**Function**

Declare function name, inputs, and outputs

**Syntax**

function [y1,...,yN] = myfun(x1,...,xM)

**Description**

function [y1,...,yN] = myfun(x1,...,xM) declares a function named myfun that accepts inputs x1,...,xM and returns outputs y1,...,yN. This declaration statement must be the first executable line of the function. Valid function names begin with an alphabetic character, and can contain letters, numbers, or underscores.

You can save your function:

* In a function file which contains only function definitions. The name of the file should match the name of the first function in the file.
* In a script file which contains commands and function definitions. Functions must be at the end of the file. Script files cannot have the same name as a function in the file. Functions are supported in scripts in R2016b or later.

Files can include multiple local functions or nested functions. For readability, use the [end](https://www.mathworks.com/help/matlab/ref/end.html) keyword to indicate the end of each function in a file. The end keyword is required when:

* Any function in the file contains a nested function.
* The function is a local function within a function file, and any local function in the file uses the end keyword.
* The function is a local function within a script file.

my\_string = 'test abd 11';

str\_ascii = uint8(my\_string) % 8-bit ascii values

str\_back\_to\_char= char(str\_ascii)

str\_16bit = uint16(my\_string) % 16-bit ascii values

str\_back\_to\_char = char(str\_16bit)

name = 'Raman Sinha ';

position = 'Nidhi Visht ';

worksAt = 'R N Tagore Cardiology Research Center';

profile = [name ', ' position ', ' worksAt]

profile = strcat(name, ', ', position, ', ', worksAt)

Define a function in a file named stat.m that returns the mean and standard deviation of an input vector.

function [m,s] = stat(x)

n = length(x);

m = sum(x)/n;

s = sqrt(sum((x-m).^2/n));

end

Call the function from the command line.

values = [12.7, 45.4, 98.9, 26.6, 53.1];

[ave,stdev] = stat(values)

ave =

47.3400

stdev =

29.4124

### Multiple Functions in a Function File

Define two functions in a file named stat2.m, where the first function calls the second.

function [m,s] = stat2(x)

n = length(x);

m = avg(x,n);

s = sqrt(sum((x-m).^2/n));

end

function m = avg(x,n)

m = sum(x)/n;

end

Function avg is a local function. Local functions are only available to other functions within the same file.

Call function stat2 from the command line.

values = [12.7, 45.4, 98.9, 26.6, 53.1];

[ave,stdev] = stat2(values)

ave =

47.3400

stdev =

29.4124

## Add Functions to Scripts

MATLAB® scripts, including live scripts, can contain code to define functions. These functions are called local functions. Local functions are useful if you want to reuse code within a script. By adding local functions, you can avoid creating and managing separate function files. They are also useful for experimenting with functions, which can be added, modified, and deleted easily as needed. Functions in scripts are supported in R2016b or later.

### Add Local Functions

Local functions are only visible within the file where they are defined, both to the script code and other local functions within the file. They are not visible to functions in other files, and cannot be called from the command line. They are equivalent to subroutines in other programming languages, and are sometimes called subfunctions.

To add local functions to a script, first, create the script. Go to the **Home** tab and select **New** > **Script**. For more information about creating scripts

x = 1:10;

n = length(x);

avg = mymean(x,n);

med = mymedian(x,n);

function a = mymean(v,n)

% MYMEAN Example of a local function.

a = sum(v)/n;

end

function m = mymedian(v,n)

% MYMEDIAN Another example of a local function.

w = sort(v);

if rem(n,2) == 1

m = w((n + 1)/2);

else

m = (w(n/2) + w(n/2 + 1))/2;

end

end

function max = mymax(n1, n2, n3, n4, n5)

%This function calculates the maximum of the

% five numbers given as input

max = n1;

if(n2 > max)

max = n2;

end

if(n3 > max)

max = n3;

end

if(n4 > max)

max = n4;

end

if(n5 > max)

max = n5;

end

help mymax

mymax(34, 78, 89, 23, 11)

function [x1,x2] = quadratic(a,b,c)

%this function returns the roots of

% a quadratic equation.

% It takes 3 input arguments

% which are the co-efficients of x2, x and the

%constant term

% It returns the roots

d = disc(a,b,c);

x1 = (-b + d) / (2\*a);

x2 = (-b - d) / (2\*a);

end % end of quadratic

function dis = disc(a,b,c)

%function calculates the discriminant

dis = sqrt(b^2 - 4\*a\*c);

end % end of sub-function

quadratic(2,4,-4)

Our text file weeklydata.txt looks like this −

SunDay MonDay TuesDay WednesDay ThursDay FriDay SaturDay

95.01 76.21 61.54 40.57 55.79 70.28 81.53

73.11 45.65 79.19 93.55 75.29 69.87 74.68

60.68 41.85 92.18 91.69 81.32 90.38 74.51

48.60 82.14 73.82 41.03 0.99 67.22 93.18

89.13 44.47 57.63 89.36 13.89 19.88 46.60

Create a script file and type the following code in it −

filename = 'weeklydata.txt';

delimiterIn = ' ';

headerlinesIn = 1;

A = importdata(filename,delimiterIn,headerlinesIn);

% View data

for k = [1:7]

disp(A.colheaders{1, k})

disp(A.data(:, k))

disp(' ')

end