Welcome to the spoken tutorial on Scripts and Functions with Scilab.

Let us start with a brief introduction to the file formats in Scilab.

When several commands are to be executed, it may be more convenient to write these statements into a file with Scilab editor. These are called as SCRIPT files.

To execute the commands written in such a script file, the exec function can be used, followed by the name of the script file.

These file generally have the extension .sce or .sci, depending on its content.

Files having the .sci extension contains Scilab functions and/or user defined functions and executing them loads the functions into Scilab environment (but does not execute them),whereas

Files having the .sce extension contains both Scilab functions and executable statements.

Please remember that the convention of naming
the extension as .sce and .sci are not RULES, but a convention followed by the scilab community.

Let us Open Scilab on the computer.

Check the present working directory by typing cd on the command prompt

```
-->cd
ans  =

C:\Documents and Settings\cdeep_lap12
```

Go to the Task bar of scilab console window and click on editor option to Open the scilab editor

I have already typed the commands in a file and saved it as helloworld.sce, therefore I will open that file using Open a file shortcut icon.

```
disp("Hello World")
a=1; b=2; c=a+b;
d=a+b+c;
disp(d)
disp("Goodbye World")
```

You may type the commands in the new file and Save this file to the current working directory as HelloWorld.sce through the File Menu.

Keep in mind that there should be no spaces in the filename.

Go to Execute button on the scilab editors menu bar and select Load into Scilab option.

This will load the file in scilab console.

After loading the file in the scilab console the script produces the output as you can see:

```
-->disp("Hello World")

 Hello World

--> a=1; b=2; c=a+b;

--> d=a+b+c;

--> disp(d)

    6.

--> disp("Goodbye World")

 Goodbye World
```

It contains both the commands and the resulting output for the respective commands.

Now change the value of a to 5 in the editor,go to File menu , again select save and close it.

We can also execute the script directly from the scilab interpreter using the exec command:

```
-->exec("Hello World.sce")
```

The script file produces a similar output with the use exec function.

```
-->disp("Hello World")

Hello World

-->a=5; b=2; c=a+b;
-->d=a+b+c;
-->disp(d)

    14.
```

```
-->disp("Goodbye World")

 Goodbye World
```

Let us now talk about Functions:

A function definition starts with the keyword function and ends with the keyword endfunction.

I have already saved this function in a file function.sci using the scilab editor.

The function is defined as you see,

```
 function [degrees] = radians2degrees(radians)
   degrees = radians*(180/%pi);
 endfunction
```

Here degrees is the output parameter and radians is the input parameter to the function named radians2degrees.

I will now load this function in Scilab using the Execute menu option.

It can also be loaded using exec command

```
 -->exec functions.sci
```

The function is now loaded in the scilab console.

```
 -->function [degrees] = radians2degrees(radians)
 -->  degrees = radians*(180/%pi);
 -->endfunction
```

Once a function is loaded, it can be used as if it was any other Scilab function by passing the arguments to that function.

Make a mental note of the % sign and recall the reason for its use.

Now let us find values for radians2degrees(%pi/2) and radians2degrees(%pi/4).

```
 -->radians2degrees(%pi/2)
  ans  =

    90.

 -->radians2degrees(%pi/4)
```

```
 ans  =

    45.
```

One of the interesting feature of scilab is that you can define any number of functions in a single .sci file.
While doing this please remember that by default all the variables defined in a function are local and the scope of variables used in a particular function ends with the endfunction keyword of the function definition.

Advantage of this feature is that we can use same variable names in different function. These variables won't get mixed up unless we use the global option.

To know more about the global variables type help global and see a detailed help yourself.

Please note that if any variable is to be "watched" or monitored inside a function, then disp is required.

Inside a function file, you can check for yourself the effect of putting a semicolon (;) at the end of a statement with or without a semicolon. Also check this for disp("..." statements.

Inline Functions:

Functions are segments of code that have well defined input and output as well as local variables. The simplest way to define a function is by using the command `deff'.

Scilab allows the creation of in-line functions and are especially useful when the body of the function is short. This can be done with the help of the function deff(). It takes two string parameters. The first string defines the interface to the function and the second string that defines the statements of the function.

The deff command defines the function in the scilab and also loads it.

There is no need to load the function defined by using deff command explicitly through execute menu option.

Let us see an example to illustrate this concept:

```
 deff("[radians] = degrees2radians(degrees)","radians = degrees*(%pi/180)")
```

As mentioned earlier the first string defines the interface to the function and the second string that defines the statements of the function.

We can directly use it to find the values of degrees2radians(90) and degrees2radians(45).

```
-->degrees2radians(90)
 ans  =

    1.5707963

-->degrees2radians(45)
 ans  =

    0.7853982
```

A function could call, not just other functions within itself, but also ITSELF. This is "recursive" calling of a function. This is required, for example, when writing a function to calculate the factorial of an integer.

Let us extend the discussion on file formats in Scilab:

As mentioned earlier SCILAB uses two types of file formats, namely the SCE file format and the SCI file format.

The files with the .sce file extension are the script files, which contain the SCILAB commands that you enter during an interactive kind of SCILAB session. They can comprise of comment lines utilized in documenting the function and they can also use the command EXEC to execute the script.

The files with the .sci file extension are the function files that start with the function statement.
A single .sci file can have multiple function definitions which themselves contain any number of SCILAB statements that perform operations on the function arguments, or on the output variables after they have been evaluated.

This brings us to the end of this spoken tutorial on Scripts and Functions in Scilab. There are many other functions in Scilab which will be covered in other spoken tutorials. Keep watching the Scilab links.

Spoken Tutorials are part of the Talk to a Teacher project, supported by the National Mission on Education through ICT. More information on the same is available at http://spoken-tutorial.org/NMEICT-Intro.

Thanks for joining. Goodbye.