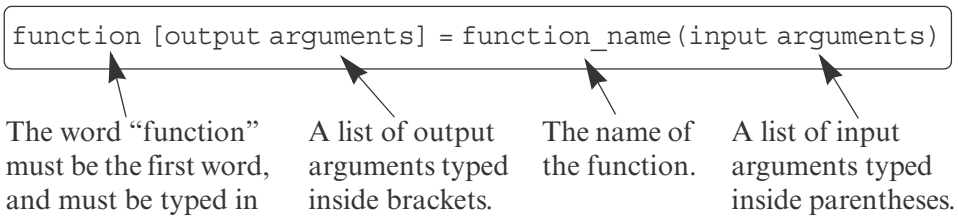
The various parts of the function file are described in detail in the following sections.

7.2.1 Function Definition Line

The first executable line in a function file *must* be the function definition line. Otherwise the file is considered a script file. The function definition line:

- Defines the file as a function file
- Defines the name of the function
- Defines the number and order of the input and output arguments

The form of the function definition line is:



The word "function," typed in lowercase letters, must be the first word in the function definition line. On the screen the word function appears in blue. The function name is typed following the equal sign. The name can be made up of letters, digits, and the underscore character (the name cannot include a space). The rules for the name are the same as the rules for naming variables described in Section 1.6.2. It is good practice to avoid names of built-in functions and names of variables already defined by the user or predefined by MATLAB.

7.2.2 Input and Output Arguments

The input and output arguments are used to transfer data into and out of the function. The input arguments are listed inside parentheses following the function name. Usually, there is at least one input argument, although it is possible to have a function that has no input arguments. If there are more than one, the input arguments are separated with commas. The computer code that performs the calculations within the function file is written in terms of the input arguments and assumes that the arguments have assigned numerical values. This means that the mathematical expressions in the function file must be written according to the dimensions of the arguments, since the arguments can be scalars, vectors, or arrays. In the example shown in Figure 7-2 there are three input arguments (`amount,rate,years`), and in the mathematical expressions they are assumed to be scalars. The actual values of the input arguments are assigned

when the function is used (called). Similarly, if the input arguments are vectors or arrays, the mathematical expressions in the function body must be written to follow linear algebra or element-by-element calculations.

The output arguments, which are listed inside brackets on the left side of the assignment operator in the function definition line, transfer the output from the function file. Function files can have zero, one, or several output arguments. If there are more than one, the output arguments are separated with commas. If there is only one output argument, it can be typed without brackets. *For the function file to work, the output arguments must be assigned values in the computer program that is in the function body.* In the example in Figure 7-2 there are two output arguments, `mpay` and `tpay`. When a function does not have an output argument, the assignment operator in the function definition line can be omitted. A function without an output argument can, for example, generate a plot or write data to a file.

It is also possible to transfer strings into a function file. This is done by typing the string as part of the input variables (text enclosed in single quotes). Strings can be used to transfer names of other functions into the function file.

Usually, all the input to, and the output from, a function file transferred through the input and output arguments. In addition, however, all the input and output features of script files are valid and can be used in function files. This means that any variable that is assigned a value in the code of the function file will be displayed on the screen unless a semicolon is typed at the end of the command. In addition, the `input` command can be used to input data interactively, and the `disp`, `fprintf`, and `plot` commands can be used to display information on the screen, save to a file, or plot figures just as in a script file. The following are examples of function definition lines with different combinations of input and output arguments.

Function definition line	**Comments**
function[mpay,tpay]= loan(amount,rate,years)	Three input arguments, two output arguments.
function [A] = RectArea(a,b)	Two input arguments, one output argument.
function A = RectArea(a,b)	Same as above; one output argument can be typed without the brackets.
function [V, S] = SphereVolArea(r)	One input variable, two output variables.
function trajectory(v,h,g)	Three input arguments, no output arguments.