Name	Manish Shashikant Jadhav
UID no.	2023301005
Subject	Linear Algebra
Department	Computer Engineering-B

Experiment 1	
AIM :	Implementation of Basic Commands and Operations on Matrix
Code and Output	1. **Basic Scilab Commands:** clc; printf("Display identity matrix of order 3:"); disp(eye(3,3)); printf("Display matrix with all elements ONE of order 3:"); disp(ones(3,3)); printf("Display given matrix A:"); A=[1 2 3;4 5 6;7 8 0] disp(A); printf("Display random matrix of order 3:"); disp(rand(3,3)); printf("Display lower triangular matrix from A:"); disp(tril(A)); printf("Display upper triangular matrix from A:"); disp(triu(A)); printf("Display transpose of A:") disp(A');

	printf("Display size of matrix A:"); disp(size(A)); printf("Display A33 element of A:"); disp(A(3,3)); printf("Display 2nd column of A:"); disp(A(:,2)); printf("Display 3rd row of A:"); disp(A(3,:)); printf("Display sum of all elements of A:"); disp(sum(A)); printf("Display product of all elements of A:"); disp(prod(A)); printf("Display sum of elements of 2nd column in matrix A:"); disp(sum(A(:,2))); printf("Display product of elements of 2nd column in matrix A:"); disp(prod(A(:,2))); printf("Display sum of elements of 3rd row in matrix A:"); disp(sum(A(3,:))); printf("Display product of elements of 3rd row in matrix A:"); disp(prod(A(3,:))); printf("Display sum of all columns in order in matrix A:"); disp(sum(A,'r')); printf("Display product of all columns in order in matrix A:"); disp(prod(A,'r'));

	printf("Display sum of all rows in order in matrix A:"); disp(sum(A,'c')); printf("Display product of all rows in order in matrix A:"); disp(prod(A,'c')); printf("Display imaginary part of matrix A:"); disp(imag(A)); printf("Display real part of matrix A:"); disp(real(A)); printf("Display inverse of matrix A:"); disp(inv(A)); printf("Display determinant of matrix A:"); disp(det(A)); printf("Display trace of matrix A:"); disp(trace(A)); printf("Display rank of matrix A:"); disp(rank(A)); printf("Display diagonal matrix A:"); disp(eye(3,3).*A); printf("Display only diagonal elements of matrix A:"); disp(diag(A)); printf("Display conjugate of matrix A:"); disp(conj(A));

	Scilab 6.0.2 Console ``` Display identity matrix of order 3: 1. 0. 0. 0. 1. 0. 0. 0. 1. Display matrix with all elements ONE of order 3: 1. 1. 1. 1. 1. 1. 1. 1. 1. Display given matrix A: 1. 2. 3. 4. 5. 6. 7. 8. 0. Display random matrix of order 3: 0.1280058 0.1121355 0.6970851 0.7783129 0.6856896 0.8415518 0.211903 0.1531217 0.4062025 Display lower triangular matrix from A: 1. 0. 0. 4. 5. 0. 7. 8. 0. Display upper triangular matrix from A: 1. 2. 3. 0. 5. 6. 0. 0. 0. Display transpose of A: 1. 4. 7. 2. 5. 8. 3. 6. 0. Display size of matrix A: 3. 3. Display A33 element of A: 0. Display 2nd column of A: 2. 5. 8. Display 3rd row of A: 7. 8. 0. Display sum of all elements of A: 36. ```

	Scilab 6.0.2 Console

```
Display product of all elements of A:
   0.
Display sum of elements of 2nd column in matrix A:
   15.
Display product of elements of 2nd column in matrix A:
   80.
Display sum of elements of 3rd row in matrix A:
   15.
Display product of elements of 3rd row in matrix A:
   0.
Display sum of all columns in order in matrix A:
   12.   15.   9.
Display product of all columns in order in matrix A:
   28.   80.   0.
Display sum of all rows in order in matrix A:
   6.
   15.
   15.
Display product of all rows in order in matrix A:
   6.
   120.
   0.
Display imaginary part of matrix A:
   0.   0.   0.
   0.   0.   0.
   0.   0.   0.
Display real part of matrix A:
   1.   2.   3.
   4.   5.   6.
   7.   8.   0.
Display inverse of matrix A:
  -1.7777778   0.8888889  -0.1111111
   1.5555556  -0.7777778   0.2222222
  -0.1111111   0.2222222  -0.1111111
Display determinant of matrix A:
   27.
Display trace of matrix A:
   6.
Display rank of matrix A:
   3.
Display diagonal matrix A:
   1.   0.   0.
   0.   5.   0.
   0.   0.   0.
Display only diagonal elements of matrix A:
   1.
   5.
   0.
Display conjugate of matrix A:
   1.   2.   3.
   4.   5.   6.
   7.   8.   0.

-->
```

2. Exercise:

```
C=rand(4,4);
printf("The random generated matrix C is: ");
disp(C);

sum_first_column = sum(C(:,1));
printf("Sum of first column elements: ");
disp(sum_first_column);

product_second_row = prod(C(2,:));
printf("Product of second row elements: ");
disp(product_second_row);

sum_matrix = sum(C);
printf("Sum of all elements: ");
disp(sum_matrix);

determinant = det(C);
printf("Determinant of Matrix A: ");
disp(determinant);

trace_matrix = trace(C);
printf("Trace of Matrix A: ");
disp(trace_matrix);
```

```
Scilab 6.0.2 Console

--> exec('C:\Users\Admin\Documents\exercise1.sce', -1)
The random generated matrix C is:
   0.8433565   0.1867539   0.2124056   0.9110545
   0.0748595   0.4920584   0.579502    0.8082667
   0.8532815   0.7489608   0.2628148   0.8102653
   0.012459    0.9414957   0.4360987   0.2590428
Sum of first column elements:
   1.7839565
Product of second row elements:
   0.0172533
Sum of all elements:
   8.4326757
Determinant of Matrix A:
   0.0307057
Trace of Matrix A:
   1.8572725

-->
```

3. **Code:**

```
A=[1 2+%i 4; 3-4*%i 9 -2; 2 -5 1-%i]
disp(A);
printf("Display real part of matrix A:");
disp(real(A));
printf("Display imaginary part of matrix A:");
disp(imag(A));

printf("Display random matrix of order 3 with elements from 0 to 9:")
disp(rand(3,3)*10);

printf("Display random matrix of order 3 with integer elements from 0 to 9:")
disp(int(rand(3,3)*10));

B=[1 3 5; 2 4 1; 1 2 3]
printf("Display matrix B:")
disp(B);

printf("Display reduced row echelon form of B:");
```

	disp(rref(B)); printf("Display multiplication of A & B: "); disp(A*B); printf("Display reciprocal of elements in B: ") disp(1./B); printf("Display square root of 25:"); disp(sqrt(25)); printf("Display the sine of pi/2"); disp(sin(%pi/2)); printf("Display the given value of x:"); x= 3^2; disp(x); printf("Display reciprocal of given value"); disp(1/x);

Output:

```
Scilab 6.0.2 Console

--> exec('C:\Users\Admin\Manish\Prac2\prac2.sce', -1)

   1.          2. + i     4.
   3. - 4.i    9.        -2.
   2.         -5.         1. - i
Display real part of matrix A:
   1.   2.   4.
   3.   9.  -2.
   2.  -5.   1.
Display imaginary part of matrix A:
   0.   1.   0.
  -4.   0.   0.
   0.   0.  -1.
Display random matrix of order 3 with elements from 0 to 9:
   5.7614598   1.2326993   5.7694048
   7.1491304   2.8655522   3.9386961
   9.321636    0.1247996   6.8885837
Display random matrix of order 3 with integer elements from 0 to 9:
   9.   8.   5.
   8.   1.   9.
   3.   5.   9.
Display matrix B:
   1.   3.   5.
   2.   4.   1.
   1.   2.   3.
Display reduced row echelon form of B:
   1.   0.   0.
   0.   1.   0.
   0.   0.   1.
Display multiplication of A & B:
   9. + 2.i    19. + 4.i    19. + i
   19. - 4.i   41. - 12.i   18. - 20.i
  -7. - i     -12. - 2.i    8. - 3.i
Display reciprocal of elements in B:
   1.    0.3333333   0.2
   0.5   0.25        1.
   1.    0.5         0.3333333
Display square root of 25:
   5.
Display the sine of pi/2
   1.
Display the given value of x:
   9.
Display reciprocal of given value
   0.1111111

-->
```

CONCLUSION: Hence, by completing this experiment I came to know about Implementation of Basic Commands and Operations on Matrix

Name	Manish Shashikant Jadhav
UID no.	2023301005
Subject	Linear Algebra
Department	Computer Engineering-B

Experiment 2	
AIM :	Implementation of Conditional Branching and Looping in Scilab.
Code 1:	**Write a program to find the Pythagorean triplet up to the number 50 or 20.** ```clc for i=1:50 for j=i+1:50 c=(i*i)+(j*j) for k=i+2:50 if c==(k*k) printf("\n") printf("Pythagorean Triplet is %d, %d, %d, \n", i,j,k); end end end end```

<table>
<tr>
<td></td>
<td>Scilab 6.0.2 Console

Pythagorean Triplet is 3, 4, 5,

Pythagorean Triplet is 5, 12, 13,

Pythagorean Triplet is 6, 8, 10,

Pythagorean Triplet is 7, 24, 25,

Pythagorean Triplet is 8, 15, 17,

Pythagorean Triplet is 9, 12, 15,

Pythagorean Triplet is 9, 40, 41,

Pythagorean Triplet is 10, 24, 26,

Pythagorean Triplet is 12, 16, 20,

Pythagorean Triplet is 12, 35, 37,

Pythagorean Triplet is 14, 48, 50,

Pythagorean Triplet is 15, 20, 25,

Pythagorean Triplet is 15, 36, 39,

Pythagorean Triplet is 16, 30, 34,

Pythagorean Triplet is 18, 24, 30,

Pythagorean Triplet is 20, 21, 29,

Pythagorean Triplet is 21, 28, 35,

Pythagorean Triplet is 24, 32, 40,

Pythagorean Triplet is 27, 36, 45,

Pythagorean Triplet is 30, 40, 50,</td>
</tr>
<tr>
<td>Code 2:</td>
<td>If Un=4(Un-1)+4 and U0=4 , Print 20th term of the Sequence:

clc
U=[4];
for n=1:19
   U(n+1)=4*U(n)+4;
end
disp(U)
printf("\n")
printf("the 20th term is: ")
disp(U(20))</td>
</tr>
</table>

```
Scilab 6.0.2 Console

   4.
   20.
   84.
   340.
   1364.
   5460.
   21844.
   87380.
   349524.
   1398100.
   5592404.
   22369620.
   89478484.
   3.579D+08
   1.432D+09
   5.727D+09
   2.291D+10
   9.163D+10
   3.665D+11
   1.466D+12

the 20th term is:
   1.466D+12

-->
```

Code 3:

Write a Scilab code to input a matrix and check whether the matrix is Symmetric, Skew Symmetric or none.

```
A = input("Enter the matrix: ");
[m, n] = size(A);
if m ~= n
   disp('Matrix must be square');
   return;
end
isSymmetric = isequal(A, A');
isSkewSymmetric = isequal(A, -A');

if isSymmetric
   disp('Matrix type: symmetric');
elseif isSkewSymmetric
   disp('Matrix type: skew-symmetric');
else
   disp('Matrix is neither symmetric nor skew-symmetric');
end
```

```
--> exec('D:\1.sce', -1)
Enter the matrix: [1 1 -1; 1 2 0; -1 0 5]


 Matrix type: symmetric

-->
```

Code 4:

Write a Scilab code to input a matrix and check whether the matrix is Hermitian or not

```
A = [1 2+3*%i 3-4*%i; 2-3*%i 5 6+7*%i; 3+4*%i 6-7*%i 8];

if A == A' then
    disp("The input matrix is Hermitian.");
else
    disp("The input matrix is not Hermitian.");
end
```

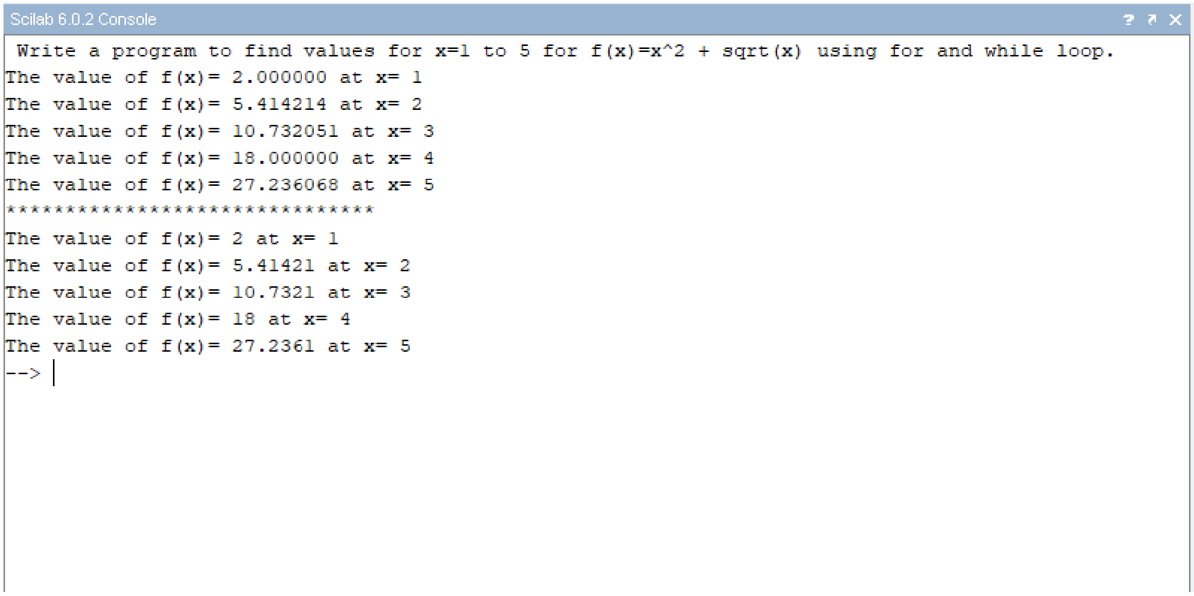
```
--> exec('D:\1.sce', -1)

 The input matrix is Hermitian.

-->
```

Code 5:

Write a Scilab code to input a matrix and check whether the matrix is invertible or not

```
B = [1,1 ; 2,2];
if B == ((-B)') then
    printf("It is Invertible \n");
else
    printf("It is not Invertible \n");
end

printf("\n");
```

	Startup execution: loading initial environment --> exec('D:\1.sce', -1) It is not Invertible -->
Code 6:	**Write a program to find values for x=1 to 5 for f(x)=x^2 +sqrt(x) using for loop and while loop** clc; printf("Write a program to find values for x=1 to 5 for f(x)=x^2 + sqrt(x) using for and while loop."); for x= 1:1:5 f(x)= x^2 + sqrt(x); printf('\nThe value of f(x)= %f at x= %i',f(x),x) end //using while loop printf("\n********************************"); x=1; while x<=5 f(x)= x^2 + sqrt(x); printf('\nThe value of f(x)= %g at x= %i',f(x),x) x=x+1; end



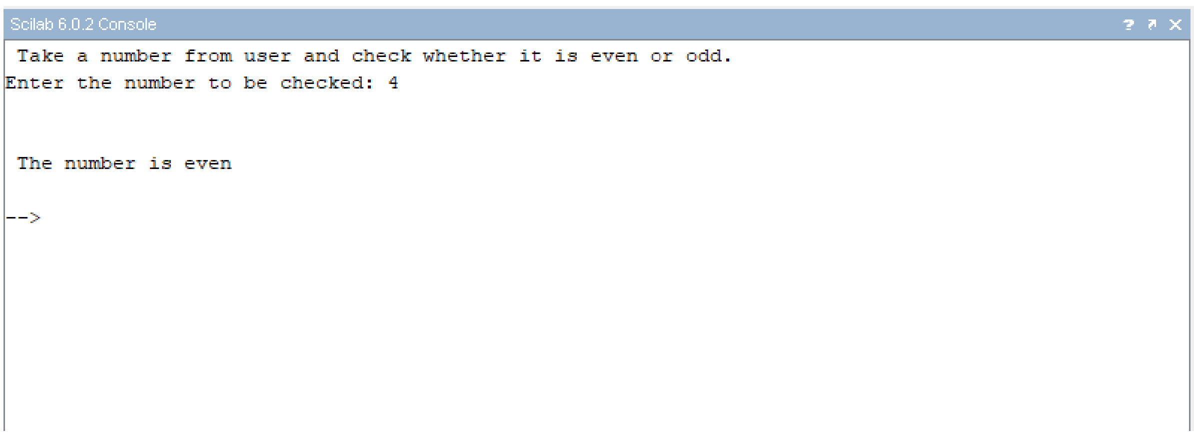
Code 7:

Display values from 10 to 20 using while loop

```
clc;
printf("Using while loop");
a=10;
while a<=20
printf("\nValue of a: %d",a);
a=a+1;
end
```

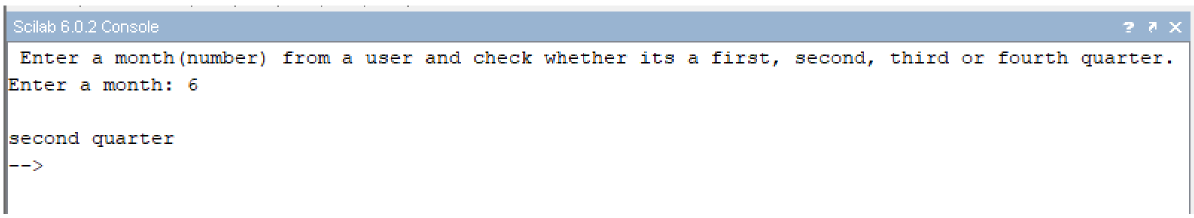
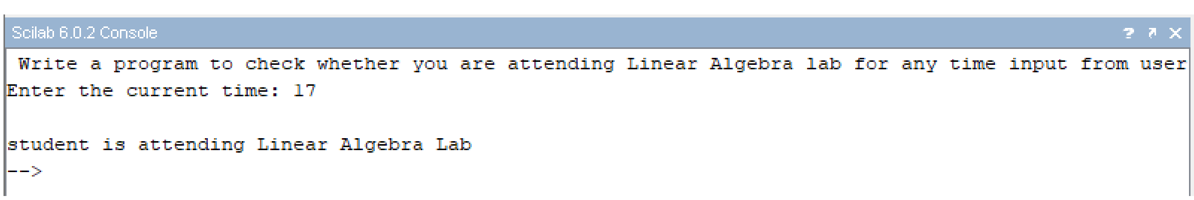
Code 8: **Take a number from user and check whether it is odd or even. (Use modulo command)**

```
clc;
printf("Take a number from user and check whether it is even or odd.");
x=input("Enter the number to be checked: ")
if modulo(x,2)==0 then
disp("The number is even") ;
else
disp("The number is odd");
end
```



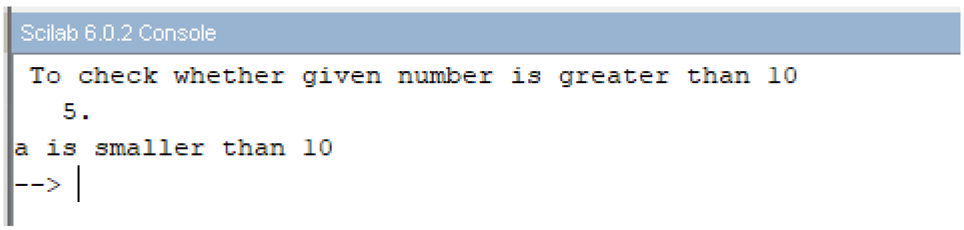
Code 9: **Enter a month (number) from a user and check whether it's a first, second, third and forth quarter**

```
clc;
printf("Enter a month(number) from a user and check whether its a first, second, third or fourth
quarter.");
month = input("Enter a month: ")
if month >=1 & month <=3
then
printf("first quarter");
elseif month >= 4 & month <= 6
then
printf("second quarter");
elseif month >= 7 & month <=9
then
printf("third quarter");
```

	elseif month >= 10 & month <=12 then printf("fourth quarter"); else printf("Invalid month"); end 
Code 10:	**Write a program to check whether you are attending Linear Algebra lab for any time input from user.** clc; printf("Write a program to check whether you are attending Linear Algebra lab for any time input from user."); t=input("Enter the current time: ") if t>16 & t<18 then printf("student is attending Linear Algebra Lab"); else printf("Student has finished Linear Algebra Lab") end 

<table>
<tr><td>Code 11:</td><td>To check whether given number is greater than 10

clc;
printf("To check whether given number is greater than 10");
a=5;
disp(a);
if a>10 then
printf("a is more than 10");
else
printf("a is smaller than 10");
end


</td></tr>
<tr><td>CONCLUSION:</td><td>Hence, by completing this experiment I came to know about Implementation of Conditional Branching and Looping in Scilab.</td></tr>
</table>

Name	Manish Shashikant Jadhav
UID no.	2023301005
Subject	Linear Algebra
Department	Computer Engineering-B

Experiment 3	
AIM :	Implementation of Row Echelon Form in Scilab.
Row Echelon Form 2x2	clc A = [1 2; 3 -1]; printf("The Matrix A is\n"); disp(A); n = 2; for i = 1:n if A(i,i) == 0 A(i,:) = A(i,:); else A(i,:) = A(i,:) / A(i,i); disp(A); for j = 1:n-1 if i+j <= n A(i+j,:) = A(i+j,:) - A(i+j,i)*A(i,:); end end end end if A(1,2) == A(2,2) A(1,:) = A(1,:) - A(2,:); end printf("The final matrix is: ") disp(A);

```
Scilab 6.0.2 Console
 The Matrix A is

   1.   2.
   3.  -1.

   1.   2.
   3.  -1.

   1.   2.
   0.   1.
The final matrix is:
   1.   2.
   0.   1.

-->
```

Row Echelon Form 3x3

```
clc
A = [1 2 -1 ; 3 -1 1 ; 2 -2 3];
printf("The Matrix A is\n");
disp(A);
n = 3;

for i = 1:n
    if A(i,i) == 0
        A(i,:) = A(i,:);
    else
        A(i,:) = A(i,:) / A(i,i);
        disp(A);
        for j = 1:n-1
            if i+j <= n
                A(i+j,:) = A(i+j,:) - A(i+j,i)*A(i,:);
            end
        end
    end
end
if A(1,2) == A(2,2)
    A(1,:) = A(1,:) - A(2,:);
end
printf("The final matrix is: ")
disp(A);
```

<table>
<tr>
<td></td>
<td>
<pre>
Scilab 6.0.2 Console
 The Matrix A is

   1.   2. -1.
   3. -1.   1.
   2. -2.   3.

   1.   2. -1.
   3. -1.   1.
   2. -2.   3.

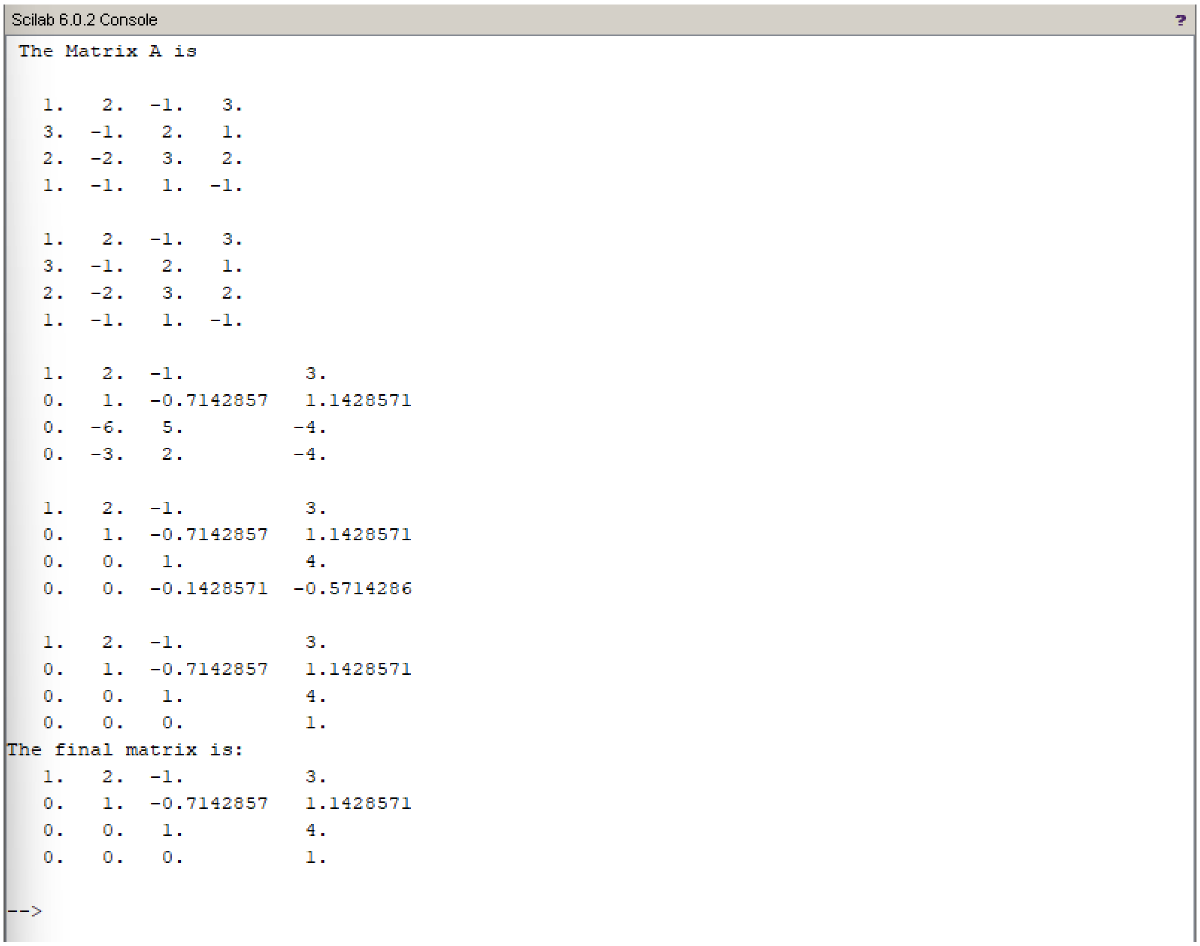
   1.   2. -1.
   0.   1. -0.5714286
   0. -6.   5.

   1.   2. -1.
   0.   1. -0.5714286
   0.   0.   1.
The final matrix is:
   1.   2. -1.
   0.   1. -0.5714286
   0.   0.   1.

-->
</pre>
</td>
</tr>
<tr>
<td>Row Echelon Form 4x4</td>
<td>
<pre>
clc
A = [1 2 -1 3 ; 1 -1 1 -1 ; 2 -2 3 2 ; 3 -1 2 1]
printf("The Matrix A is\n");
disp(A);
n = 4;

for i = 1:n
    if A(i,i) == 0
        A(i,:) = A(i,:);
    else
        A(i,:) = A(i,:) / A(i,i);
        disp(A);
        for j = 1:n-1
            if i+j <= n
                A(i+j,:) = A(i+j,:) - A(i+j,i)*A(i,:);
            end
        end
    end
</pre>
</td>
</tr>
</table>

```
end
if A(1,2) == A(2,2)
   A(1,:) = A(1,:) - A(2,:);
end

printf("The final matrix is: ")
disp(A);
```

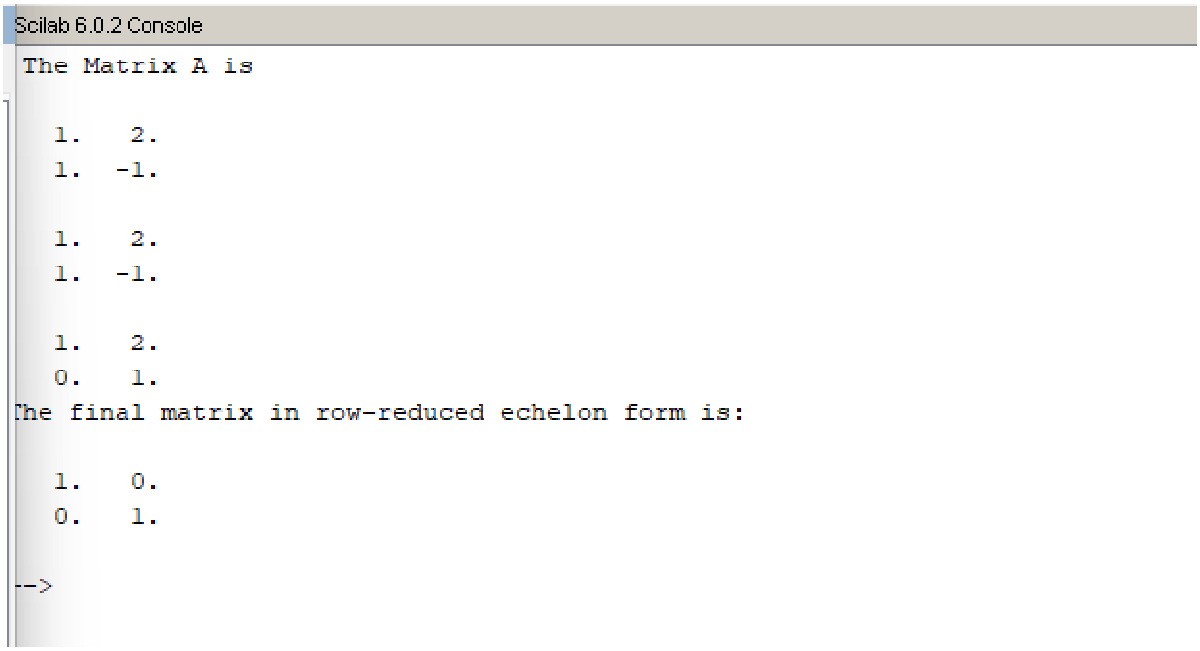


```
Scilab 6.0.2 Console
 The Matrix A is

   1.   2.  -1.   3.
   3.  -1.   2.   1.
   2.  -2.   3.   2.
   1.  -1.   1.  -1.

   1.   2.  -1.   3.
   3.  -1.   2.   1.
   2.  -2.   3.   2.
   1.  -1.   1.  -1.

   1.   2.  -1.          3.
   0.   1.  -0.7142857   1.1428571
   0.  -6.   5.         -4.
   0.  -3.   2.         -4.

   1.   2.  -1.          3.
   0.   1.  -0.7142857   1.1428571
   0.   0.   1.          4.
   0.   0.  -0.1428571  -0.5714286

   1.   2.  -1.          3.
   0.   1.  -0.7142857   1.1428571
   0.   0.   1.          4.
   0.   0.   0.          1.
The final matrix is:
   1.   2.  -1.          3.
   0.   1.  -0.7142857   1.1428571
   0.   0.   1.          4.
   0.   0.   0.          1.


-->
```

CONCLUSION:	Hence, by completing this experiment I came to know about Implementation of Row Echelon Form in Scilab.

Name	Manish Shashikant Jadhav
UID no.	2023301005
Subject	Linear Algebra
Department	Computer Engineering-B

Experiment 4	
AIM :	Implementation of Reduced Row Echelon Form in Scilab.
Reduced Row Echelon Form 2x2	<pre>clc A = [1 2 ; 1 -1]; printf("The Matrix A is\n"); disp(A); n = 2; for i = 1:n if A(i,i) == 0 A(i,:) = A(i,:); else A(i,:) = A(i,:) / A(i,i); disp(A); for j = 1:n-1 if i+j <= n A(i+j,:) = A(i+j,:) - A(i+j,i)*A(i,:); end end end end for i = n:-1:2 for j = i-1:-1:1 A(j,:) = A(j,:) - A(j,i)*A(i,:); end end printf("The final matrix in row-reduced echelon form is: \n"); disp(A);</pre>



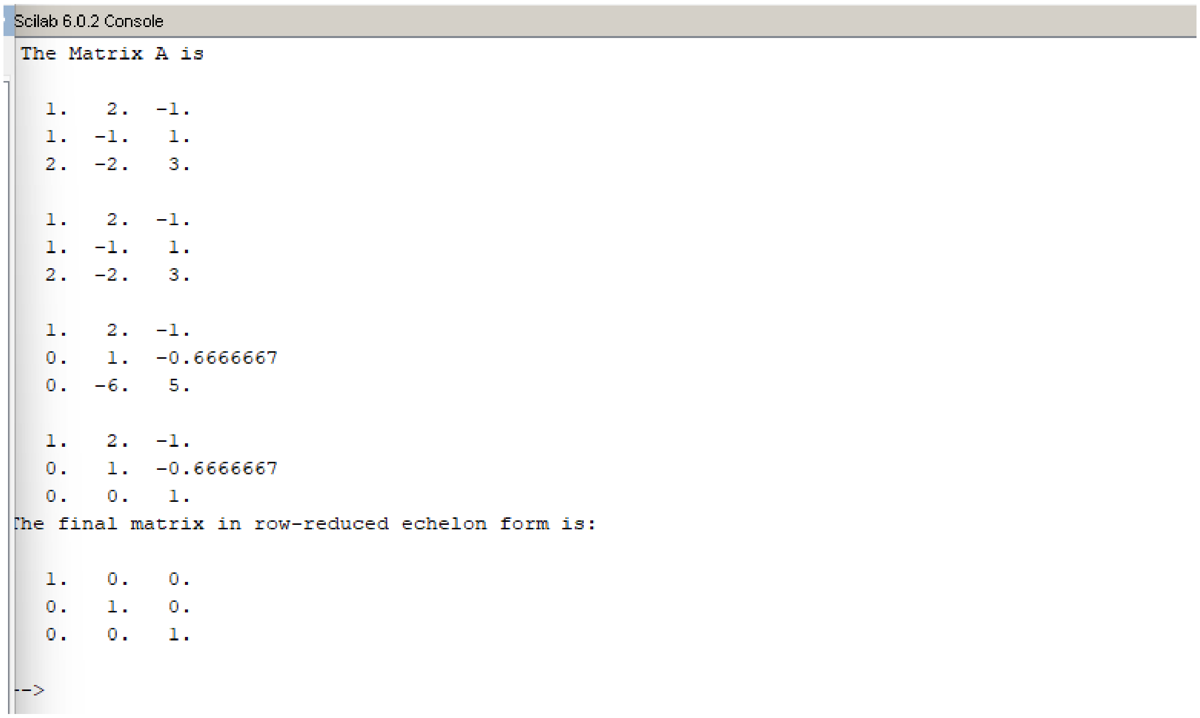
Scilab 6.0.2 Console

```
The Matrix A is

   1.   2.
   1.  -1.

   1.   2.
   1.  -1.

   1.   2.
   0.   1.
The final matrix in row-reduced echelon form is:

   1.   0.
   0.   1.

-->
```

Reduced Row Echelon Form 3x3

```
clc
A = [1 2 -1 ; 1 -1 1 ; 2 -2 3];
printf("The Matrix A is\n");
disp(A);
n = 3;

for i = 1:n
    if A(i,i) == 0
        A(i,:) = A(i,:);
    else
        A(i,:) = A(i,:) / A(i,i);
        disp(A);
        for j = 1:n-1
            if i+j <= n
                A(i+j,:) = A(i+j,:) - A(i+j,i)*A(i,:);
            end
        end
    end
end

for i = n:-1:2
```

```
    for j = i-1:-1:1
        A(j,:) = A(j,:) - A(j,i)*A(i,:);
    end
end

printf("The final matrix in row-reduced echelon form is: \n");
disp(A);
```



```
Scilab 6.0.2 Console
 The Matrix A is

   1.   2.  -1.
   1.  -1.   1.
   2.  -2.   3.

   1.   2.  -1.
   1.  -1.   1.
   2.  -2.   3.

   1.   2.  -1.
   0.   1.  -0.6666667
   0.  -6.   5.

   1.   2.  -1.
   0.   1.  -0.6666667
   0.   0.   1.
The final matrix in row-reduced echelon form is:

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

-->
```

Reduced Row Echelon Form 4x4

```
clc
A = [3 -1 2 1 ; 2 -2 3 2 ; 1 -1 1 -1 ; 1 2 -1 3];
printf("The Matrix A is\n");
disp(A);
n = 4;

for i = 1:n
    if A(i,i) == 0
        A(i,:) = A(i,:);
    else
        A(i,:) = A(i,:) / A(i,i);
        disp(A);
        for j = 1:n-1
            if i+j <= n
```

```
            A(i+j,:) = A(i+j,:) - A(i+j,i)*A(i,:);
         end
      end
   end
end
for i = n:-1:2
   for j = i-1:-1:1
      A(j,:) = A(j,:) - A(j,i)*A(i,:);
   end
end
printf("The final matrix in row-reduced echelon form is: \n");
disp(A);
```

```
Scilab 6.0.2 Console
 The Matrix A is

   1.   2.  -1.   3.
   1.  -1.   1.  -1.
   2.  -2.   3.   2.
   3.  -1.   2.   1.

   1.   2.  -1.   3.
   1.  -1.   1.  -1.
   2.  -2.   3.   2.
   3.  -1.   2.   1.

   1.   2.  -1.          3.
   0.   1.  -0.6666667   1.3333333
   0.  -6.   5.         -4.
   0.  -7.   5.         -8.

   1.   2.  -1.          3.
   0.   1.  -0.6666667   1.3333333
   0.   0.   1.          4.
   0.   0.   0.3333333   1.3333333

   1.   2.  -1.          3.
   0.   1.  -0.6666667   1.3333333
   0.   0.   1.          4.
   0.   0.   0.          1.
The final matrix in row-reduced echelon form is:

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

-->
```

CONCLUSION: Hence, by completing this experiment I came to know about Implementation of Reduced Row Echelon Form in Scilab.

Name	Manish Shashikant Jadhav
UID no.	2023301005
Subject	Linear Algebra
Department	Computer Engineering-B

Experiment 5	
AIM :	Implementation of Gauss Elimination in Scilab.
Code	See below

```
A = [1 1 1;1 2 3;1 3 2]
disp(A);
B = [3;0;3]
disp(B);
C = [A,B]
disp(C);
n = 3;
for i=1:n
   C(i,:) = C(i,:)/C(i,i);
   disp(C)
   for j=1:n-1
      if i+j<n+1
         C(i+j,:)= C(i+j,:)-C(i+j,i)*C(i,:)
      end
   end
   disp(C)
end

z=C(3,4);
y=C(2,4)-C(2,3)*z;
x=C(1,4)-C(1,3)*z-C(1,2)*y;

printf("X=");
disp(x);
printf("Y=");
disp(y);
printf("Z=");
disp(z);
```

Output

```
Scilab 6.0.2 Console
   1.   3.   2.

   3.
   0.
   3.

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

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

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

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

   1.   1.   1.   3.
   0.   1.   2.  -3.
   0.   0.  -3.   6.

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

   1.   1.   1.   3.
   0.   1.   2.  -3.
   0.   0.   1.  -2.
X=
   4.
Y=
   1.
Z=
  -2.

-->
```

CONCLUSION:	Hence, by completing this experiment I came to know about Implementation of Gauss Elimination in Scilab.

Name	Manish Shashikant Jadhav
UID no.	2023301005
Subject	Linear Algebra
Department	Computer Engineering-B

Experiment 6	
AIM :	Implementation of Gauss Jordan in Scilab.
Code	(see below)

```
A = [1 3 2;2 7 7; 2 5 2]
disp(A);
B = [2;-1;7]
disp(B);
C = [A,B]
disp(C);
n = 3;
for i=1:n
   C(i,:) = C(i,:)/C(i,i);
   disp(C)
   for j=1:n-1
      if i+j<n+1
         C(i+j,:)= C(i+j,:)-C(i+j,i)*C(i,:)
      end
   end
   disp(C)
end

for i=n:-1:2
   for j=1:i-1
      C(j,:)=C(j,:)-C(j,i)*C(i,:);
   end
end

disp("X=");
disp(C(1,4));
disp("Y=");
disp(C(2,4));
disp("Z=");
disp(C(3,4));
```

<table>
<tr>
<td>Output</td>
<td>

```
Scilab 6.0.2 Console

--> exec('D:\Manish\SPIT\4th SEM\LA\Prac5\gelimination.sce', -1)

   1.   3.   2.
   2.   7.   7.
   2.   5.   2.

   2.
  -1.
   7.

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

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

   1.   3.   2.   2.
   0.   1.   3.  -5.
   0.  -1.  -2.   3.

   1.   3.   2.   2.
   0.   1.   3.  -5.
   0.  -1.  -2.   3.

   1.   3.   2.   2.
   0.   1.   3.  -5.
   0.   0.   1.  -2.

   1.   3.   2.   2.
   0.   1.   3.  -5.
   0.   0.   1.  -2.

   1.   3.   2.   2.
   0.   1.   3.  -5.
   0.   0.   1.  -2.

 X=

   3.

 Y=

   1.

 Z=

  -2.

-->
```

</td>
</tr>
<tr>
<td>CONCLUSION:</td>
<td>Hence, by completing this experiment I came to know about Implementation of Gauss Jordan in Scilab.</td>
</tr>
</table>

Name	Manish Shashikant Jadhav
UID no.	2023301005
Subject	Linear Algebra
Department	Computer Engineering-B

Experiment 7	
AIM :	Implementation of Gauss Jacobi in Scilab.
Code	```clc A=[5 -2 3;-3 9 1; 2 -1 -7]; B=[-1;2;3]; n=5 x=0; y=0; z=0; for i=1:n printf("\nIteration number: %g",i); X=(B(1)-A(1,2)*y-A(1,3)*z)/A(1,1); Y=(B(2)-A(2,1)*x-A(2,3)*z)/A(2,2); Z=(B(3)-A(3,1)*x-A(3,2)*y)/A(3,3); printf("\nTHE value of x:%g",X); printf("\nTHE value of y:%g",Y); printf("\nTHE value of z:%g",Z); x=X; y=Y; z=Z; end```

Output	Scilab 6.0.2 Console Iteration number: 1 THE value of x:-0.2 THE value of y:0.222222 THE value of z:-0.428571 Iteration number: 2 THE value of x:0.146032 THE value of y:0.203175 THE value of z:-0.51746 Iteration number: 3 THE value of x:0.191746 THE value of y:0.328395 THE value of z:-0.415873 Iteration number: 4 THE value of x:0.180882 THE value of y:0.332346 THE value of z:-0.4207 Iteration number: 5 THE value of x:0.185359 THE value of y:0.329261 THE value of z:-0.424369 -->
CONCLUSION:	Hence, by completing this experiment I came to know about Implementation of Gauss Jacobi in Scilab.

Name	Manish Shashikant Jadhav
UID no.	2023301005
Subject	Linear Algebra
Department	Computer Engineering-B

Experiment 8	
AIM :	Implementation of Gauss Scidel in Scilab.
Code	clc A=[27 6 -1;6 15 2;1 1 54]; B=[85;72;110]; n=5; x=0; y=0; z=0; for i=1:n printf("\nIteration number: %g",i); x=(B(1)-(A(1,2)*y)-(A(1,3)*z))/A(1,1); y=(B(2)-(A(2,1)*x)-(A(2,3)*z))/A(2,2); z=(B(3)-(A(3,1)*x)-(A(3,2)*y))/A(3,3); printf("\nTHE value of x:%g",x); printf("\nTHE value of y:%g",y); printf("\nTHE value of z:%g",z); end

<table>
<tr><td>Output</td><td>

```
Scilab 6.0.2 Console

Iteration number: 1
THE value of x:3.14815
THE value of y:3.54074
THE value of z:1.91317
Iteration number: 2
THE value of x:2.43217
THE value of y:3.57204
THE value of z:1.92585
Iteration number: 3
THE value of x:2.42569
THE value of y:3.57294
THE value of z:1.92595
Iteration number: 4
THE value of x:2.42549
THE value of y:3.57301
THE value of z:1.92595
Iteration number: 5
THE value of x:2.42548
THE value of y:3.57302
THE value of z:1.92595
-->
```

</td></tr>
<tr><td>CONCLUSION:</td><td>Hence, by completing this experiment I came to know about Implementation of Gauss Scidel in Scilab.</td></tr>
</table>

Name	Manish Shashikant Jadhav
UID no.	2023301005
Subject	Linear Algebra
Department	Computer Engineering-B

Experiment 9

AIM : Implementation of Eigen values and Eigen Vectors in Scilab.

Code 1

```
//Eigen values and vectors:
clc
A=[2 -1 1; 1 2 -1 ; 1 -1 2];
disp(A);
[c,d]=spec(A);
printf("The Eigen values of matrix A are:");
disp(d);
printf("The Eigen vectors of matrix A are:");
disp(c);
```

Output 1

```
Scilab 6.0.2 Console

   2.  -1.   1.
   1.   2.  -1.
   1.  -1.   2.
The Eigen values of matrix A are:
   2.   0.   0.
   0.   1.   0.
   0.   0.   3.
The Eigen vectors of matrix A are:
   0.5773503   2.621D-16   0.7071068
   0.5773503  -0.7071068   2.604D-16
   0.5773503  -0.7071068   0.7071068

-->
```

Code 2

```
//Eigen values and vectors:
clc
A=[2 2 1; 1 3 -1;1 2 2];
disp(A);
[c,d]=spec(A);
printf("The Eigen values of matrix A are:");
disp(d);
printf("The Eigen vectors of matrix A are:");
disp(c);
```

Output 2

```
Scilab 6.0.2 Console

   2.   2.   1.
   1.   3.  -1.
   1.   2.   2.
The Eigen values of matrix A are:
   1.   0.                0.
   0.   3. + 2.895D-08i   0.
   0.   0.                3. - 2.895D-08i
The Eigen vectors of matrix A are:
   0.8944272  -0.7071068                -0.7071068
  -0.4472136   6.320D-16 - 1.023D-08i    6.320D-16 + 1.023D-08i
   0.         -0.7071068                -0.7071068

--> |
```

CONCLUSION: Hence, by completing this experiment I came to know about Implementation of Eigen values and vectors in Scilab.