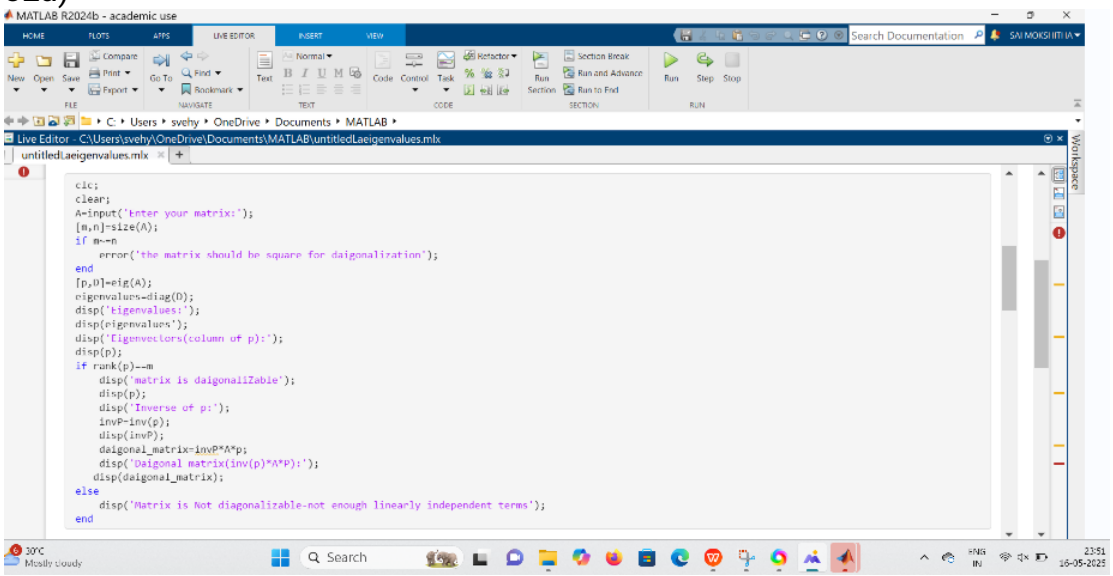
```
A=[0 2;2 0];
    triangle=[0 1 0.5 0;0 0 1 0];
    transformed=A*triangle;
    figure;
    subplot(1,2,1);
    fill(triangle(1,:),triangle(2,:),'g-');
    title('ORIGINAL traingle');
    axis equal;
    grid on;

    subplot(1,2,2);
    fill(transformed(1,:),transformed(2,:),'r-');
    title('transformed Shape');
    axis equal;
```

OUTPUT:



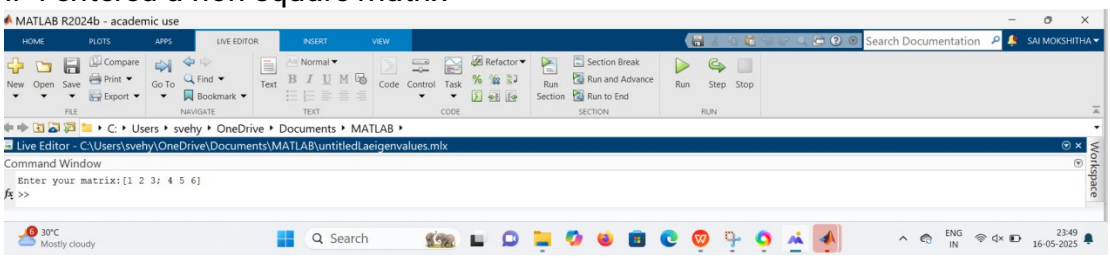
EIGEN VALUES:

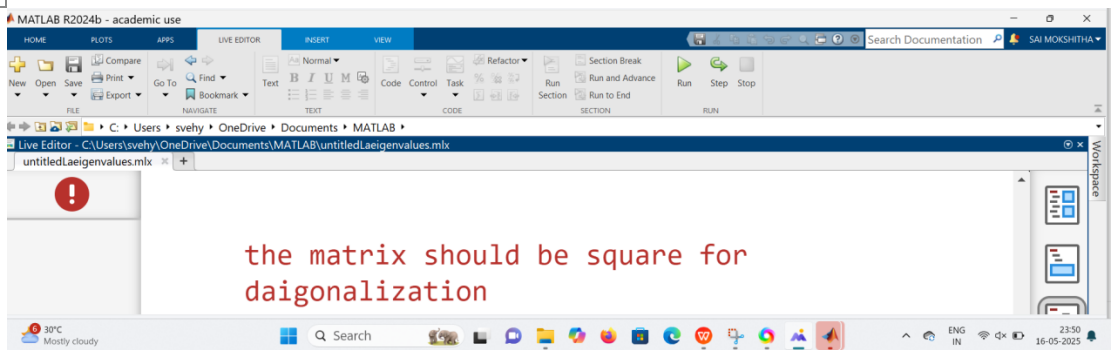
32a)



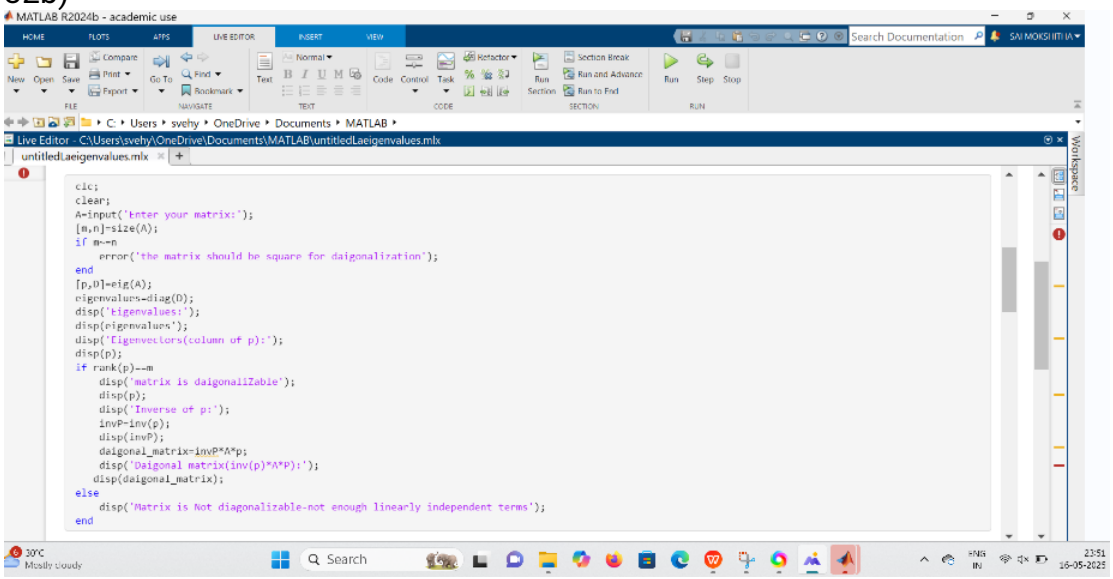
OUTPUT:

IF I entered a non square matrix



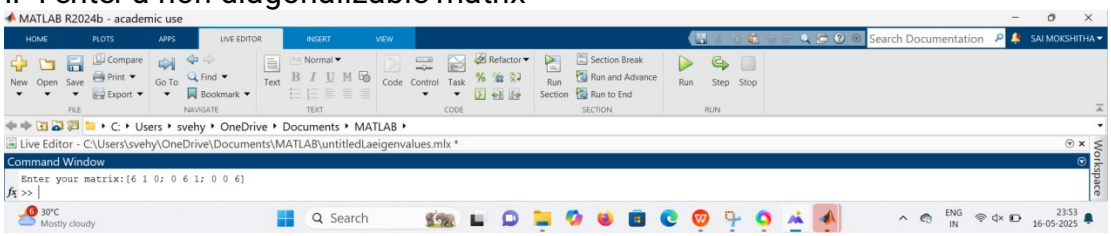


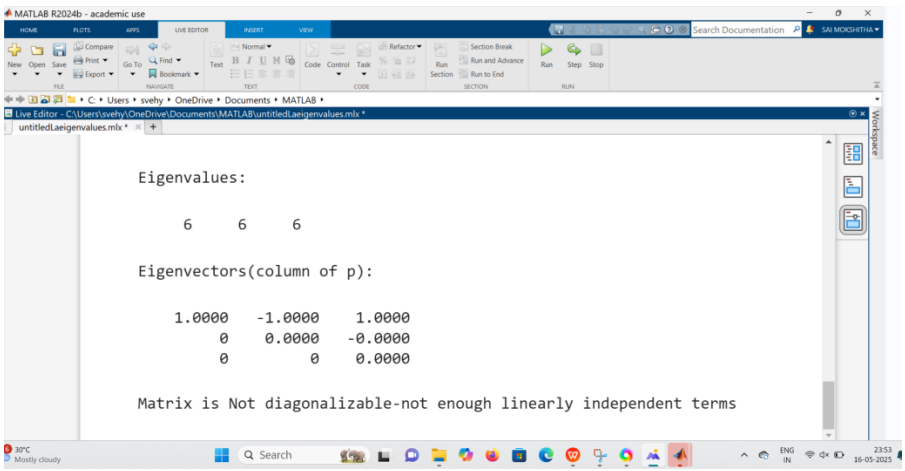
32b)



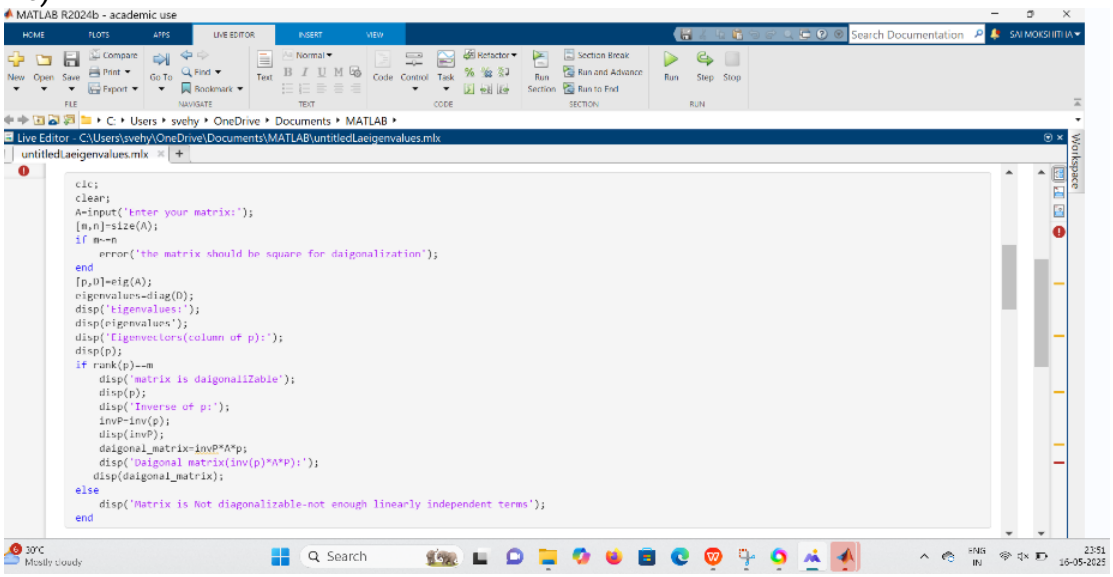
OUTPUT:

IF I enter a non diagonalizable matrix



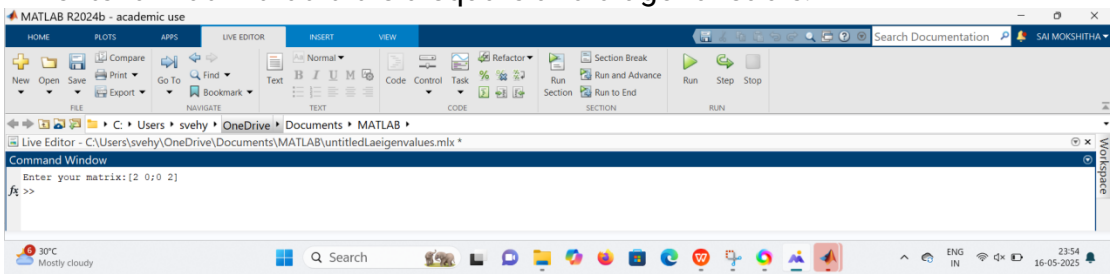


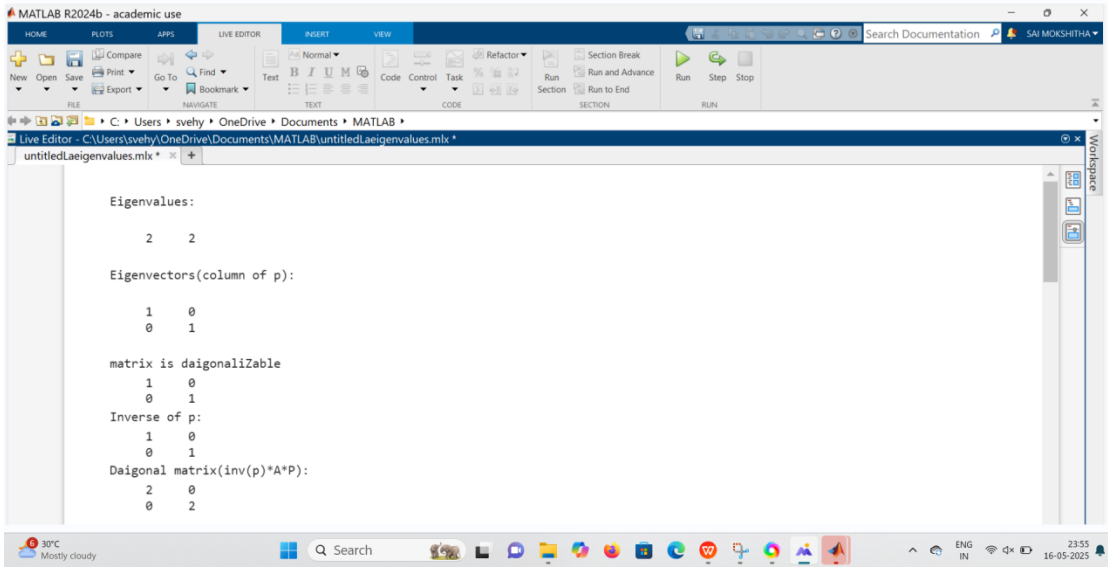
32c)



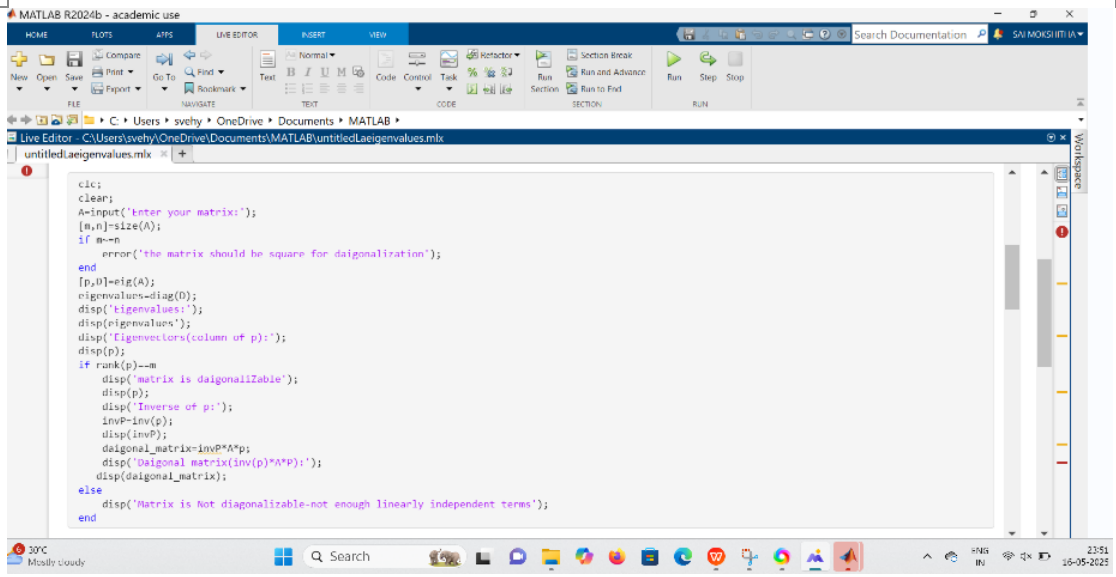
OUTPUT:

IF I enter a matrix that it it is a square and diagonalisable:



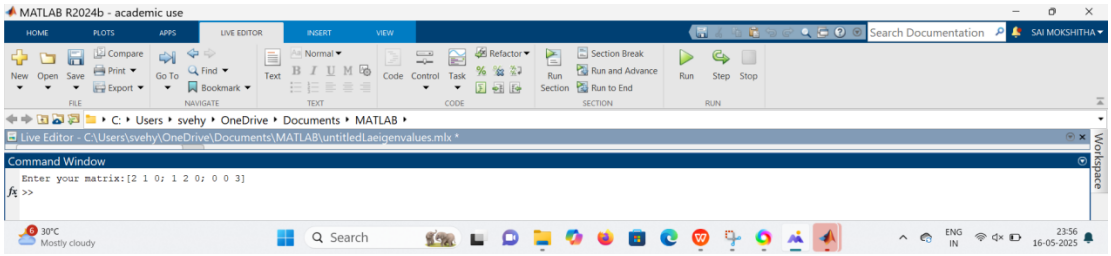


32d)

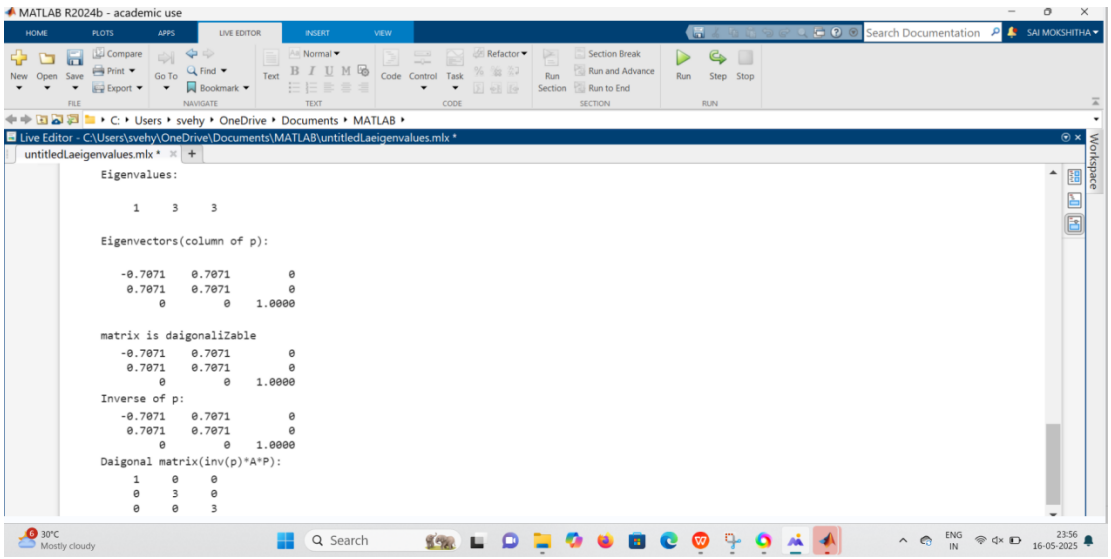


```
clc;
clear;
A=input('Enter your matrix:');
[m,n]=size(A);
if m~=n
    error('the matrix should be square for daigonalization');
end
[p,D]=eig(A);
eigenvalues=diag(D);
disp('Eigenvalues:');
disp(eigenvalues');
disp('Eigenvectors(column of p):');
disp(p);
if rank(p)==m
    disp('matrix is daigonaliZable');
    disp(p);
    disp('Inverse of p:');
    invP=inv(p);
    disp(invP);
    daigonal_matrix=invP*A*p;
    disp('Daigonal matrix(inv(p)*A*P):');
   disp(daigonal_matrix);
else
    disp('Matrix is Not diagonalizable-not enough linearly independent terms');
end
```

OUTPUT:



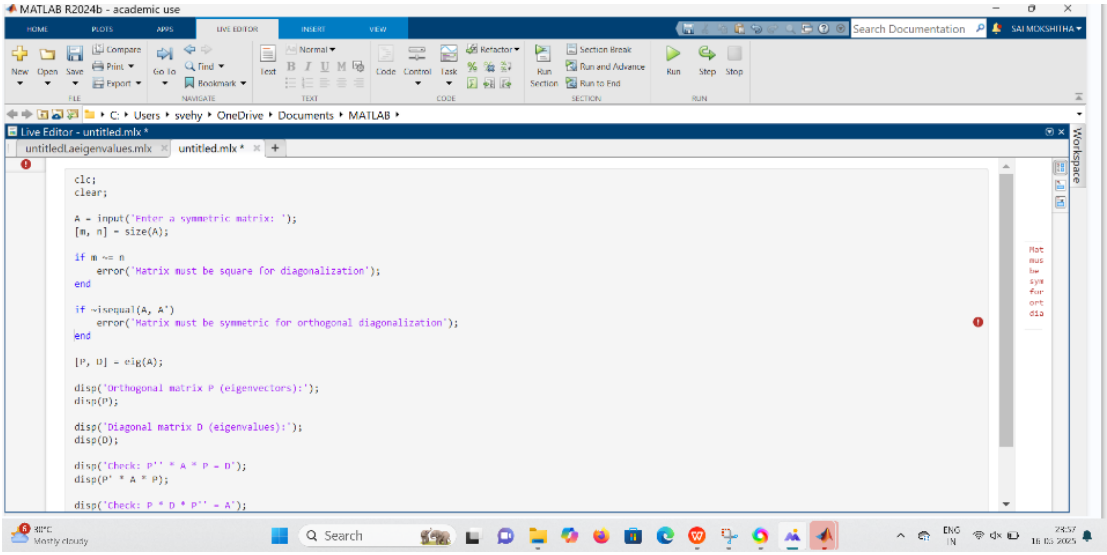
```
Enter your matrix:[2 1 0; 1 2 0; 0 0 3]
fx >>
```



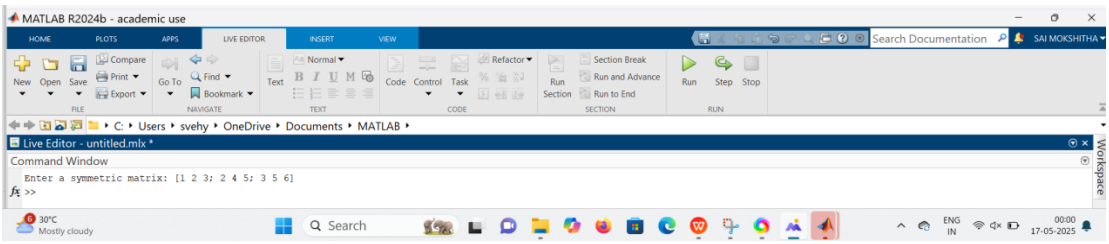
```
Eigenvalues:

     1     3     3

Eigenvectors(column of p):

   -0.7071    0.7071         0
    0.7071    0.7071         0
         0         0    1.0000

matrix is daigonaliZable
   -0.7071    0.7071         0
    0.7071    0.7071         0
         0         0    1.0000
Inverse of p:
   -0.7071    0.7071         0
    0.7071    0.7071         0
         0         0    1.0000
Daigonal matrix(inv(p)*A*P):
     1     0     0
     0     3     0
     0     0     3
```

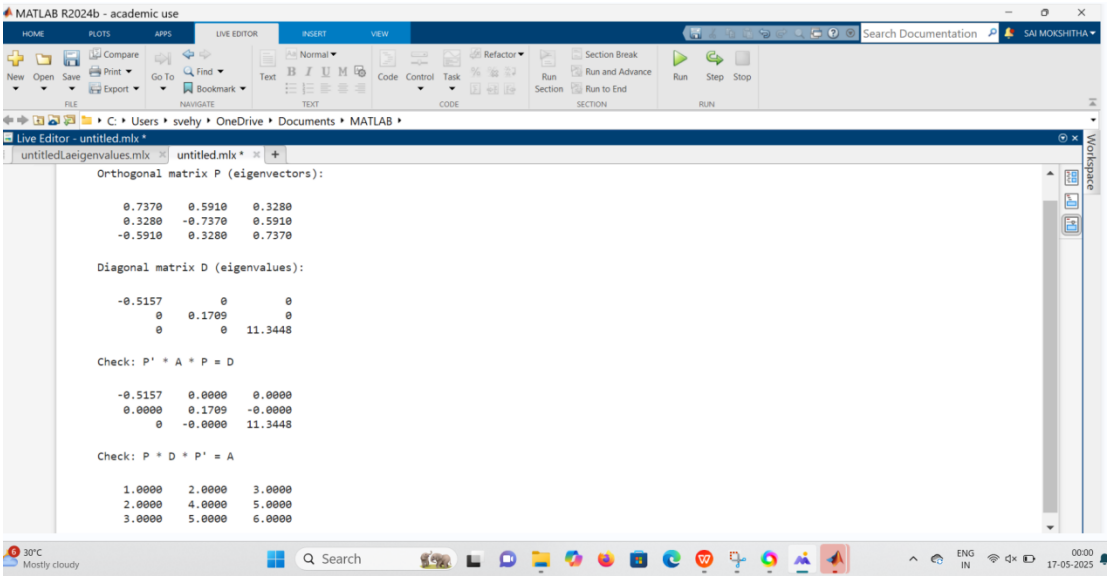
33a)EXERCISE QUESTION:WRITE A MATLAB CODE TO ORTOGONALLY DIAGONALISE A GIVEN SYMMETRIC MATRIX .ASK FOR INPUT CHECK

WHETERER THE MATRIX IS SQUARE OR NOT.
IF I ENTER A SYMMETRIC MATRIX

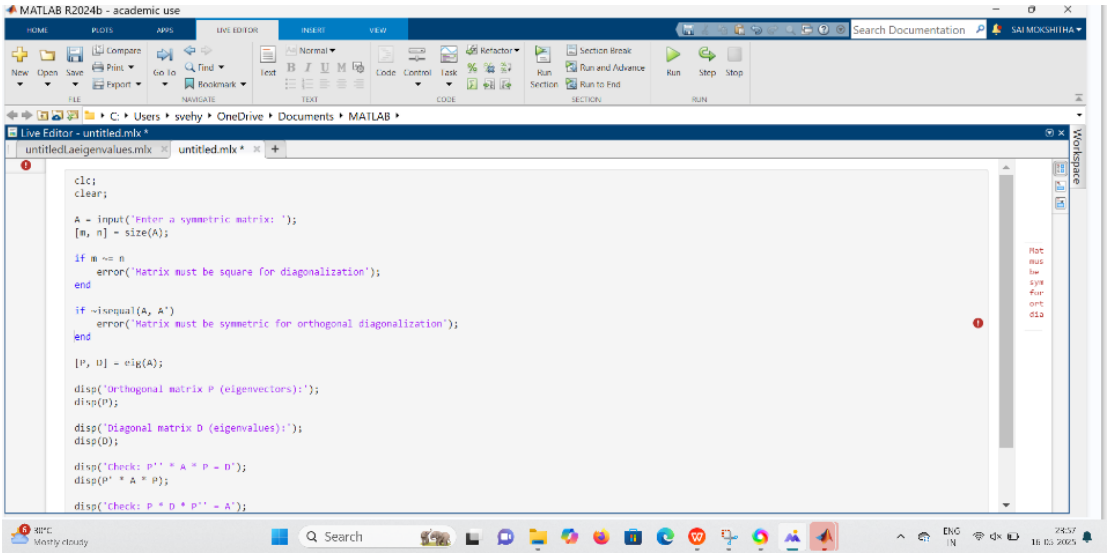


Output:





33b)



OUTPUT:
IF I ENTER A NON SYMMETRIC MATRIX