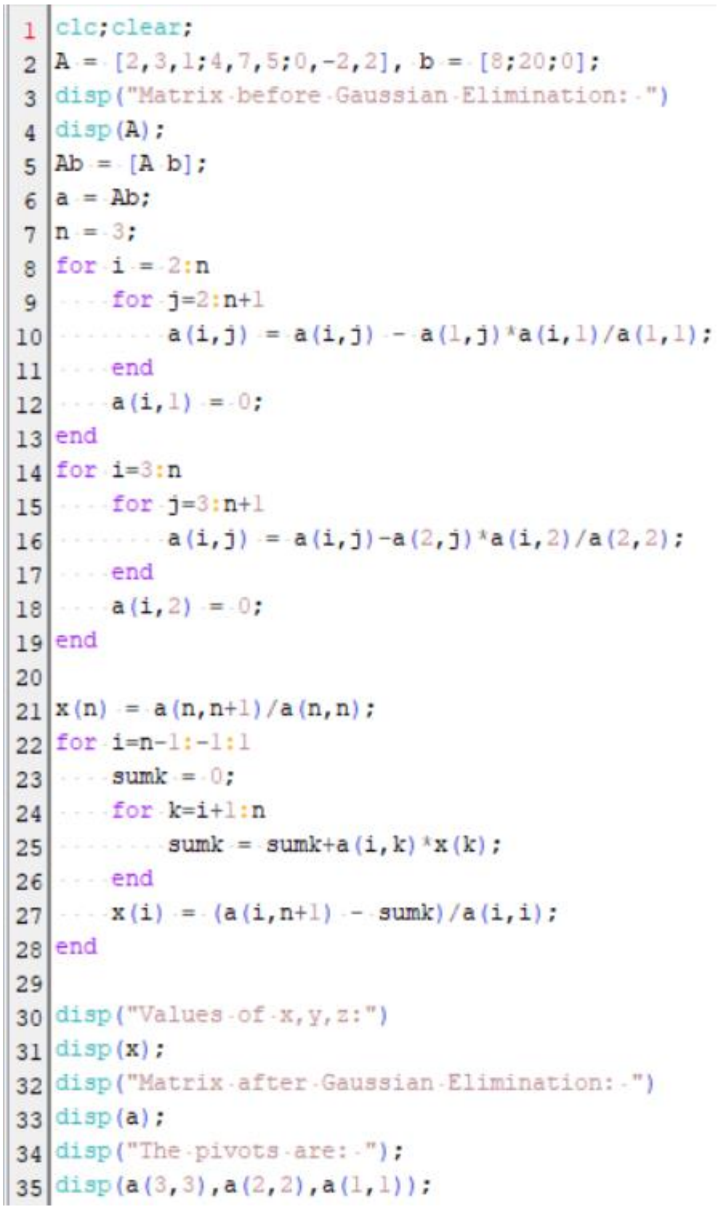
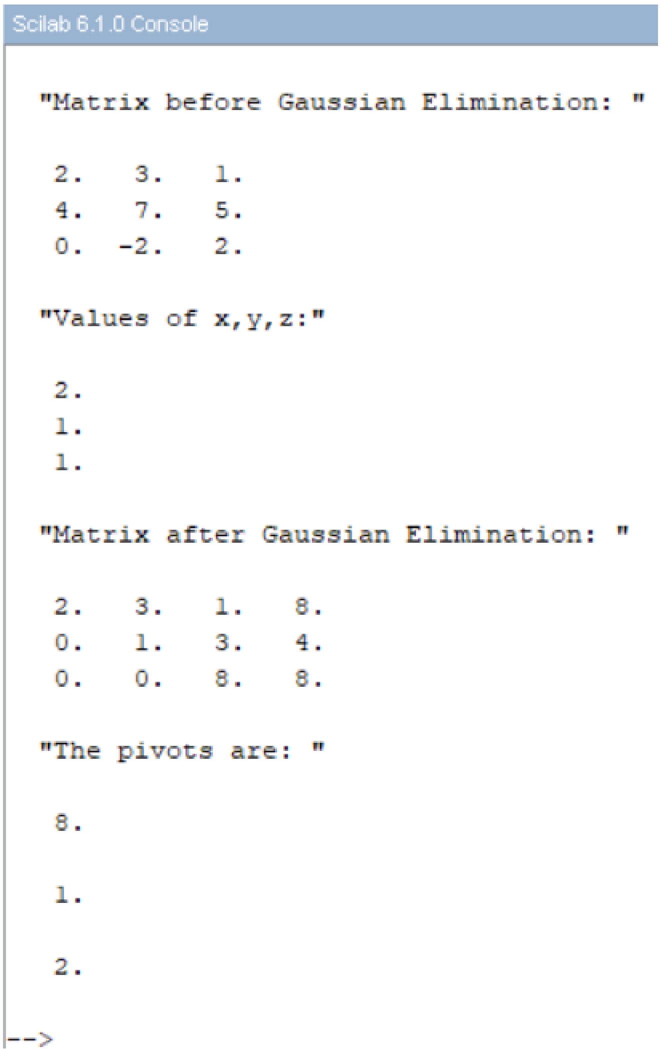
Program 1 – Gaussian Elimination

Code:



```
clc;clear;
A = [2,3,1;4,7,5;0,-2,2], b = [8;20;0];
disp("Matrix before Gaussian Elimination: ")
disp(A);
Ab = [A b];
a = Ab;
n = 3;
for i = 2:n
    for j=2:n+1
        a(i,j) = a(i,j) - a(1,j)*a(i,1)/a(1,1);
    end
    a(i,1) = 0;
end
for i=3:n
    for j=3:n+1
        a(i,j) = a(i,j)-a(2,j)*a(i,2)/a(2,2);
    end
    a(i,2) = 0;
end

x(n) = a(n,n+1)/a(n,n);
for i=n-1:-1:1
    sumk = 0;
    for k=i+1:n
        sumk = sumk+a(i,k)*x(k);
    end
    x(i) = (a(i,n+1) - sumk)/a(i,i);
end

disp("Values of x,y,z:")
disp(x);
disp("Matrix after Gaussian Elimination: ")
disp(a);
disp("The pivots are: ");
disp(a(3,3),a(2,2),a(1,1));
```

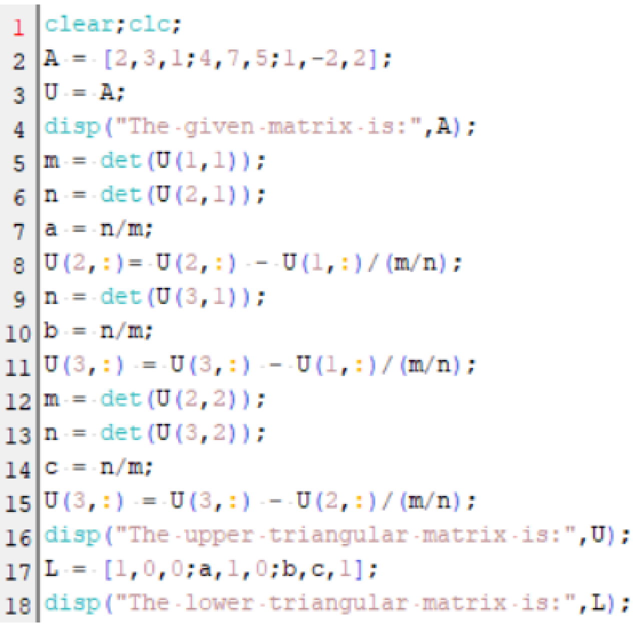
Output:



```
 "Matrix before Gaussian Elimination: "

  2.   3.   1.
  4.   7.   5.
  0.  -2.   2.

 "Values of x,y,z:"

  2.
  1.
  1.

 "Matrix after Gaussian Elimination: "

  2.   3.   1.   8.
  0.   1.   3.   4.
  0.   0.   8.   8.

 "The pivots are: "

  8.

  1.

  2.

-->
```

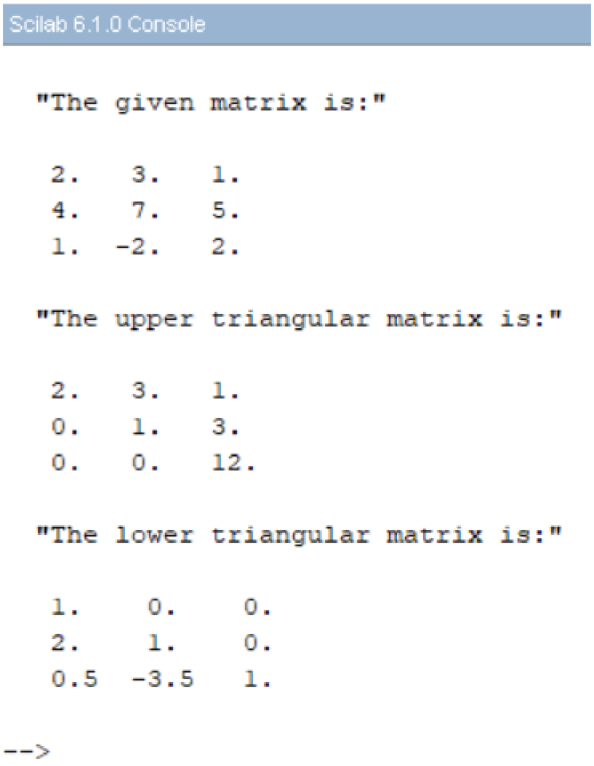
Program 2 – LU Decomposition

Code:



```
clear;clc;
A = [2,3,1;4,7,5;1,-2,2];
U = A;
disp("The given matrix is:",A);
m = det(U(1,1));
n = det(U(2,1));
a = n/m;
U(2,:)= U(2,:) - U(1,:)/(m/n);
n = det(U(3,1));
b = n/m;
U(3,:) = U(3,:) - U(1,:)/(m/n);
m = det(U(2,2));
n = det(U(3,2));
c = n/m;
U(3,:) = U(3,:) - U(2,:)/(m/n);
disp("The upper triangular matrix is:",U);
L = [1,0,0;a,1,0;b,c,1];
disp("The lower triangular matrix is:",L);
```

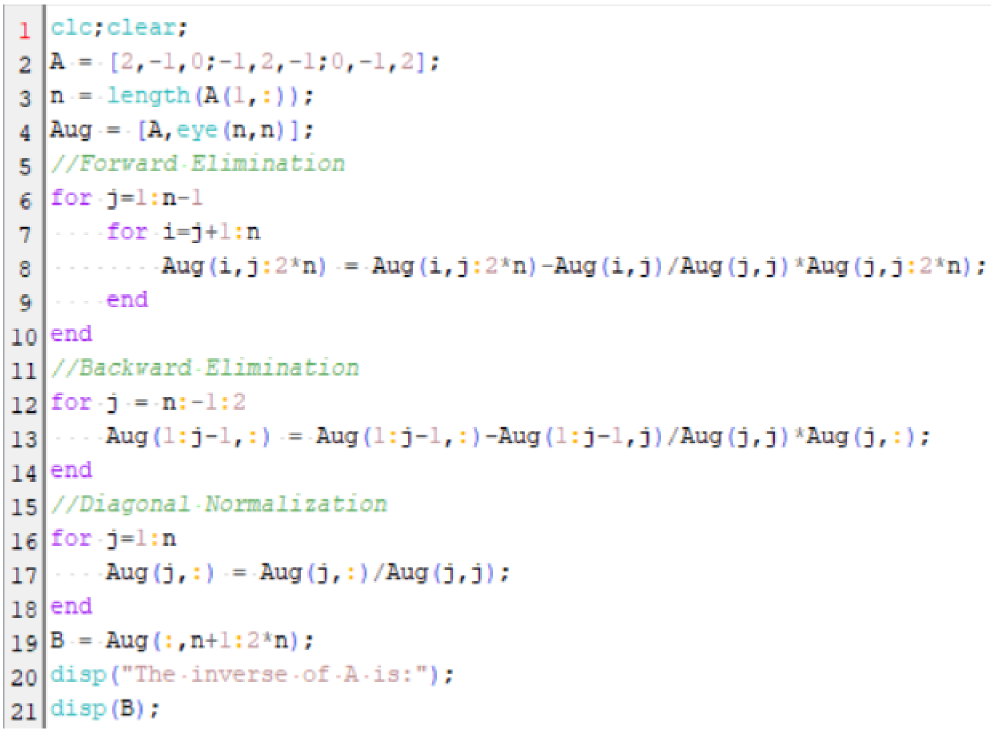
Output:



```
  "The given matrix is:"

   2.   3.   1.
   4.   7.   5.
   1.  -2.   2.

  "The upper triangular matrix is:"

   2.   3.   1.
   0.   1.   3.
   0.   0.   12.

  "The lower triangular matrix is:"

   1.     0.     0.
   2.     1.     0.
   0.5   -3.5    1.

-->
```

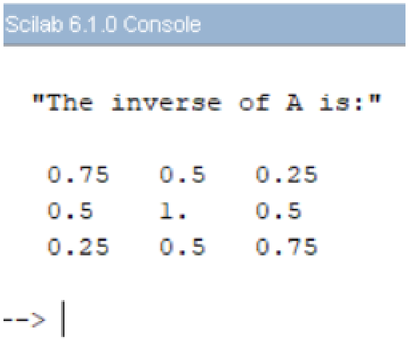
Program 3 – Gauss-Jordan Inverse

Code:



```
clc;clear;
A = [2,-1,0;-1,2,-1;0,-1,2];
n = length(A(1,:));
Aug = [A,eye(n,n)];
//Forward Elimination
for j=1:n-1
    for i=j+1:n
        Aug(i,j:2*n) = Aug(i,j:2*n)-Aug(i,j)/Aug(j,j)*Aug(j,j:2*n);
    end
end
//Backward Elimination
for j = n:-1:2
    Aug(1:j-1,:) = Aug(1:j-1,:)-Aug(1:j-1,j)/Aug(j,j)*Aug(j,:);
end
//Diagonal Normalization
for j=1:n
    Aug(j,:) = Aug(j,:)/Aug(j,j);
end
B = Aug(:,n+1:2*n);
disp("The inverse of A is:");
disp(B);
```

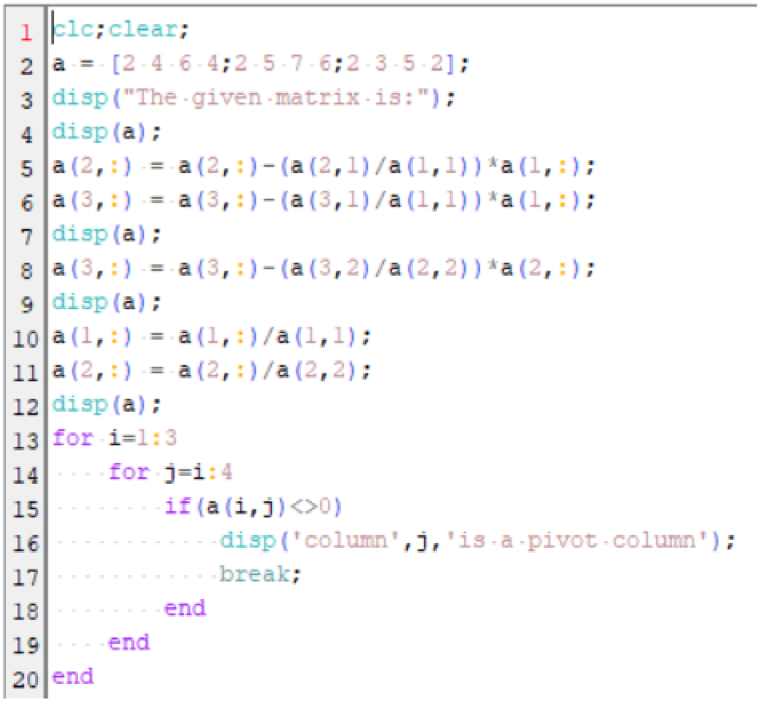
Output:



```
  "The inverse of A is:"

   0.75   0.5   0.25
   0.5    1.    0.5 
   0.25   0.5   0.75

-->
```

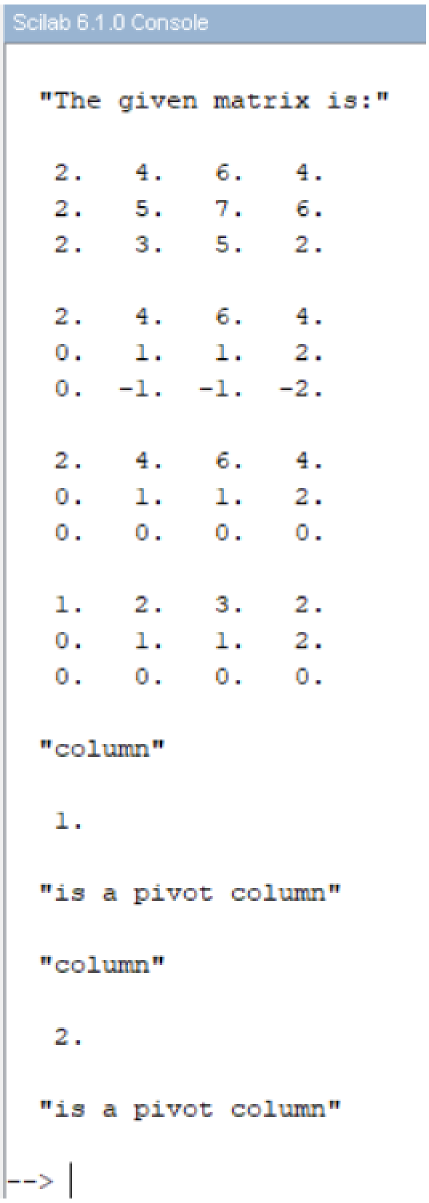
Program 4 – Span of Column Space

Code:



```
clc;clear;
a = [2 4 6 4;2 5 7 6;2 3 5 2];
disp("The given matrix is:");
disp(a);
a(2,:) = a(2,:)-(a(2,1)/a(1,1))*a(1,:);
a(3,:) = a(3,:)-(a(3,1)/a(1,1))*a(1,:);
disp(a);
a(3,:) = a(3,:)-(a(3,2)/a(2,2))*a(2,:);
disp(a);
a(1,:) = a(1,:)/a(1,1);
a(2,:) = a(2,:)/a(2,2);
disp(a);
for i=1:3
    for j=i:4
        if(a(i,j)<>0)
            disp('column',j,'is a pivot column');
            break;
        end
    end
end
```

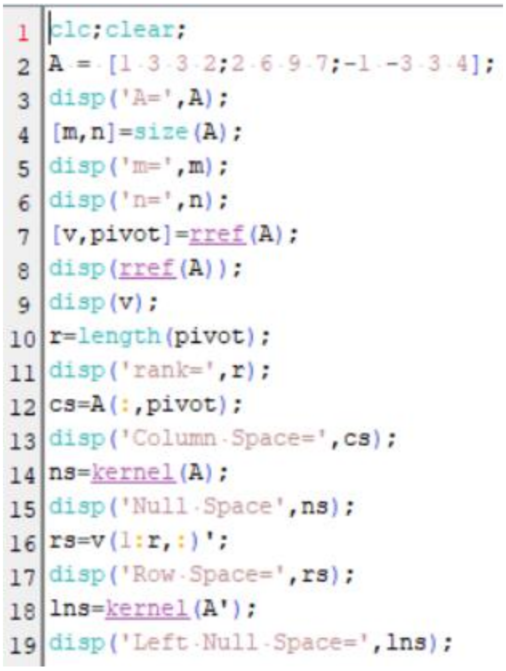
Output:



```
 "The given matrix is:"

  2.   4.   6.   4.
  2.   5.   7.   6.
  2.   3.   5.   2.

  2.   4.   6.   4.
  0.   1.   1.   2.
  0.  -1.  -1.  -2.

  2.   4.   6.   4.
  0.   1.   1.   2.
  0.   0.   0.   0.

  1.   2.   3.   2.
  0.   1.   1.   2.
  0.   0.   0.   0.

 "column"

  1.

 "is a pivot column"

 "column"

  2.

 "is a pivot column"

--> 
```

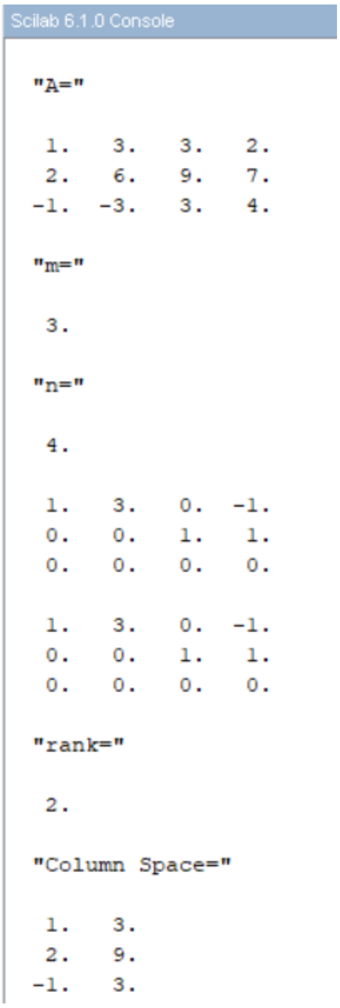
Program 5 – The Four Fundamental Subspaces

Code:

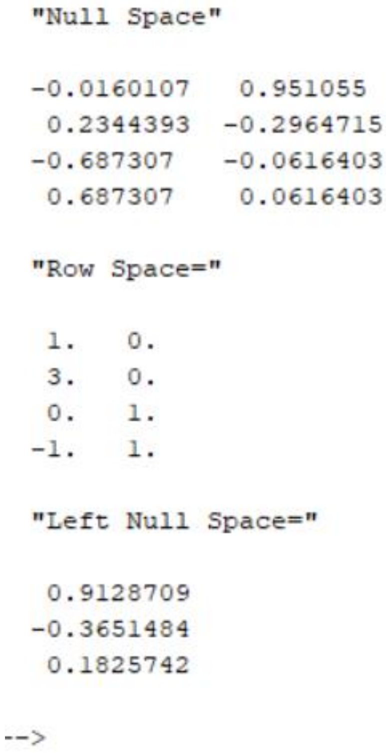


```
clc;clear;
A = [1 3 3 2;2 6 9 7;-1 -3 3 4];
disp('A=',A);
[m,n]=size(A);
disp('m=',m);
disp('n=',n);
[v,pivot]=rref(A);
disp(rref(A));
disp(v);
r=length(pivot);
disp('rank=',r);
cs=A(:,pivot);
disp('Column Space=',cs);
ns=kernel(A);
disp('Null Space',ns);
rs=v(1:r,:)';
disp('Row Space=',rs);
lns=kernel(A');
disp('Left Null Space=',lns);
```

Output:



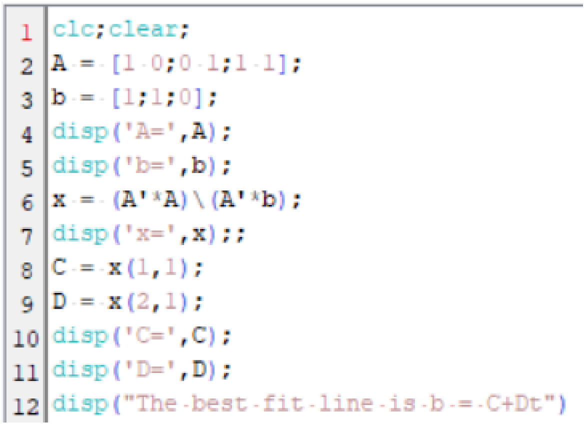
```
 "A="

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

 "m="

  3.

 "n="

  4.

  1.   3.   0.  -1.
  0.   0.   1.   1.
  0.   0.   0.   0.

  1.   3.   0.  -1.
  0.   0.   1.   1.
  0.   0.   0.   0.

 "rank="

  2.

 "Column Space="

  1.   3.
  2.   9.
 -1.   3.
```



```
 "Null Space"

 -0.0160107   0.951055
  0.2344393  -0.2964715
 -0.687307   -0.0616403
  0.687307    0.0616403

 "Row Space="

  1.   0.
  3.   0.
  0.   1.
 -1.   1.

 "Left Null Space="

  0.9128709
 -0.3651484
  0.1825742

-->
```

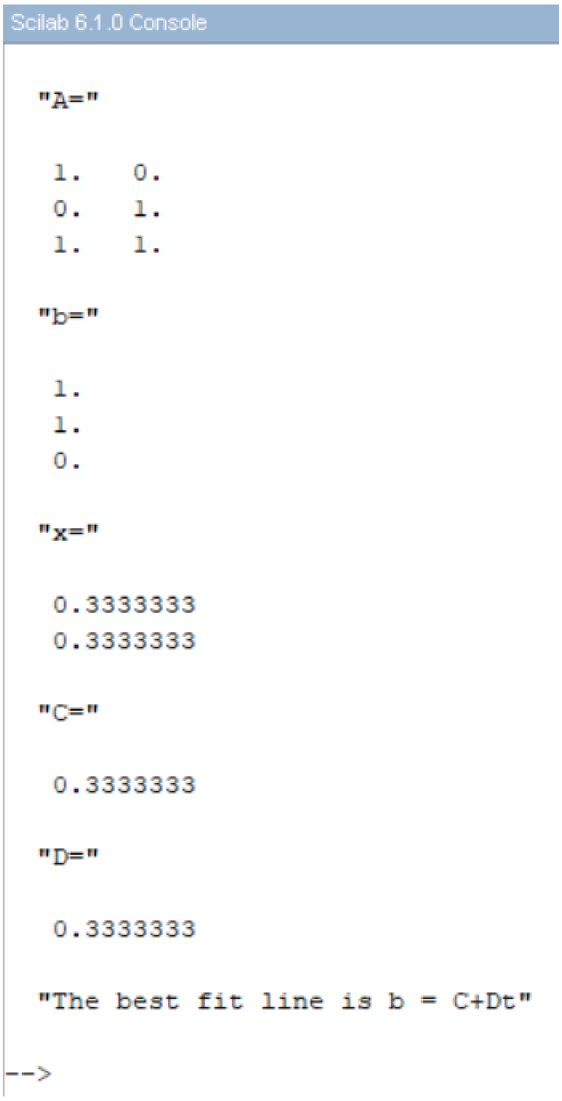
Program 6 – Projection by Least Squares

Code:



```
clc;clear;
A = [1 0;0 1;1 1];
b = [1;1;0];
disp('A=',A);
disp('b=',b);
x = (A'*A)\(A'*b);
disp('x=',x);;
C = x(1,1);
D = x(2,1);
disp('C=',C);
disp('D=',D);
disp("The best fit line is b = C+Dt")
```

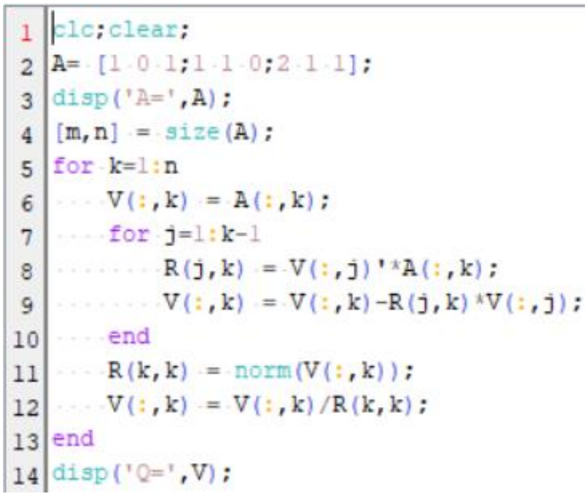
Output:



```
 "A="

  1.   0.
  0.   1.
  1.   1.

 "b="

  1.
  1.
  0.

 "x="

  0.3333333
  0.3333333

 "C="

  0.3333333

 "D="

  0.3333333

 "The best fit line is b = C+Dt"

-->
```

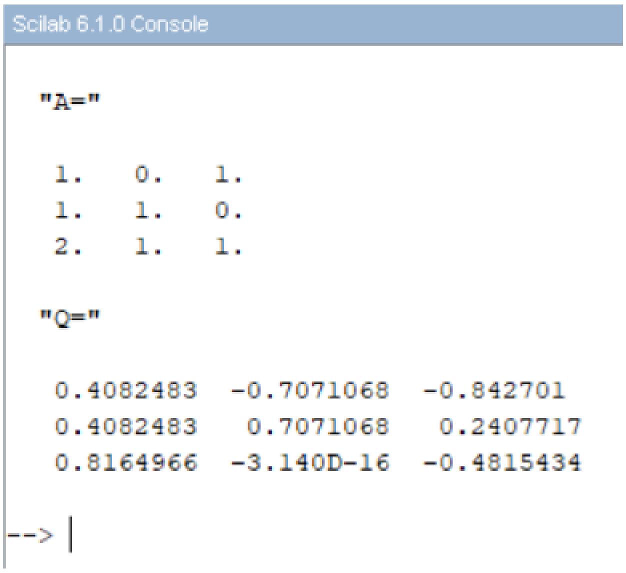
Program 7 – Gram-Schmidt Orthogonalization

Code:



```
clc;clear;
A= [1 0 1;1 1 0;2 1 1];
disp('A=',A);
[m,n] = size(A);
for k=1:n
    V(:,k) = A(:,k);
    for j=1:k-1
        R(j,k) = V(:,j)'*A(:,k);
        V(:,k) = V(:,k)-R(j,k)*V(:,j);
    end
    R(k,k) = norm(V(:,k));
    V(:,k) = V(:,k)/R(k,k);
end
disp('Q=',V);
```

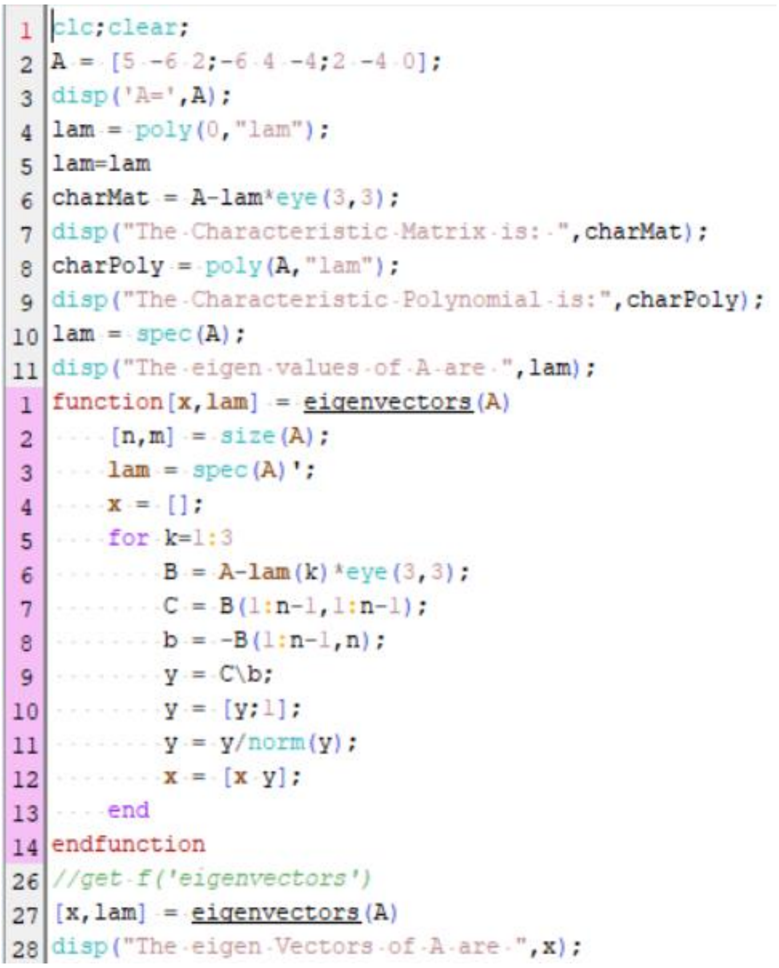
Output:



```
  "A="


   1.   0.   1.
   1.   1.   0.
   2.   1.   1.

  "Q="


   0.4082483  -0.7071068  -0.842701
   0.4082483   0.7071068   0.2407717
   0.8164966  -3.140D-16  -0.4815434

-->
```

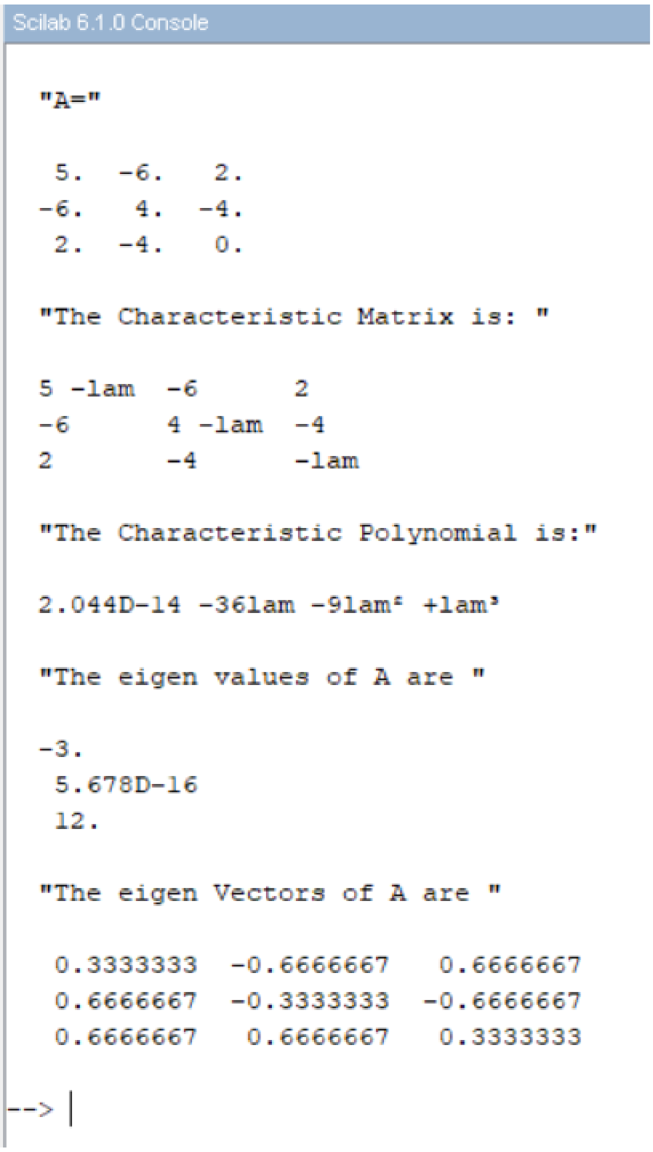
Program 8 – Eigen Values and Eigen Vectors

Code:



```
clc;clear;
A = [5 -6 2;-6 4 -4;2 -4 0];
disp('A=',A);
lam = poly(0,"lam");
lam=lam
charMat = A-lam*eye(3,3);
disp("The Characteristic Matrix is: ",charMat);
charPoly = poly(A,"lam");
disp("The Characteristic Polynomial is:",charPoly);
lam = spec(A);
disp("The eigen values of A are ",lam);
function[x,lam] = eigenvectors(A)
    [n,m] = size(A);
    lam = spec(A)';
    x = [];
    for k=1:3
        B = A-lam(k)*eye(3,3);
        C = B(1:n-1,1:n-1);
        b = -B(1:n-1,n);
        y = C\b;
        y = [y;1];
        y = y/norm(y);
        x = [x y];
    end
endfunction
//get f('eigenvectors')
[x,lam] = eigenvectors(A)
disp("The eigen Vectors of A are ",x);
```

Output:



```
 "A="

  5.  -6.   2.
 -6.   4.  -4.
  2.  -4.   0.

 "The Characteristic Matrix is: "

 5 -lam  -6       2
 -6       4 -lam  -4
 2        -4      -lam

 "The Characteristic Polynomial is:"

 2.044D-14 -36lam -9lam² +lam³

 "The eigen values of A are "

 -3.
  5.678D-16
  12.

 "The eigen Vectors of A are "

  0.3333333  -0.6666667   0.6666667
  0.6666667  -0.3333333  -0.6666667
  0.6666667   0.6666667   0.3333333

-->
```