

Vectors, Arrays and Matrixes

Assign a value to a variable (observe the effect of the semi colon):

```
>> x=3;      >> x=3
>>
           x =
           3
```

Creating a row vector of size (1×4)

```
>> x=[1 2 3 5]
```

```
x =
     1     2     3     5
```

Creating a column vector of size (4×1)

```
>> x=[1; 2; 3; 5]
```

```
x =
     1
     2
     3
     5
```

Creating an array of size (1×9) using the semicolon sign (:)

```
>> x=1:.5:5
```

```
x =
     1.0000     1.5000     2.0000     2.5000     3.0000     3.5000     4.0000     4.5000     5.0000
```

```
>> size(x)
```

```
ans =
     1     9
```

Creating an array of size (1×9) using the for loop

```
i=1;
for m=1:0.5:5
    x(i)=m;
    i=i+1;
end
x
```

Creating a matrix of size (3×3)

```
>> A=[1 4 3 ; 2 8 6; 1 7 2]
```

```
A =
     1     4     3
     2     8     6
     1     7     2
```

Creating a matrix of size (3×3) using 2 for loops

```
i=1;
for m=1:1:3
    j=1;
    for n=0:2:4
        x(i,j)=m+n;
        j=j+1;
    end
    i=i+1;
end
x
x =
```

```
     1     3     5
     2     4     6
     3     5     7
```

Basic Matrix Operations:

```
>> A=[1 -2 3; 4 -5 6; 2 7 9]
```

```
A =
```

```
    1    -2     3
    4    -5     6
    2     7     9
```

```
>> B=[2 0 3; 7 8 2; 1 -5 6]
```

```
B =
```

```
    2     0     3
    7     8     2
    1    -5     6
```

```
>> det(A)
```

```
ans =
```

```
    75
```

```
>> size(A)
```

```
ans =
```

```
     3     3
```

```
>> rank(A)
```

```
ans =
```

```
     3
```

```
>> inv(A)
```

```
ans =
```

```
 -1.1600    0.5200    0.0400
 -0.3200    0.0400    0.0800
  0.5067   -0.1467    0.0400
```

```
>> A'
```

```
ans =
```

```
     1     4     2
    -2    -5     7
     3     6     9
```

```
>> A+B
```

```
ans =
```

```
     3    -2     6
    11     3     8
     3     2    15
```

```
>> A-B
```

```
ans =
```

```
    -1    -2     0
    -3   -13     4
     1    12     3
```

```
>> A*B
```

```
ans =
```

```
    -9   -31    17
   -21   -70    38
    62    11    74
```

```
>> A.*B
```

```
ans =
```

```
     2     0     9
    28   -40    12
     2   -35    54
```

Eigenvectors and Eigenvalues of a Matrix:

```
>> eig(A)
```

```
ans =
```

```
 12.1652
 -1.0000
 -6.1652
```

```
>> [Eigvectors, Eigvalues]=eig(A)
```

```
Eigvectors =
```

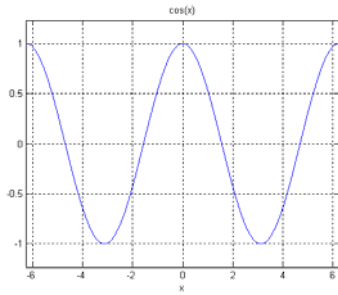
```
 -0.1809  -0.8739  -0.4045
 -0.3618  -0.2984  -0.8089
 -0.9145   0.3837   0.4267
```

```
Eigvalues =
```

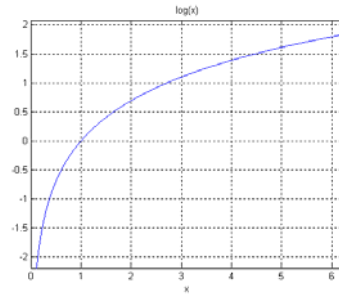
```
 12.1652     0     0
     0  -1.0000     0
     0     0  -6.1652
```

Graphic Utilities

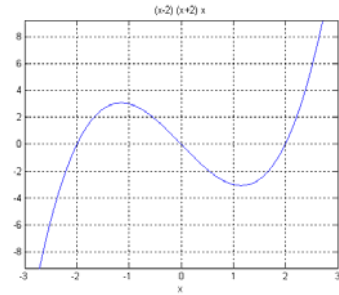
```
>> ezplot('cos(x)')
>> grid on
```



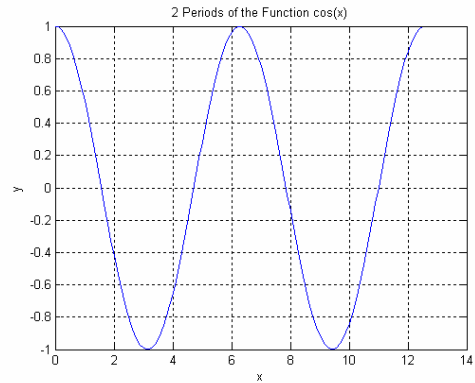
```
>> ezplot('log(x)')
>> grid on
```



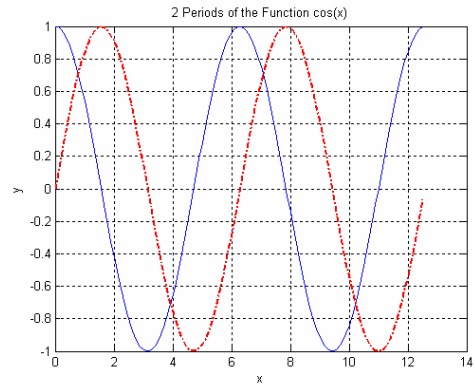
```
>> ezplot('(x-2)*(x+2)*x',[-3,3])
>> grid on
```



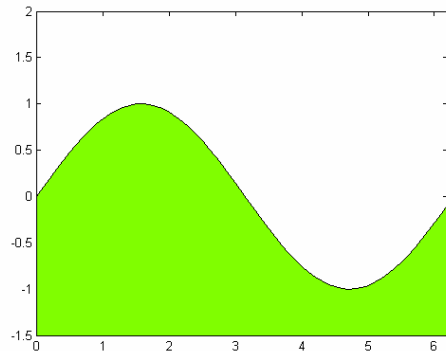
```
>> x=0:0.1:4*pi;
>> y=cos(x);
>> grid on
>> x=0:0.1:4*pi;
>> y=cos(x);
>> plot(x,y)
>> grid on
>> title('2 Periods of the Function cos(x)')
>> xlabel('x')
>> ylabel('Y')
```



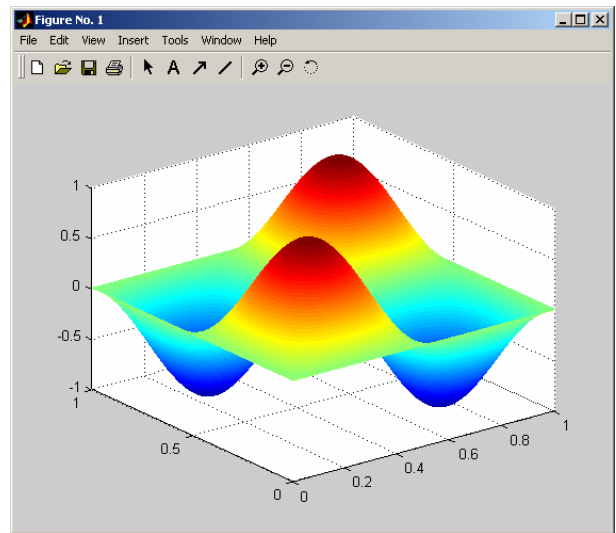
```
>> z=sin(x);
>> hold on
>> plot(x,z,'r-.','Linewidth',2)
```



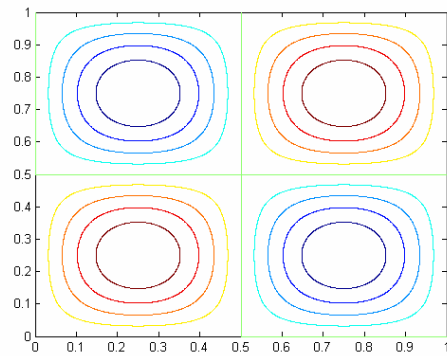
```
x=0:0.1:2*pi;
y=sin(x);
area(x,y,-1.5,'FaceColor',[0.5 1.0 0.0])
ylim([-1.5 2])
```



```
clear all
i=1;
for dt1=0:0.001:1
    x(i)=dt1;
    j=1;
    for dt2=0:0.001:1
        y(j)=dt2;
        z(i,j)=sin(2*pi*dt1)*sin(2*pi*dt2);
        j=j+1;
    end
    i=i+1;
end
mesh(x,y,z)
```



```
contour(x,y,z)
```



Complex Numbers

Defining a complex number:

```
>> x=3+4i
```

```
x =
```

```
3.0000 + 4.0000i
```

```
>> x=3+4j
```

```
x =
```

```
3.0000 + 4.0000i
```

Conjugate

```
>> y=x'
```

```
y =
```

```
3.0000 - 4.0000i
```

Absolute Value

```
>> abs(x)
```

```
ans =
```

```
5
```

Angle in Radians

```
>> angle(x)
```

```
ans =
```

```
0.9273
```

Angle in Degrees

```
>> angle(x)*180/pi
```

```
ans =
```

```
53.1301
```

```
>> y=2-5i
```

```
y =
```

```
2.0000 - 5.0000i
```

```
>> x*y
```

```
ans =
```

```
26.0000 - 7.0000i
```

```
>> x/y
```

```
ans =
```

```
-0.4828 + 0.7931i
```

```
>> x-y
```

```
ans =
```

```
1.0000 + 9.0000i
```

```
>> x+y
```

```
ans =
```

```
5.0000 - 1.0000i
```

Loading Data from a File and Saving Data to a File

```
>> A=load('datalarim.dat')
```

A =

	0	0
0.1000	0.0100	
0.2000	0.0400	
0.3000	0.0900	
0.4000	0.1600	
0.5000	0.2500	
0.6000	0.3600	
0.7000	0.4900	
0.8000	0.6400	
0.9000	0.8100	
1.0000	1.0000	
1.1000	1.2100	
1.2000	1.4400	
1.3000	1.6900	
1.4000	1.9600	
1.5000	2.2500	
1.6000	2.5600	
1.7000	2.8900	
1.8000	3.2400	
1.9000	3.6100	
2.0000	4.0000	

```
>> B(:,1)=A(:,1);
```

```
>> B(:,2)=2*A(:,2)
```

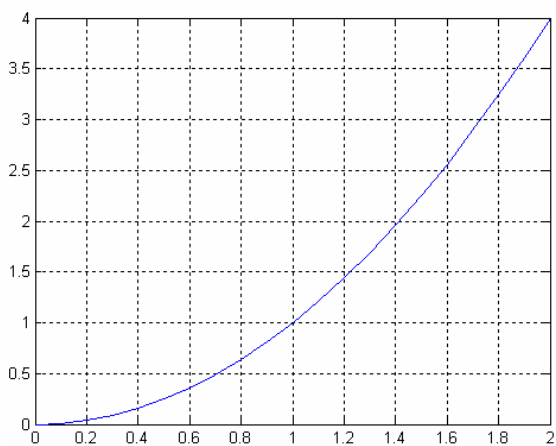
B =

	0	0
0.1000	0.0200	
0.2000	0.0800	
0.3000	0.1800	
0.4000	0.3200	
0.5000	0.5000	
0.6000	0.7200	
0.7000	0.9800	
0.8000	1.2800	
0.9000	1.6200	
1.0000	2.0000	
1.1000	2.4200	
1.2000	2.8800	
1.3000	3.3800	
1.4000	3.9200	
1.5000	4.5000	
1.6000	5.1200	
1.7000	5.7800	
1.8000	6.4800	
1.9000	7.2200	
2.0000	8.0000	

```
>> save('datalarim2.dat', 'B', '-ascii', '-tabs')
```

```
>> plot(A(:,1),A(:,2))
```

```
>> grid on
```



```
>> format short
>> x=1/3
```

```
x =
    0.3333
```

```
>> format long
>> x=1/3
```

```
x =
    0.3333333333333333
```

Defining a Polynom

$$P(x) = x^3 + 2x^2 + 3x + 4$$

```
>> coef=[1 2 3 4]
```

```
coef =
     1     2     3     4
```

Roots of a Polynom

```
>> roots(coef)
```

```
ans =
   -1.6506
  -0.1747 + 1.5469i
  -0.1747 - 1.5469i
```

clc: To clear the command window
clear all: To clear all the variables in memory

Symbolic Integration

```
>> syms x
>> y=int(1/(x-3))
```

```
y =
log(x-3)
```

Numeric Integration

```
>> x=0:0.1:1;
>> y=x.^2;
>> Area=trapz(x,y)
```

```
Area =
    0.3350
```

Mean Value, Variance and Standard Deviation of a Set of Data

```
>> a=[1 3 -4 5 6 7 2 1 0 -14 2 9 11 7 8]
```

```
a =
     1     3    -4     5     6     7     2     1     0    -14     2     9    11     7     8
```

```
>> mean(a)
```

```
ans =
    2.9333
```

```
>> var(a)
```

```
ans =
   37.6381
```

```
>> std(a)
```

```
ans =
    6.1350
```

```
>> fprintf (' This will appear on the screen \n')
This will appear on the screen
```

```
>> format long
>> pi
```

```
ans =

3.14159265358979
```

```
>> fprintf('%1.2f',pi)
3.14
```

```
>> fprintf('%1.5f',pi)
3.14159
```

```
>> fprintf('%1.8f',pi)
3.14159265
```

```
>> a=3; b=1.2; c=0.0003;
>> fprintf ('%d \n %3.1f \n %6.5f \n', a,b,c)
3
1.2
0.00030
```

round		floor		ceil	
Round towards nearest integer.		Round towards minus infinity.		Round towards plus infinity.	
>> round(6.7)	>> round(6.3)	>> floor(6.7)	>> floor(6.3)	>> ceil(6.7)	>> ceil(6.3)

ans =	ans =	ans =	ans =	ans =	ans =
7	6	6	6	7	7

```
>> rand(1,10)
```

```
ans =

0.5678 0.7942 0.0592 0.6029 0.0503 0.4154 0.3050 0.8744 0.0150 0.7680
```

```
>> randint(1,10)
```

```
ans =

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

```
>> randint(3,10)
```

```
ans =

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

Sample Matlab code 1: To add natural numbers from 1 to a user-given maximum number; (a) using the short formulae, (b) using iteration

```
clear all
clc
n=input('n Degerini Giriniz: ');

% a)
Total1=n*(n+1)/2

% b)
Total2=0;
for i=1:n
    Total2=Total2+i;
end
Total2
```

Sample Matlab code 2: To calculate e^x function using the first n terms in its Taylor Expansion; (a) via the calculation of the error when the user supplies the number of terms, (b) via the calculation of the number of terms when the error is given.

a)

```
clear all;
clc;
x=input('Give x');
n=input('Number of terms = ?');

exp1=exp(x);
exp2=1;
for k=1:n
    exp2=exp2+x^k/factorial(k);
end
error=exp1-exp2
```

b)

```
clear all;
clc;
x=input('Give x');
eps=input('Give the error');

exp1=exp(x);
exp2=1;

error=1+eps;
n=1;
while error >= eps
    exp2=exp2+x^n/factorial(n);
    error=exp1-exp2;
    n=n+1;
end
n
```

Sample Matlab code 3: Calling a function

```
function cagatay
clear all;
clc;
a=5.345;
b=3.123;
c=7.547;
average(a,b,c)

function res=average(a,b,c)
res=(a+b+c)/3;
```

Sample Matlab code 4: To find out the number in a given decimal digit

a) m-File

```
clear all; clc;
a=input('Give a real number ');
n=input('Which decimal digit do you want = ? ');

% Truncate the number after nth digit
b=floor(a*10^n)/10^n;

% Truncate the number after (n-1)th digit
c=floor(a*10^(n-1))/10^(n-1);

% To keep only the nth digit
d=b-c;
e=d*10^n;

% To display the result with a given format
fprintf('The number after %1.f digit = %1.f',n,e)
```

b) Screen output

```
Give a real number 0.12345678
Which decimal digit do you want = ? 4
The number after 4 digit = 4
>> |
```