MATLAB PROJECT 3

Please include this page in your Group file, as a front page. Type in the group number and the names of all members WHO PARTICIPATED in this project.

GROUP # 4

FIRST & LAST NAMES (UFID numbers are NOT required):

_1._Edwin Salcedo

_2._Anisha Paul

_3._Brandon Tran

_4._John Dinh

_5._Ananya Kakaveti

_6._Yaaseen Mohammad

By including your names above, each of you had confirmed that you did the work and agree with the work submitted.

Project 3

Part I. Subspaces & Bases

Exercise 1

```
type columnspaces.m
```

```
function []=columnspaces(A,B)
format
m=size(A,1);
n=size(B,1);

if m~=n
    fprintf('Col A and Col B are subspaces of different spaces')
    return
else
    fprintf('Col A and Col B are subspaces of R^%i\n',m)
    f=rank(A);
    g=rank(B);
    if f~=g
        fprintf(['dimensions of Col A and Col B are different\n' ...
        'and Col A cannot be the same as Col B\n'])
        if isequal(f,m)
            fprintf('Col A is the whole R^%i\n',m)
        else if isequal(g,m)
            fprintf('Col B is the whole R^%i\n',m)
            end
        end
    else
        if f == m && g == m
            fprintf('Col A = Col B = R^%i\n',m)
        else
            if isequal(A,B)
                disp('ColA = ColB')
            else
                fprintf(['dimensions of Col A and Col B are the same ' ...
                    'but ColA ~= ColB\n'])
            end
        end
    end
end
end
```

```
%(a)
A=magic(3)
```

```
A = 3×3
     8     1     6
     3     5     7
     4     9     2
```

```
B=hilb(3)
```

```
B = 3×3
    1.0000    0.5000    0.3333
    0.5000    0.3333    0.2500
    0.3333    0.2500    0.2000
```

```
columnspaces(A,B)
```

```
Col A and Col B are subspaces of R^3
Col A = Col B = R^3
```

```
%(b)
A=magic(4)
```

```
A = 4×4
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

```
B=hilb(4)
```

```
B = 4×4
    1.0000    0.5000    0.3333    0.2500
    0.5000    0.3333    0.2500    0.2000
    0.3333    0.2500    0.2000    0.1667
    0.2500    0.2000    0.1667    0.1429
```

```
columnspaces(A,B)
```

```
Col A and Col B are subspaces of R^4
dimensions of Col A and Col B are different
and Col A cannot be the same as Col B
Col B is the whole R^4
```

```
%(c)
A=pascal(4)
```

```
A = 4×4
     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20
```

```
B=magic(4)
```

```
B = 4×4
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

```
columnspaces(A,B)
```

```
Col A and Col B are subspaces of R^4
dimensions of Col A and Col B are different
and Col A cannot be the same as Col B
Col A is the whole R^4
```

```
%(d)
A=magic(4)
```

```
A = 4×4
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

```
B=eye(3)
```

```
B = 3×3
     1     0     0
     0     1     0
     0     0     1
```

```
columnspaces(A,B)
```

```
Col A and Col B are subspaces of different spaces
```

```
%(e)
A=magic(4)
```

```
A = 4×4
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

```
B=rref(A)
```

```
B = 4×4
     1     0     0     1
     0     1     0     3
     0     0     1    -3
     0     0     0     0
```

```
columnspaces(A,B)
```

```
Col A and Col B are subspaces of R^4
dimensions of Col A and Col B are the same but ColA ~= ColB
```

```
%(f)
A=magic(4);
B=rref(A);
A=A'
```

```
A = 4×4
    16     5     9     4
     2    11     7    14
     3    10     6    15
    13     8    12     1
```

```
B=B'
```

```
B = 4×4
     1     0     0     0
     0     1     0     0
     0     0     1     0
     1     3    -3     0
```

```
columnspaces(A,B)
```

```
Col A and Col B are subspaces of R^4
dimensions of Col A and Col B are the same but ColA ~= ColB
```

```
%(g)
A=[2 -4 -2 3;6 -9 -5 8;2 -7 -3 9;4 -2 -2 -1;-6 3 3 4]
```

```
A = 5×4
     2    -4    -2     3
     6    -9    -5     8
```

```
     2    -7    -3     9
     4    -2    -2    -1
    -6     3     3     4
```

```
B=rref(A)
```

```
B = 5×4
    1.0000         0   -0.3333         0
         0    1.0000    0.3333         0
         0         0         0    1.0000
         0         0         0         0
         0         0         0         0
```

```
columnspaces(A,B)
```

```
Col A and Col B are subspaces of R^5
dimensions of Col A and Col B are the same but ColA ~= ColB
```

```
%(h)
A=[2 -4 -2 3;6 -9 -5 8;2 -7 -3 9;4 -2 -2 -1;-6 3 3 4];
B=([rref(A);zeros(5,4)]);
A=A'
```

```
A = 4×5
     2     6     2     4    -6
    -4    -9    -7    -2     3
    -2    -5    -3    -2     3
     3     8     9    -1     4
```

```
B=B'
```

```
B = 4×10
    1.0000         0         0         0         0         0         0         0···
         0    1.0000         0         0         0         0         0         0
   -0.3333    0.3333         0         0         0         0         0         0
         0         0    1.0000         0         0         0         0         0
```

```
columnspaces(A,B)
```

```
Col A and Col B are subspaces of R^4
dimensions of Col A and Col B are the same but ColA ~= ColB
```

```
%(i)
A=magic(4)
```

```
A = 4×4
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

```
B=[eye(3);zeros(1,3)]
```

```
B = 4×3
     1     0     0
     0     1     0
     0     0     1
     0     0     0
```

```
columnspaces(A,B)
```

```
Col A and Col B are subspaces of R^4
dimensions of Col A and Col B are the same but ColA ~= ColB
```

```
%(j)
A=pascal(3)
```

```
A = 3×3
     1     1     1
     1     2     3
     1     3     6
```

```
B=[hilb(3),eye(3)]
```

```
B = 3×6
    1.0000    0.5000    0.3333    1.0000         0         0
    0.5000    0.3333    0.2500         0    1.0000         0
    0.3333    0.2500    0.2000         0         0    1.0000
```

```
columnspaces(A,B)
```

```
Col A and Col B are subspaces of R^3
Col A = Col B = R^3
```

In exercises f and h, transposing two matricies will not change the row spaces even if there are row operations performed.

In exercises e and g, the use of elementary row operations would affect the way column space works as it considers all scalar multiples rather than the values of each column.

```
type shrink
```

```
function [pivot,B]=shrink(A)
[~,pivot]=rref(A);
B=A(:,pivot);
end
```

```
%Part 1
A = magic(4)
```

```
A = 4×4
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

```
[pivot, colbasis1] = shrink(A)
```

```
pivot = 1×3
     1     2     3
colbasis1 = 4×3
    16     2     3
     5    11    10
     9     7     6
     4    14    15
```

```
R = rref(A'), sym_colbasis2 = colspace(sym(A))
```

```
R = 4×4
     1     0     0     1
     0     1     0     3
     0     0     1    -3
     0     0     0     0
sym_colbasis2 =
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 3 & -3 \end{pmatrix}$$

```
colbasis2 = double(sym_colbasis2)
```

```
colbasis2 = 4×3
     1     0     0
     0     1     0
     0     0     1
     1     3    -3
```

```
%Part 2

type homobasis
```

```
function C = homobasis(A)
  format
  [~,n]=size(A);
  R=rref(A);
  rankA=rank(A);
  if rankA==n
```

```
        disp('the homogeneous system has only the trivial solution')
        C=zeros(n,1);
    else
        disp('the homogeneous system has non-trivial solutions')
        [~,pivot]=rref(A);
        nonpivot=setdiff(1:n,pivot);
        q=numel(nonpivot);
        j=1:q;
        fprintf('a free variable is x%i\n',nonpivot(j))
        C=zeros(n,q);
        R=R(1:rankA,nonpivot);
        C(pivot,:)=-R;
        C(nonpivot,:)=eye(q);
        if rank(C)== size(C,2) && ~any(closetozeroroundoff(A*C,5),'all')
            disp('columns of C form a basis for solution set of homogeneous system')
        else
C=[]; end
end
```

```
type noll
```

```
function [C,p] = noll(A)
n=size(A,2);
rankA=rank(A);

p = n - rankA;
fprintf('Nul A is a %i-dimensional subspace of R^%i/n', p , n)
C = homobasis(A)
zerovector = 1;

n1 = size(A, 1);
for i = 1:n1
    if n ==1 &&A(i, :)==0
        zerovector = 1;

    else
        zerovector = 0;
        break
    end
end

if zerovector==1
    disp('Nul A is spanned by the column of C')
else
    disp('a basis for Nul A is formed by the columns of the matrix C')
end
```

```
A = magic(5)
```

```
A = 5×5
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
```

```
[C, p] = noll(A);
```

```
Nul A is a 0-dimensional subspace of R^5/nthe homogeneous system has only the trivial solution
C = 5×1
     0
     0
     0
     0
     0
a basis for Nul A is formed by the columns of the matrix C
```

```
N = null(A, 'r')
```

```
N =

  5×0 empty double matrix
```

```
isequal(C, N)
```

```
ans = logical
   0
```

```
%(a)
A = [2 -4 -2 3; 6 -9 -5 8; 2 -7 -3 9; 4 -2 -2 -1; -6 3 3 4]
```

```
A = 5×4
     2    -4    -2     3
     6    -9    -5     8
     2    -7    -3     9
     4    -2    -2    -1
    -6     3     3     4
```

```
[C, p] = noll(A);
```

```
Nul A is a 1-dimensional subspace of R^4/nthe homogeneous system has non-trivial solutions
a free variable is x3
columns of C form a basis for solution set of homogeneous system
C = 4×1
    0.3333
   -0.3333
    1.0000
         0
a basis for Nul A is formed by the columns of the matrix C
```

```
%(b)
A = ones(5)
```

```
A = 5×5
     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
```

```
[C, p] = noll(A);
```

```
Nul A is a 4-dimensional subspace of R^5/nthe homogeneous system has non-trivial solutions
a free variable is x2
a free variable is x3
a free variable is x4
a free variable is x5
columns of C form a basis for solution set of homogeneous system
```

```
C = 5×4
    -1    -1    -1    -1
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
a basis for Nul A is formed by the columns of the matrix C
```

```
%(c)
A = [magic(4), ones(4, 1)]
```

```
A = 4×5
    16     2     3    13     1
     5    11    10     8     1
     9     7     6    12     1
     4    14    15     1     1
```

```
[C, p] = noll(A);
```

```
Nul A is a 2-dimensional subspace of R^5/nthe homogeneous system has non-trivial solutions
a free variable is x4
a free variable is x5
columns of C form a basis for solution set of homogeneous system
C = 5×2
   -1.0000   -0.0588
   -3.0000   -0.1176
    3.0000    0.0588
    1.0000         0
         0    1.0000
a basis for Nul A is formed by the columns of the matrix C
```

```
%(d)
A = [pascal(4); ones(2, 4)]
```

```
A = 6×4
     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20
     1     1     1     1
     1     1     1     1
```

```
[C, p] = noll(A);
```

```
Nul A is a 0-dimensional subspace of R^4/nthe homogeneous system has only the trivial solution
C = 4×1
     0
     0
     0
     0
a basis for Nul A is formed by the columns of the matrix C
```

```
%Part 3
A = [2 -4 -2 3; 6 -9 -5 8; 4 -5 -3 5; 4 -2 -2 -1; -6 3 3 4]
```

```
A = 5×4
     2    -4    -2     3
     6    -9    -5     8
     4    -5    -3     5
     4    -2    -2    -1
    -6     3     3     4
```

```
[pivot , rowbasis1] = shrink(A)
```

```
pivot = 1×3
     1     2     4
rowbasis1 = 5×3
     2    -4     3
     6    -9     8
     4    -5     5
     4    -2    -1
    -6     3     4
```

```
B = rref(A)
```

```
B = 5×4
    1.0000         0   -0.3333         0
         0    1.0000    0.3333         0
         0         0         0    1.0000
         0         0         0         0
         0         0         0         0
```

```
[pivot, rowbasis2] = shrink(A)
```

```
pivot = 1×3
     1     2     4
rowbasis2 = 5×3
     2    -4     3
     6    -9     8
     4    -5     5
     4    -2    -1
    -6     3     4
```

Part II. Isomorphism & Change of Basis

Exercise 3

```
type closetozeroroundoff.m
```

```
function B=closetozeroroundoff(A,p)
A(abs(A)<10^-p)=0;
B=A;
end
```

```
type shrink.m
```

```
function [pivot,B]=shrink(A)
[~,pivot]=rref(A);
B=A(:,pivot);
end
```

```
type polyspace.m
```

```
function P=polyspace(B,Q,r)
format
n=numel(B);     %number of elements in matrix B also dimension of Pn-1
P = zeros(n);
for i=1:n
    P(:,i) = fliplr(sym2poly(B(i)));
end
P=closetozeroroundoff(P,7);
fprintf('matrix of E-coordinate vectors of the polynomials in B is\n')
P

if rank(P) == n
    fprintf('the polynomials in B form a basis for P%d\n',n-1)
else
    fprintf('the polynomials in B do not form a basis for P%d\n',n-1)
    [~,P]=shrink([P eye(n)]);
    disp('the new matrix P is')
    P
    for i=1:n
        B(i) = poly2sym(flipud(P(:,i)));
    end
    disp('the constructed basis is')
    B
end

% Given Q through standard basis E
Q = fliplr(sym2poly(Q))';
Q = closetozeroroundoff(Q,7);
Matrix = [P Q];
Matrix = rref(Matrix);
for i=1:n+1 % makes sure the last column (B-coord of Q) is assigned to q
    q = Matrix(:,i);
end
q = closetozeroroundoff(q,7);
fprintf('the B-coordinate vector of Q is\n')
q

% Given B-coordinate vector r
for i=1:n
    s = r(i);
    B(:,i) = s*B(:,i);
end
R = fliplr(sym2poly(sum(B)))';
```

```
closetozeroroundoff(R,7);
fprintf('the polynomial whose B-coordinates form the vector r is\n')
R
```

```
syms x
%(a)
B=[x^3+3*x^2,10^(-8)*x^3+x,10^(-8)*x^3+4*x^2+x,x^3+x]
```

B =

$$\left(x^3+3\,x^2 \quad \frac{x^3}{100000000}+x \quad \frac{x^3}{100000000}+4\,x^2+x \quad x^3+x\right)$$

```
Q=10^(-8)*x^3+x^2+6*x+3
```

Q =

$$\frac{x^3}{100000000}+x^2+6\,x+3$$

```
r=[2;-1;3;-2]
```

```
r = 4×1
     2
    -1
     3
    -2
```

```
P=polyspace(B,Q,r);
```

```
matrix of E-coordinate vectors of the polynomials in B is
P = 4×4
     0     0     0     0
     0     1     1     1
     3     0     4     0
     1     0     0     1
the polynomials in B do not form a basis for P3
the new matrix P is
P = 4×4
     0     0     0     1
     0     1     1     0
     3     0     4     0
     1     0     0     0
the constructed basis is
```

B = $\left(x^3+3\,x^2 \quad x \quad 4\,x^2+x \quad 1\right)$

```
the B-coordinate vector of Q is
q = 4×1
         0
    5.7500
    0.2500
    3.0000
the polynomial whose B-coordinates form the vector r is
R = 4×1
    -2
     2
    18
     2
```

```
%(b)
B=[x^3-1,10^(-8)*x^3+2*x^2,10^(-8)*x^3+x,x^3+x]
```

B =

$$\left(x^3-1 \quad \frac{x^3}{100000000}+2\,x^2 \quad \frac{x^3}{100000000}+x \quad x^3+x\right)$$

```
P=polyspace(B,Q,r);
```

```
matrix of E-coordinate vectors of the polynomials in B is
P = 4×4
    -1     0     0     0
     0     0     1     1
     0     2     0     0
     1     0     0     1
the polynomials in B form a basis for P3
the B-coordinate vector of Q is
q = 4×1
   -3.0000
    0.5000
    3.0000
    3.0000
the polynomial whose B-coordinates form the vector r is
R = 4×1
   -2.0000
    1.0000
   -2.0000
    0.0000
```

```
%(c)
B=[x^3+1,10^(-8)*x^3+x^2+1,10^(-8)*x^3+x+1,10^(-8)*x^3+1]
```

B =

$$\left(x^3+1 \quad \frac{x^3}{100000000}+x^2+1 \quad \frac{x^3}{100000000}+x+1 \quad \frac{x^3}{100000000}+1\right)$$

```
Q=10^(-8)*x^3+3*x^2+x+6
```

Q =

$$\frac{x^3}{100000000}+3\,x^2+x+6$$

```
r=[1;-4;2;3]
```

```
r = 4×1
     1
    -4
     2
     3
```

```
P=polyspace(B,Q,r);
```

```
matrix of E-coordinate vectors of the polynomials in B is
P = 4×4
     1     1     1     1
     0     0     1     0
     0     1     0     0
     1     0     0     0
the polynomials in B form a basis for P3
the B-coordinate vector of Q is
q = 4×1
     0
```

```
     3
     1
     2
the polynomial whose B-coordinates form the vector r is
R = 4×1
    2.0000
    2.0000
   -4.0000
    1.0000
```

```
%(d)
B=[x^4+x^3+x^2+1,10^(-8)*x^4+x^3+x^2+x+1,10^(-8)*x^4+x^2+x+1,10^(-8)*x^4+x+1,10^(-8)*x^4+1]
```

B =

$$\left(x^4 + x^3 + x^2 + 1 \quad \frac{x^4}{100000000} + x^3 + x^2 + x + 1 \quad \frac{x^4}{100000000} + x^2 + x + 1 \quad \frac{x^4}{100000000} + x + 1 \quad \frac{x^4}{100000000} \right.$$

```
Q=10^(-8)*x^4+3*x^3-1
```

Q =

$$\frac{x^4}{100000000} + 3\,x^3 - 1$$

```
r=diag(pascal(5))
```

```
r = 5×1
     1
     2
     6
    20
    70
```

```
P=polyspace(B,Q,r);
```

```
matrix of E-coordinate vectors of the polynomials in B is
P = 5×5
     1     1     1     1     1
     0     1     1     1     0
     1     1     1     0     0
     1     1     0     0     0
     1     0     0     0     0
the polynomials in B form a basis for P4
the B-coordinate vector of Q is
q = 5×1
     0
     3
    -3
     0
    -1
the polynomial whose B-coordinates form the vector r is
R = 5×1
   99.0000
   28.0000
    9.0000
    3.0000
    1.0000
```

```
%(e)
B=[x^3+3*x^2,10^(-8)*x^3+x,x^3+3*x^2,2*x^3+6*x^2+x]
```

B =

$$\left(x^3+3\,x^2 \quad \frac{x^3}{100000000}+x \quad x^3+3\,x^2 \quad 2\,x^3+6\,x^2+x\right)$$

```
Q=10^(-8)*x^3+x^2+6*x+3
```

Q =

$$\frac{x^3}{100000000}+x^2+6\,x+3$$

```
r=[-1;3;2;4]
```

```
r = 4×1
    -1
     3
     2
     4
```

```
P=polyspace(B,Q,r);
```

```
matrix of E-coordinate vectors of the polynomials in B is
P = 4×4
     0     0     0     0
     0     1     0     1
     3     0     3     6
     1     0     1     2
the polynomials in B do not form a basis for P3
the new matrix P is
P = 4×4
     0     0     1     0
     0     1     0     0
     3     0     0     1
     1     0     0     0
the constructed basis is
```

B = $\left(x^3+3\,x^2 \quad x \quad 1 \quad x^2\right)$

```
the B-coordinate vector of Q is
q = 4×1
     0
     6
     3
     1
the polynomial whose B-coordinates form the vector r is
R = 4×1
     2
     3
     1
    -1
```

```
%(f)
B=[x^3+2*x^2+3*x+1,2*x^3+4*x^2+6*x+2,3*x^3+6*x^2+9*x+3,10^(-8)*x^3-2*x^2+1]
```

B =

$$\left(x^3+2\,x^2+3\,x+1 \quad 2\,x^3+4\,x^2+6\,x+2 \quad 3\,x^3+6\,x^2+9\,x+3 \quad \frac{x^3}{100000000}-2\,x^2+1\right)$$

```
Q=10^(-8)*x^3+x^2+6*x+3
```

Q =

$$\frac{x^3}{100000000} + x^2 + 6\,x + 3$$

```
r=[-2;1;-3;5]
```

```
r = 4×1
    -2
     1
    -3
     5
```

```
P=polyspace(B,Q,r);
```

```
matrix of E-coordinate vectors of the polynomials in B is
P = 4×4
     1     2     3     1
     3     6     9     0
     2     4     6    -2
     1     2     3     0
the polynomials in B do not form a basis for P3
the new matrix P is
P = 4×4
     1     1     1     0
     3     0     0     1
     2    -2     0     0
     1     0     0     0
the constructed basis is
```

B = $\left(x^3 + 2\,x^2 + 3\,x + 1 \quad 1 - 2\,x^2 \quad 1 \quad x\right)$

```
the B-coordinate vector of Q is
q = 4×1
         0
   -0.5000
    3.5000
    6.0000
the polynomial whose B-coordinates form the vector r is
R = 4×1
    -4
    -1
    -6
    -2
```

Exercise 4

```
type closetozeroroundoff.m
```

```
function B=closetozeroroundoff(A,p)
A(abs(A)<10^-p)=0;
B=A;
end
```

```
type quer.m
```

```
function [Q,R] = quer(A)
factorization = false;
unitary = false;
utriangle = false;
format
[m,n]=size(A);
[Q,R]=qr(A)

if(closetozeroroundoff(A-Q*R,7)==0)
    factorization = true;
end

if (closetozeroroundoff(Q'*Q-eye(m),7)==0)
    unitary = true;
end

if(istriu(R)==1)
    utriangle = true;
end

if(factorization && unitary && utriangle)
    disp('Q*R forms an orthogonal-triangular decomposition of A')
else
    disp('What is wrong?!')
    return;
end

if(m~=n)
    return;
end

if(m==n)
    k=0;
    while(any(closetozeroroundoff(A-triu(A),9),'all'))
        A=R*Q;
        [Q,R]=qr(A);
        k=k+1;
    end
    B=A;
    fprintf(['the matrix which is both close to upper triangular\n'     'and similar to A is\n'])
    B
    fprintf('the number of iterations is %i\n',k)
    disp('the vector of eigenvalues of matrix A is approximated by')
    E=diag(B)
    E=sort(E); L=eig(A); L=sort(L);
    if ~any(closetozeroroundoff(E-L,7))
        disp('Great! The eigenvalues are found correctly!')
    else
        disp('Oh no! I need to check the code!')
    end
end
```

```
%(a)
A=randi(10,4,5)
```

```
A = 4×5
     6     7     8     5     6
     3     2     1     5     6
     8     4    10     5     9
     2     7     8     4     8
```

```
[Q,R] = quer(A);
```

```
Q = 4×4
   -0.5644    0.3184    0.3977   -0.6495
   -0.2822   -0.0785    0.7094    0.6411
   -0.7526   -0.4208   -0.4832    0.1518
   -0.1881    0.8458   -0.3242    0.3796
R = 4×5
  -10.6301   -8.8428  -13.8286   -8.7487  -13.3582
         0    6.3091    5.0271    2.4785    4.4184
         0         0   -3.5353    1.8222   -0.3006
         0         0         0    2.2354    4.3526
Q*R forms an orthogonal-triangular decomposition of A
```

```
%(b)
A=ones(5,4);
[Q,R] = quer(A);
```

```
Q = 5×5
   -0.4472    0.8944   -0.0000         0         0
   -0.4472   -0.2236    0.8660         0    0.0000
   -0.4472   -0.2236   -0.2887    0.8165   -0.0000
   -0.4472   -0.2236   -0.2887   -0.4082   -0.7071
   -0.4472   -0.2236   -0.2887   -0.4082    0.7071
R = 5×4
   -2.2361   -2.2361   -2.2361   -2.2361
         0   -0.0000   -0.0000   -0.0000
         0         0   -0.0000   -0.0000
         0         0         0   -0.0000
         0         0         0         0
Q*R forms an orthogonal-triangular decomposition of A
```

```
%(c)
A=diag([1,2,3,4,5])
```

```
A = 5×5
     1     0     0     0     0
     0     2     0     0     0
     0     0     3     0     0
     0     0     0     4     0
     0     0     0     0     5
```

```
[Q,R] = quer(A);
```

```
Q = 5×5
     1     0     0     0     0
     0     1     0     0     0
     0     0     1     0     0
     0     0     0     1     0
     0     0     0     0     1
R = 5×5
     1     0     0     0     0
     0     2     0     0     0
```

```
     0     0     3     0     0
     0     0     0     4     0
     0     0     0     0     5
Q*R forms an orthogonal-triangular decomposition of A
the matrix which is both close to upper triangular
and similar to A is
B = 5×5
     1     0     0     0     0
     0     2     0     0     0
     0     0     3     0     0
     0     0     0     4     0
     0     0     0     0     5
the number of iterations is 0
the vector of eigenvalues of matrix A is approximated by
E = 5×1
     1
     2
     3
     4
     5
Great! The eigenvalues are found correctly!
```

```
%(d)
A=triu(magic(4))
```

```
A = 4×4
    16     2     3    13
     0    11    10     8
     0     0     6    12
     0     0     0     1
```

```
[Q,R] = quer(A);
```

```
Q = 4×4
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
R = 4×4
    16     2     3    13
     0    11    10     8
     0     0     6    12
     0     0     0     1
Q*R forms an orthogonal-triangular decomposition of A
the matrix which is both close to upper triangular
and similar to A is
B = 4×4
    16     2     3    13
     0    11    10     8
     0     0     6    12
     0     0     0     1
the number of iterations is 0
the vector of eigenvalues of matrix A is approximated by
E = 4×1
    16
    11
     6
     1
Great! The eigenvalues are found correctly!
```

```
%(e)
A=tril(magic(4),1)
```

```
A = 4×4
    16     2     0     0
     5    11    10     0
     9     7     6    12
     4    14    15     1
```

```
[Q,R] = quer(A);
```

```
Q = 4×4
   -0.8230    0.4186    0.1367   -0.3590
   -0.2572   -0.5155   -0.7600   -0.3009
   -0.4629   -0.1305   -0.0997    0.8711
   -0.2057   -0.7363    0.6275   -0.1478
R = 4×4
  -19.4422  -10.5955   -8.4352   -5.7607
         0  -16.0541  -16.9816   -2.3024
         0         0    1.2138   -0.5692
         0         0         0   10.3048
Q*R forms an orthogonal-triangular decomposition of A
the matrix which is both close to upper triangular
and similar to A is
B = 4×4
   25.4509  -11.0265    2.9773   -9.2243
    0.0000   14.9803    3.3384   -2.0094
   -0.0000   -0.0000   -7.7521    9.4488
    0.0000    0.0000    0.0000    1.3209
the number of iterations is 44
the vector of eigenvalues of matrix A is approximated by
E = 4×1
   25.4509
   14.9803
   -7.7521
    1.3209
Great! The eigenvalues are found correctly!
```

```
%(f)
A=triu(magic(5),-1)
```

```
A = 5×5
    17    24     1     8    15
    23     5     7    14    16
     0     6    13    20    22
     0     0    19    21     3
     0     0     0     2     9
```

```
[Q,R] = quer(A);
```

```
Q = 5×5
   -0.5944    0.7548    0.1595   -0.2009    0.1055
   -0.8042   -0.5579   -0.1179    0.1485   -0.0779
         0    0.3449   -0.5399    0.6798   -0.3569
         0         0   -0.8180   -0.5093    0.2673
         0         0         0    0.4648    0.8854
R = 5×5
  -28.6007  -18.2863   -6.2236  -16.0136  -21.7827
         0   17.3958    1.3333    5.1260    9.9838
         0         0  -23.2269  -28.3509  -13.8254
         0         0         0    4.3031   16.9742
         0         0         0         0    1.2544
Q*R forms an orthogonal-triangular decomposition of A
the matrix which is both close to upper triangular
and similar to A is
B = 5×5
```

```
   42.5639  -13.6250   -0.8346   -7.5113  -26.8386
   -0.0000   30.4211    4.8200    2.9240    2.5847
         0   -0.0000  -13.7128    4.7837    0.6927
         0         0   -0.0000    6.2866   14.3215
         0         0         0    0.0000   -0.5588
the number of iterations is 71
the vector of eigenvalues of matrix A is approximated by
E = 5×1
   42.5639
   30.4211
  -13.7128
    6.2866
   -0.5588
Great! The eigenvalues are found correctly!
```

```
%(g)
A=tril(A,1)
```

```
A = 5×5
    17    24     0     0     0
    23     5     7     0     0
     0     6    13    20     0
     0     0    19    21     3
     0     0     0     2     9
```

```
[Q,R] = quer(A);
```

```
Q = 5×5
   -0.5944    0.7548    0.1617   -0.1876    0.1248
   -0.8042   -0.5579   -0.1195    0.1387   -0.0923
         0    0.3449   -0.5473    0.6348   -0.4224
         0         0   -0.8124   -0.4855    0.3230
         0         0         0    0.5540    0.8325
R = 5×5
  -28.6007  -18.2863   -5.6292         0         0
         0   17.3958    0.5784    6.8982         0
         0         0  -23.3875  -28.0068   -2.4372
         0         0         0    3.6103    3.5293
         0         0         0         0    8.4619
Q*R forms an orthogonal-triangular decomposition of A
the matrix which is both close to upper triangular
and similar to A is
B = 5×5
   39.0188   -0.5303    0.2774   -0.6039   -0.2386
   -0.0000   33.7156    0.7246    0.7145    1.0669
         0   -0.0000  -14.8076    0.0083    0.7486
         0         0   -0.0000    9.0824   -0.6970
         0         0         0    0.0000   -2.0092
the number of iterations is 157
the vector of eigenvalues of matrix A is approximated by
E = 5×1
   39.0188
   33.7156
  -14.8076
    9.0824
   -2.0092
Great! The eigenvalues are found correctly!
```

```
%(h)
A=[1 1 4;0 -4 0;-5 -1 -8]
```

```
A = 3×3
     1     1     4
```

```
     0    -4     0
    -5    -1    -8
```

```
[Q,R] = quer(A);
```

```
Q = 3×3
   -0.1961    0.1887    0.9623
         0   -0.9813    0.1925
    0.9806    0.0377    0.1925
R = 3×3
   -5.0990   -1.1767   -8.6291
         0    4.0762    0.4529
         0         0    2.3094
Q*R forms an orthogonal-triangular decomposition of A
the matrix which is both close to upper triangular
and similar to A is
B = 3×3
   -4.0000   -0.0000   -9.1091
    0.0000   -4.0000    0.1562
    0.0000    0.0000   -3.0000
the number of iterations is 65
the vector of eigenvalues of matrix A is approximated by
E = 3×1
   -4.0000
   -4.0000
   -3.0000
Great! The eigenvalues are found correctly!
```

Exercise 5

Part 1

```
type elu.m
```

```
function [L,U,N,L1,P]=elu(A)
format
[m,n]=size(A);
rankA=rank(A);
A=sym(A);
R=A;
L=eye(m);
P=eye(m);
N=0;

% Forward Phase
% m = total rows, n = total columns
% For each row of matrix R
for k=1:m % increments rows
    %k
    for j = 1:n % increments columns
        %j
        if any(R(k:m,j))
            currentColumn = j;
            break;
        end
    end

    % Getting largest by absolute value column entry to be a pivot
    [~,y]=max(abs(R(k:m,j)));
    %y
    y=y+k-1;
    %fprintf('new y: ')
    %y
    %k
    % If that column entry isn't in the pivot position
    if y ~= k
         % Using row interchanging operation to move it into the pivot position

         %R([i y],j:n)=R([y i],j:n); (Original)
         %Complete ele2
         %R=ele2(m,k,y)*R;
         E2 = ele2(m,k,y);
         %for num1=1:m
         %      for num2=1:n
         %           R(num1,num2) = R(num1,num2)*E2(num1,num2);
         %      end
         %end
         R=E2*R;
         L=L*E2;
         %P
         P=E2*P;
         N=N+1;
    end

    if (k<m) % making zeroes below the pivot
        for i = (k+1):m
            %i
            %y
            if (R(i,currentColumn)~= 0)
                %R(y,currentColumn)
                %R(k+1,currentColumn)
                r = R(i,currentColumn)/R(k,currentColumn);
```

```
                E1=ele1(m,k,i,-r);
                %for num1=1:m
                %    for num2=1:n
                %        R(num1,num2) = R(num1,num2)*E1(num1,num2);
                %    end
                %end
                R=E1*R;
                L=L*inv(E1);
            end
        end
    end

    R=closetozeroroundoff(R,7);
end

U=R;
L1=P*L;
if rankA < m
    U = U(1:rankA,:);
    L = L(:,1:rankA);
end

if  ~any(closetozeroroundoff(A-L*U,7),"all")
    U=double(U);
else
    L=[];
    U=[];
end

%istril(L1)

i=1;
j=1;
isTril = 1;
while (i<m && j<n)
    if L1(i,j)==0
        isTril = 0;
        break;
    end
    i=i+1;
    j=j+1;
end


if (isTril)
    for i=1:size(L1,1)
        if L1(i,i) ~=1
            L1=[];
            P =[];
            break
        end
    end
else
    L1=[];
    P=[];
end
```

```
type closetozeroroundoff.m
```

```
function B=closetozeroroundoff(A,p)
A(abs(A)<10^-p)=0;
B=A;
end
```

```
%(a)
A=[0 0 1;1 0 0;0 1 0]
```

```
A = 3×3
     0     0     1
     1     0     0
     0     1     0
```

```
[L,U,N,L1,P]=elu(A)
```

```
L = 3×3
     0     0     1
     1     0     0
     0     1     0
U = 3×3
     1     0     0
     0     1     0
     0     0     1
N = 2
L1 = 3×3
     1     0     0
     0     1     0
     0     0     1
P = 3×3
     0     1     0
     0     0     1
     1     0     0
```

```
%(b)
A=magic(3)
```

```
A = 3×3
     8     1     6
     3     5     7
     4     9     2
```

```
[L,U,N,L1,P]=elu(A)
```

```
L = 3×3
    1.0000         0         0
    0.3750    0.5441    1.0000
    0.5000    1.0000         0
U = 3×3
    8.0000    1.0000    6.0000
         0    8.5000   -1.0000
         0         0    5.2941
N = 1
L1 = 3×3
    1.0000         0         0
    0.5000    1.0000         0
    0.3750    0.5441    1.0000
P = 3×3
     1     0     0
     0     0     1
     0     1     0
```

```
%(c)
A=magic(4)
```

```
A = 4×4
    16     2     3    13
     5    11    10     8
```

```
     9     7     6    12
     4    14    15     1
```

```
[L,U,N,L1,P]=elu(A)
```

```
L = 4×3
    1.0000         0         0
    0.3125    0.7685    1.0000
    0.5625    0.4352    1.0000
    0.2500    1.0000         0
U = 3×4
   16.0000    2.0000    3.0000   13.0000
         0   13.5000   14.2500   -2.2500
         0         0   -1.8889    5.6667
N = 1
L1 = 4×4
    1.0000         0         0         0
    0.2500    1.0000         0         0
    0.5625    0.4352    1.0000         0
    0.3125    0.7685    1.0000    1.0000
P = 4×4
     1     0     0     0
     0     0     0     1
     0     0     1     0
     0     1     0     0
```

```
%(d)
A=[1 1 4 3;0 -4 0 2;-5 -1 -8 1]
```

```
A = 3×4
     1     1     4     3
     0    -4     0     2
    -5    -1    -8     1
```

```
[L,U,N,L1,P]=elu(A)
```

```
L = 3×3
   -0.2000   -0.2000    1.0000
         0    1.0000         0
    1.0000         0         0
U = 3×4
   -5.0000   -1.0000   -8.0000    1.0000
         0   -4.0000         0    2.0000
         0         0    2.4000    3.6000
N = 1
L1 = 3×3
    1.0000         0         0
         0    1.0000         0
   -0.2000   -0.2000    1.0000
P = 3×3
     0     0     1
     0     1     0
     1     0     0
```

```
%(e)
A=[1 1 4;0 -4 0;-5 -1 -8; 2 3 -1]
```

```
A = 4×3
     1     1     4
     0    -4     0
    -5    -1    -8
     2     3    -1
```

```
[L,U,N,L1,P]=elu(A)
```

```
L = 4×3
   -0.2000   -0.2000   -0.5714
         0    1.0000         0
    1.0000         0         0
   -0.4000   -0.6500    1.0000
U = 3×3
   -5.0000   -1.0000   -8.0000
         0   -4.0000         0
         0         0   -4.2000
N = 2
L1 = 4×4
    1.0000         0         0         0
         0    1.0000         0         0
   -0.4000   -0.6500    1.0000         0
   -0.2000   -0.2000   -0.5714    1.0000
P = 4×4
     0     0     1     0
     0     1     0     0
     0     0     0     1
     1     0     0     0
```

```
%(f)
A=[1 2 3;2 4 6;3 6 4]
```

```
A = 3×3
     1     2     3
     2     4     6
     3     6     4
```

```
[L,U,N,L1,P]=elu(A)
```

```
L = 3×2
    0.3333    0.5000
    0.6667    1.0000
    1.0000         0
U = 2×3
    3.0000    6.0000    4.0000
         0         0    3.3333
N = 1
L1 = 3×3
    1.0000         0         0
    0.6667    1.0000         0
    0.3333    0.5000    1.0000
P = 3×3
     0     0     1
     0     1     0
     1     0     0
```

```
%(g)
A=magic(5)
```

```
A = 5×5
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
```

```
[L,U,N,L1,P]=elu(A)
```

```
L = 5×5
    0.7391    1.0000         0         0         0
    1.0000         0         0         0         0
    0.1739    0.2527    0.5164    1.0000         0
    0.4348    0.4839    0.7231    0.9231    1.0000
    0.4783    0.7687    1.0000         0         0
U = 5×5
   23.0000    5.0000    7.0000   14.0000   16.0000
         0   20.3043   -4.1739   -2.3478    3.1739
         0         0   24.8608   -2.8908   -1.0921
         0         0         0   19.6512   18.9793
         0         0         0         0  -22.2222
N = 3
L1 = 5×5
    1.0000         0         0         0         0
    0.7391    1.0000         0         0         0
    0.4783    0.7687    1.0000         0         0
    0.1739    0.2527    0.5164    1.0000         0
    0.4348    0.4839    0.7231    0.9231    1.0000
P = 5×5
     0     1     0     0     0
     1     0     0     0     0
     0     0     0     0     1
     0     0     1     0     0
     0     0     0     1     0
```

```
%(h)
A=randi(10,6,3)
```

```
A = 6×3
    10    10     9
     9     9    10
     5     4     6
     8     5     6
     5     3     2
    10     8     9
```

```
[L,U,N,L1,P]=elu(A)
```

```
L = 6×3
    1.0000         0         0
    0.9000         0    1.0000
    0.5000    0.3333    1.0000
    0.8000    1.0000         0
    0.5000    0.6667   -0.8947
    1.0000    0.6667    0.4211
U = 3×3
   10.0000   10.0000    9.0000
         0   -3.0000   -1.2000
         0         0    1.9000
N = 1
L1 = 6×6
    1.0000         0         0         0         0         0
    0.8000    1.0000         0         0         0         0
    0.5000    0.3333    1.0000         0         0         0
    0.9000         0    1.0000    1.0000         0         0
    0.5000    0.6667   -0.8947         0    1.0000         0
    1.0000    0.6667    0.4211         0         0    1.0000
P = 6×6
     1     0     0     0     0     0
     0     0     0     1     0     0
     0     0     1     0     0     0
     0     1     0     0     0     0
```

```
     0     0     0     0     1     0
     0     0     0     0     0     1
```

```
%(i)
A=randi(10,3,6)
```

```
A = 3×6
     5     8     7     5     3     4
     3     9     7     3     9     5
     9     3     2     8     9     7
```

```
[L,U,N,L1,P]=elu(A)
```

```
L = 3×3
    0.5556    0.7917    1.0000
    0.3333    1.0000         0
    1.0000         0         0
U = 3×6
    9.0000    3.0000    2.0000    8.0000    9.0000    7.0000
         0    8.0000    6.3333    0.3333    6.0000    2.6667
         0         0    0.8750    0.2917   -6.7500   -2.0000
N = 1
L1 = 3×3
    1.0000         0         0
    0.3333    1.0000         0
    0.5556    0.7917    1.0000
P = 3×3
     0     0     1
     0     1     0
     1     0     0
```

```
%(j)
A=pascal(5)
```

```
A = 5×5
     1     1     1     1     1
     1     2     3     4     5
     1     3     6    10    15
     1     4    10    20    35
     1     5    15    35    70
```

```
[L,U,N,L1,P]=elu(A)
```

```
L = 5×5
    1.0000         0         0         0         0
    1.0000    0.2500    0.7500   -1.0000    1.0000
    1.0000    0.5000    1.0000         0         0
    1.0000    0.7500    0.7500    1.0000         0
    1.0000    1.0000         0         0         0
U = 5×5
    1.0000    1.0000    1.0000    1.0000    1.0000
         0    4.0000   14.0000   34.0000   69.0000
         0         0   -2.0000   -8.0000  -20.5000
         0         0         0   -0.5000   -2.3750
         0         0         0         0   -0.2500
N = 1
L1 = 5×5
    1.0000         0         0         0         0
    1.0000    1.0000         0         0         0
    1.0000    0.5000    1.0000         0         0
    1.0000    0.7500    0.7500    1.0000         0
    1.0000    0.2500    0.7500   -1.0000    1.0000
```

```
P = 5×5
     1     0     0     0     0
     0     0     0     0     1
     0     0     1     0     0
     0     0     0     1     0
     0     1     0     0     0
```

```
%(k)
A=hilb(3)
```

```
A = 3×3
    1.0000    0.5000    0.3333
    0.5000    0.3333    0.2500
    0.3333    0.2500    0.2000
```

```
[L,U,N,L1,P]=elu(A)
```

```
L = 3×3
    1.0000         0         0
    0.5000    1.0000         0
    0.3333    1.0000    1.0000
U = 3×3
    1.0000    0.5000    0.3333
         0    0.0833    0.0833
         0         0    0.0056
N = 0
L1 = 3×3
    1.0000         0         0
    0.5000    1.0000         0
    0.3333    1.0000    1.0000
P = 3×3
     1     0     0
     0     1     0
     0     0     1
```

Part 2

```
type eluinv.m
```

```
function [invA,detA] = eluinv(A)
[~,n]=size(A);

if (rank(A)~= n)
    fprintf('A is not invertible')
    invA=[];
    detA=0;
    return
end


[L,U,N]=elu(A);
%fprintf('L U and N above are final')
%n
invL=rref([L eye(n)]);
invU=rref([U eye(n)]);

invL=invL(1:n,(n+1):2*n);
```

```
invU=invU(1:n,(n+1):2*n);

invA = invU*invL;
F=inv(A);

if  any(closetozeroroundoff(F-invA,7),"all")
    invA=[];
end

%U
diagVector = diag(U);
detA=((-1)^N)*(prod(diagVector));


d=det(A);
if  any(closetozeroroundoff(d-detA,7),"all")
    %fprintf('this')
    detA=[];
end
end
```

```
%(a)
A=[1 1 4;0 -4 0;-5 -1 -8]
```

```
A = 3×3
     1     1     4
     0    -4     0
    -5    -1    -8
```

```
[invA,detA] = eluinv(A)
```

```
invA = 3×3
   -0.6667   -0.0833   -0.3333
         0   -0.2500         0
    0.4167    0.0833    0.0833
detA = -48
```

```
%(b)
A=magic(4)
```

```
A = 4×4
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

```
[invA,detA] = eluinv(A)
```

```
A is not invertible
invA =

     []
detA = 0
```

```
%(c)
A=[2 1 -3 1;0 5 -3 5;-4 3 3 3;-2 5 1 3]
```

```
A = 4×4
     2     1    -3     1
     0     5    -3     5
    -4     3     3     3
    -2     5     1     3
```

```
[invA,detA] = eluinv(A)
```

```
A is not invertible
invA =

     []
detA = 0
```

```
%(d)
A=magic(5)
```

```
A = 5×5
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
```

```
[invA,detA] = eluinv(A)
```

```
invA = 5×5
   -0.0049    0.0512   -0.0354    0.0012    0.0034
    0.0431   -0.0373   -0.0046    0.0127    0.0015
   -0.0303    0.0031    0.0031    0.0031    0.0364
    0.0047   -0.0065    0.0108    0.0435   -0.0370
    0.0028    0.0050    0.0415   -0.0450    0.0111
detA = 5070000
```

```
%(e)
A=hilb(3)
```

```
A = 3×3
    1.0000    0.5000    0.3333
    0.5000    0.3333    0.2500
    0.3333    0.2500    0.2000
```

```
[invA,detA] = eluinv(A)
```

```
invA = 3×3
     9   -36    30
   -36   192  -180
    30  -180   180
detA = 4.6296e-04
```

```
%(f)
A=pascal(4)
```

```
A = 4×4
     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20
```

```
[invA,detA] = eluinv(A)
```

```
invA = 4×4
    4.0000   -6.0000    4.0000   -1.0000
   -6.0000   14.0000  -11.0000    3.0000
    4.0000  -11.0000   10.0000   -3.0000
   -1.0000    3.0000   -3.0000    1.0000
detA = 1
```

Part 3

```
type msystem.m
```

```
function [X1,X2,X] = msystem(A,B)
[~,n]=size(A);
[~,p]=size(B);
X1=[];
X2=[];
X=[];

if (rank(A)~= n)
    disp('A is not invertible and there is no unique solution')
    return
end

X1 = inv(A)*B;
disp('the solution calculated by using the inverse of A is')
disp(X1)

X2=A\B;
disp('the solution calculated by using the backslash operator is')
disp(X2)

[L,U] = elu(A);

%%% third method here
%n
%p
%n+1 is lft bound and n+p is rght bound
Y = rref([L B]);
Y=Y(:,(n+1):(n+p));

X=rref([U Y]);
X=X(:,(n+1):(n+p));

disp('the solution calculated by using LU factorization is')
disp(X)

if  any(closetozeroroundoff(X1-X2,7),"all") || any(closetozeroroundoff(X1-X,7),"all")
    disp('What! They do not match! Something is off...')
else
    disp('The solutions calculated by different methods match')
end



end
```

```
%(a)
A=magic(4), B=pascal(4);
```

```
A = 4×4
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

```
[X1,X2,X] = msystem(A,B);
```

```
A is not invertible and there is no unique solution
```

```
%(b)
A=[1 1 4;0 -4 0;-5 -1 -8], B=hilb(3)
```

```
A = 3×3
     1     1     4
     0    -4     0
    -5    -1    -8
B = 3×3
    1.0000    0.5000    0.3333
    0.5000    0.3333    0.2500
    0.3333    0.2500    0.2000
```

```
[X1,X2,X] = msystem(A,B);
```

```
the solution calculated by using the inverse of A is
   -0.8194   -0.4444   -0.3097
   -0.1250   -0.0833   -0.0625
    0.4861    0.2569    0.1764

the solution calculated by using the backslash operator is
   -0.8194   -0.4444   -0.3097
   -0.1250   -0.0833   -0.0625
    0.4861    0.2569    0.1764

the solution calculated by using LU factorization is
   -0.8194   -0.4444   -0.3097
   -0.1250   -0.0833   -0.0625
    0.4861    0.2569    0.1764

The solutions calculated by different methods match
```

```
%(c)
A=magic(5), B=randi(10,5,3)
```

```
A = 5×5
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
B = 5×3
     9     8     5
     7     9     5
     6     8     2
     4     1     9
     5     7     4
```

```
[X1,X2,X] = msystem(A,B);
```

```
the solution calculated by using the inverse of A is
    0.1229    0.1628    0.1843
    0.1576   -0.0046    0.1401
   -0.0379    0.0682    0.0436
    0.0499   -0.1507    0.2554
    0.1845    0.4320   -0.2388

the solution calculated by using the backslash operator is
    0.1229    0.1628    0.1843
    0.1576   -0.0046    0.1401
   -0.0379    0.0682    0.0436
    0.0499   -0.1507    0.2554
    0.1845    0.4320   -0.2388
```

```
the solution calculated by using LU factorization is
    0.1229    0.1628    0.1843
    0.1576   -0.0046    0.1401
   -0.0379    0.0682    0.0436
    0.0499   -0.1507    0.2554
    0.1845    0.4320   -0.2388

The solutions calculated by different methods match
```

```
%(d)
A=pascal(3),B=randi(10,3,5)
```

```
A = 3×3
     1     1     1
     1     2     3
     1     3     6
B = 3×5
     3     6     4    10     9
     4     6     6     8     2
     4     4     7     5     1
```

```
[X1,X2,X] = msystem(A,B);
```

```
the solution calculated by using the inverse of A is
    1.0000    4.0000    1.0000   11.0000   22.0000
    3.0000    4.0000    4.0000    0.0000  -19.0000
   -1.0000   -2.0000   -1.0000   -1.0000    6.0000

the solution calculated by using the backslash operator is
     1     4     1    11    22
     3     4     4     0   -19
    -1    -2    -1    -1     6

the solution calculated by using LU factorization is
     1     4     1    11    22
     3     4     4     0   -19
    -1    -2    -1    -1     6

The solutions calculated by different methods match
```

```
%(e)
A=magic(3), B=[magic(3),eye(3)]
```

```
A = 3×3
     8     1     6
     3     5     7
     4     9     2
B = 3×6
     8     1     6     1     0     0
     3     5     7     0     1     0
     4     9     2     0     0     1
```

```
[X1,X2,X] = msystem(A,B);
```

```
the solution calculated by using the inverse of A is
  Columns 1 through 5

    1.0000   -0.0000   -0.0000    0.1472   -0.1444
         0    1.0000         0   -0.0611    0.0222
         0    0.0000    1.0000   -0.0194    0.1889
```

```
  Column 6

    0.0639
    0.1056
   -0.1028

the solution calculated by using the backslash operator is
  Columns 1 through 5

    1.0000   -0.0000         0    0.1472   -0.1444
         0    1.0000         0   -0.0611    0.0222
         0    0.0000    1.0000   -0.0194    0.1889

  Column 6

    0.0639
    0.1056
   -0.1028

the solution calculated by using LU factorization is
  Columns 1 through 5

    1.0000         0         0    0.1472   -0.1444
         0    1.0000         0   -0.0611    0.0222
         0         0    1.0000   -0.0194    0.1889

  Column 6

    0.0639
    0.1056
   -0.1028

The solutions calculated by different methods match
```

%%% BONUS %%%

The left matrix is an identity matrix of size 3, otherwise notated as eye(3).

```
if  ~any(closetozeroroundoff(X(:,1:3)-eye(3),7),"all")
    disp('The left X matrix and the identity matrix are the same!')
end
```

```
The left X matrix and the identity matrix are the same!
```

The right matrix is the inverse of A.

```
if  ~any(closetozeroroundoff(X(:,4:6)-inv(A),7),"all")
    disp('The right X matrix and the inverse matrix of A are the same!')
end
```

```
The right X matrix and the inverse matrix of A are the same!
```

Exercise 6

```
type markov.m
```

```
function q = markov(P, x0)
format
[n, ~] = size(P);
q = [];

% Set a conditional statement to check if the given matrix P,
% which will have positive entries, is stochastic

% Checking if P is left stochastic by making sure the column sums equal 1

for i = 1 : length(P)
    x(i) = sum(P(:, i));
end

b2 = sum(x);

% Making sure matrix P's entries are positive

b1 = any(any(P > 0));

% Matrix P is stochastic

if b1 == 1 && b2 == length(P)
    disp('P is a stochastic matrix')

    % Output the unique steady-state vector q
    % Employ here a MATLAB command null(,'r')
    % to output a basis for the Null space

    Q = null(P - eye(n), 'r');
    s = sum(Q);

    % Scale the vector in the basis to get
    % the required probability vector q

    q = 1 / s * Q;
    disp('the steady-state vector is')
    q

    % Matrix P is not stochastic

else
    disp('P is not a stochastic matrix')

    % The empty output for q will stay

    q = [];

    return
end

% Verifying that the Markov chain converges to q

k = 0;

% Setting up a “while” loop and assigning each consecutive
% iteration to x0 again

while any(closetozeroroundoff(x0-q,7))
    x1 = P * x0;
```

```
    x0 = x1;
    k = k + 1;
end

% Displaying the number of iterations

fprintf('number of iterations to archive required accuracy is %i\n',k)

end
```

```
type closetozeroroundoff.m
```

```
function B = closetozeroroundoff(A, p)
A(abs(A) < 10 ^ -p) = 0;
B = A;
end
```

```
%(a)
P=[.6 .3;.5 .7], x0=[.4;.6]
```

```
P = 2×2
    0.6000    0.3000
    0.5000    0.7000
x0 = 2×1
    0.4000
    0.6000
```

```
q=markov(P,x0);
```

```
P is not a stochastic matrix
```

```
%(b)
P=[.5 .3;.5 .7],x0=[.5;.5]
```

```
P = 2×2
    0.5000    0.3000
    0.5000    0.7000
x0 = 2×1
    0.5000
    0.5000
```

```
q=markov(P,x0);
```

```
P is a stochastic matrix
the steady-state vector is
q = 2×1
    0.3750
    0.6250
number of iterations to archive required accuracy is 9
```

```
%(c)
P=[.9 .2;.1 .8], x0=[.11;.89]
```

```
P = 2×2
    0.9000    0.2000
    0.1000    0.8000
x0 = 2×1
    0.1100
    0.8900
```

```
q=markov(P,x0);
```

```
P is a stochastic matrix
the steady-state vector is
q = 2×1
    0.6667
    0.3333
number of iterations to archive required accuracy is 44
```

```
%(d)
P=[.9 .2;.1 .8], x0=[.90;.10]
```

```
P = 2×2
    0.9000    0.2000
    0.1000    0.8000
x0 = 2×1
    0.9000
    0.1000
```

```
q=markov(P,x0);
```

```
P is a stochastic matrix
the steady-state vector is
q = 2×1
    0.6667
    0.3333
number of iterations to archive required accuracy is 42
```

```
%(e)
P=[.90 .01 .09;.01 .90 .01;.09 .09 .90], x0=[.5; .3; .2]
```

```
P = 3×3
    0.9000    0.0100    0.0900
    0.0100    0.9000    0.0100
    0.0900    0.0900    0.9000
x0 = 3×1
    0.5000
    0.3000
    0.2000
```

```
q=markov(P,x0);
```

```
P is a stochastic matrix
the steady-state vector is
q = 3×1
    0.4354
    0.0909
    0.4737
number of iterations to archive required accuracy is 125
```

```
%(f)
P=magic(5); P=P./sum(P), x0=randi(10,5,1);x0=x0./sum(x0)
```

```
P = 5×5
    0.2615    0.3692    0.0154    0.1231    0.2308
    0.3538    0.0769    0.1077    0.2154    0.2462
    0.0615    0.0923    0.2000    0.3077    0.3385
    0.1538    0.1846    0.2923    0.3231    0.0462
    0.1692    0.2769    0.3846    0.0308    0.1385
x0 = 5×1
    0.1290
    0.1935
```

```
    0.2258
    0.1613
    0.2903
```

```
q=markov(P,x0);
```

```
P is a stochastic matrix
the steady-state vector is
q = 5×1
    0.2000
    0.2000
    0.2000
    0.2000
    0.2000
number of iterations to archive required accuracy is 12
```

```
%(g)
x0=q
```

```
x0 = 5×1
    0.2000
    0.2000
    0.2000
    0.2000
    0.2000
```

```
q=markov(P,x0);
```

```
P is a stochastic matrix
the steady-state vector is
q = 5×1
    0.2000
    0.2000
    0.2000
    0.2000
    0.2000
number of iterations to archive required accuracy is 0
```

Exercise 7

```
type transf_1.m
```

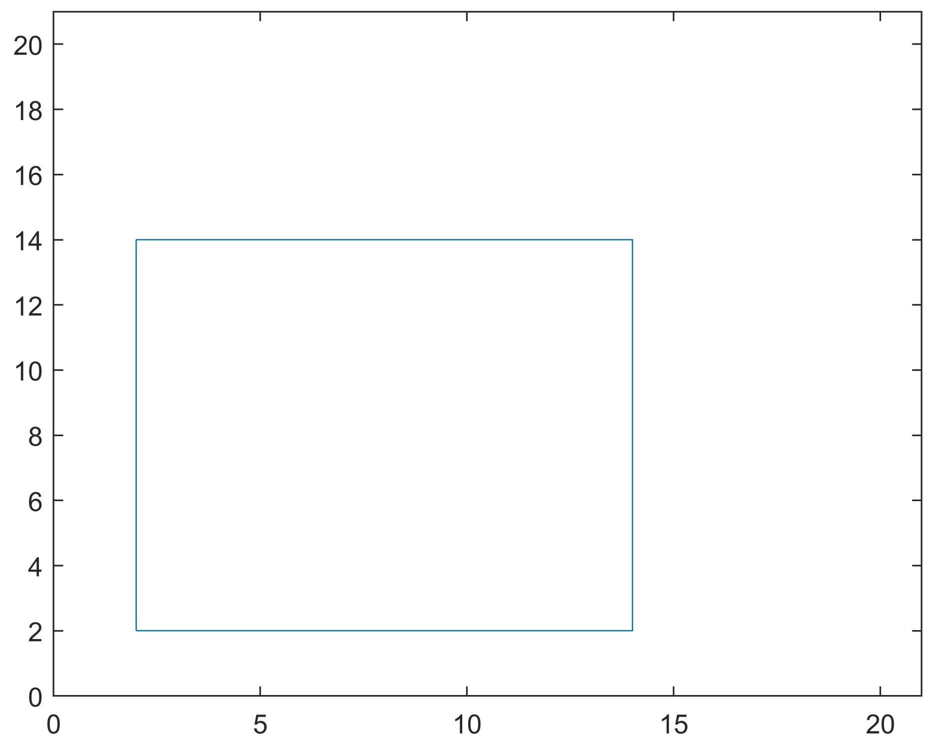
```
function C = transf_1(A, E)
C = A * E;
x = C(1, :); y = C(2, :);
plot(x, y)
v=[0 21 0 21];
axis(v)
end
```

```
type polygon.m
```

```
function Area = polygon(E)
A = eye(2);
C = transf_1(A, E);
n = size(E, 2) - 1;

% Outputting of the area of a polygon by using formula (1)
i = 1;
syms k;
Area = abs(0.5 * symsum(E(1, i) * E(2, i + 1) - E(2, i) * E(1, i + 1), k, i, n));
end
```

```
%(a)
E=[2 14 14 2 2;2 2 14 14 2];
Area = polygon(E)
```



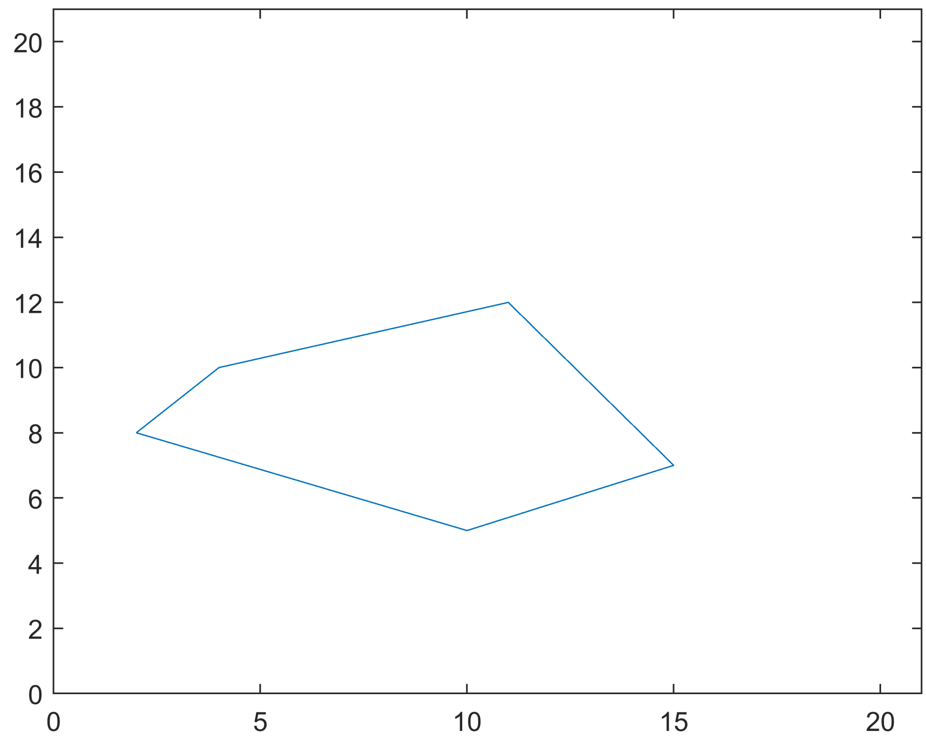
Area = 48

```
%(b)
```

```
E=[2 10 15 11 4 2;8 5 7 12 10 8]
```

```
E = 2×6
     2    10    15    11     4     2
     8     5     7    12    10     8
```

```
Area = polygon(E)
```



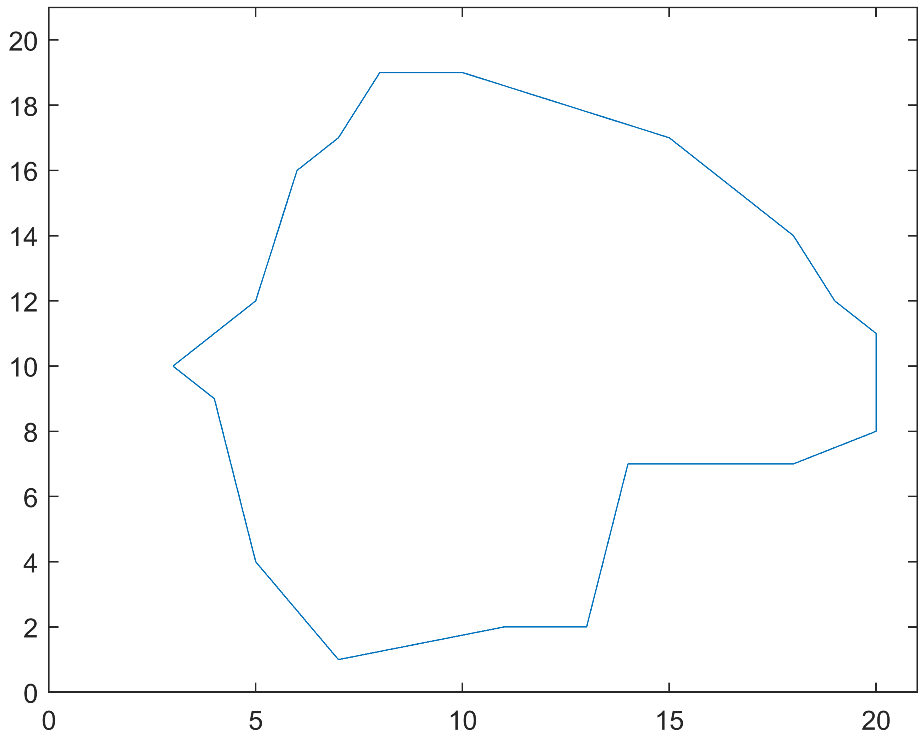
Area = 175

```
%(c)
A1=randi(10,1,5);
A1=sort(unique(A1), 'ascend');
B1=randi(10,1,size(A1,2));
B1=sort(B1,'descend');
A2=randi([11 20],1,5);
A2=sort(unique(A2), 'ascend');
B2=randi(10,1,size(A2,2));
B2=sort(B2,'ascend');
A3=randi([11 20],1,5);
A3=sort(unique(A3), 'descend');
B3=randi([11 20],1,size(A3,2));
B3=sort(B3,'ascend');
A4=randi(10,1,5);
A4=sort(unique(A4), 'descend');
B4=randi([11 20],1,size(A4,2));
B4=sort(B4,'descend');
E=[A1 A2 A3 A4 A1(1,1);B1 B2 B3 B4 B1(1,1)]
```

```
E = 2×19
```

```
     3     4     5     7    11    13    14    18    20    20    19    18    15 ...
    10     9     4     1     2     2     7     7     8    11    12    14    17
```

```
Area = polygon(E)
```



```
Area = 117
```