11 General form of a user-defined function

function [out1, out2, ...] = fname(in1, in2, ...)
% H1 comment line
% Other comment lines

% executable code—the function body

Upon invocation, each function has its own memory space that is inaccessible by other functions or the command window space—variables in a function are local to the function and can be “seen” only inside the function.

Values stored in local variables are not preserved between function calls.

The keyword function indicates that an M-file contains a user-defined function.

fname above is the name of the function and is also the name of the file (without the extension .m).

The input parameter list is enclosed in parentheses and the parameters are separated by commas.

MATLAB functions can return multiple values (output arguments). If a MATLAB function does not return any value, then omit from the function header the output parameter list, along with the square brackets, and the equal sign.

The H1 comment line is searchable by MATLAB’s lookfor command. Put a short description of the function in the H1 comment line.

The other comment lines will be displayed by MATLAB’s help command.

Calling a function:
Suppose the function header is

function [out1, out2] = foo(in1, in2, in3)
Then to call the function, write the statement

[x, y] = foo(10, rand(1,1), 2)

Above, we assume all the input parameters are numeric, scalars variables. After the function executes, the value of out1 in the function foo is stored in variable x, and the value of out2 in the function foo is stored in variable y. Note that any arguments may be matrices.

Subfunctions

There can be more than one function in an M-file.

The top function is normal, and it has the name of the M-file.

The remaining functions are subfunctions, and are accessible only by the top function.

12 Reminder about random number generator rand

MATLAB’s pre-defined function rand generates a number in the range of 0 to 1 randomly. In other words, function rand generates a number from the standard uniform distribution: any number in the range of 0 to 1 is equally likely to occur. Note that the range is the open interval (0,1).
13 2-Dimensional Array: Matrix

In MATLAB, two dimensional arrays are called *matrices*. Matrices are rectangular! Use square brackets to delimit arrays. Using a space or a comma as the separator means to put something to the right of the previous unit; using a semicolon as the separator means to put something below the previous unit.

MATLAB array index starts at 1, not zero. To access a value in a matrix, specify the row and column index values, separated by a comma, inside parentheses. For example, \(x(2,4)\) is the value in the 2nd row and the fourth column of matrix \(x\).

```matlab
m = [1 2 3 4; 5 6 7 8]  % 2-by-4 matrix m
[nr,nc] = size(m)  % nr stores the no. of rows; nc stores the no. of columns
m = [m; zeros(1,nc)]  % new matrix m: stack a row of zeros below the current m
m = [m m]  % new matrix m: put 2 m’s side-by-side
m = [m; m]
v = 1:6
newm = [m v']
newm = newm'
m1 = rand(4,3)  % 4-by-3 matrix with random values (uniform dist.)
tmp = m1(3,2)  % cell in 3rd row, 2nd column
tmp = m1(3:4,:)  % submatrix of m1: rows 3 to 4, all columns
tmp = m1(:,2)  % submatrix of m1: all rows, column 2
tmp = m1([1 4],:)  % submatrix of m1: rows 1 and 4, all columns
tmp = m1(:, [1 3])
tmp = m1([1 4],[1 3])
```

14 Logical arrays and operations

Logical arrays, i.e., arrays containing logical values, are the results of relational or logical operations. In MATLAB, logical values are zero for false and one (or any non-zero value) for true. Logical values are not just numbers—they have the logical property attached to the data, see the workspace window under “class” when you have a logical value in the MATLAB workspace.

You can write vetorized code for relational or logical operations when you need cell-by-cell comparisons. The result will be a logical vector. For example, let \(x\) be the vector \([2 \ 3 \ 5 \ 2]\). Then the expression \(x==2\) will give the logical array \([1 \ 0 \ 0 \ 1]\). Did you notice that \(x==2\) is vetorized code? It is vetorized because the relational operation (==) is performed on all cells in vector \(x\) in one step.

The use of function `find` and the extraction of subvectors based on relational operations are included below for your reference should you use MATLAB in the future. You are not responsible for this material in CS100J. You should, however, learn how to write vetorized code involving relational or logical operations (as discussed in the lab section).

```matlab
elev = 8*rand(4,3) + 10  % example, elevations on a map
elev > 16  % returns a logical array

% 1-d examples
vec = elev(1,: )  % 1st row of matrix elev
L = vec>16  % logical array indicating result from vec>16
vecHigh = vec(L)  % extract just the cells with values > 16
vecHigh = vec(vec>16)  % combine last two statements in one
% this shortcut works for VECTORS only, not matrices

ind = find(vec>16)  % get the indices where vec>16
vecHigh = vec(ind)  % extract just the cells with values > 16
```
% Create a vector same as vec above except that all the values below 16 are "zeroed out".
% (There's a simpler solution that uses vectorized multiplication. See lab exercise.)

L = (vec>16)   % a LOGICAL vector
vecHigh = zeros(1,length(vec))
vecHigh(L) = vec(L)  % assign only to the cells with logical value 1

ind = find(vec>16)   % a vector of INDICES
vecHigh = zeros(1,length(vec))
vecHigh(ind) = vec(ind)  % assign only to the cell numbers stored in ind

% 2-d examples
L = elev>16       % logical array (matrix)
elevHigh = elev(elev>16)  % a VECTOR!!!

[ri,ci] = find(elev>16)   % ri is vector that stores row index where elev>16
% ci is vector that stores col index where elev>16

15 String creation and manipulation

This topic is included for your reference should you use MATLAB in the future. You are not responsible for this topic in CS100J.

str = 'Age 19'   % a 1-d array of characters
code = double(str)  % convert chars to ASCII values
str1 = char(code)  % convert ASCII values to chars

% 2-d array of characters
block = ['one row'; 'two rows']  % Error! Rows must have same length
block = ['one row '; 'two rows']
blk = char('one row', 'two rows')
line1 = blk(1,:)  % length 8
line1trim = deblank(blk(1,:))  % length 7, trailing blank removed

% string functions
str = 'Age 19'
issrtchar(str)  % is the variable a char array? Return ONE value
isletter(str)  % is the cell content a letter? Return one value for each cell
isspace(str)
caps = upper(str)
small = lower(str)

% char arithmetic, relation
base = 'a'
nextcode = base + 1
nextletter = char(nextcode)
letter18 = char(base+18-1)
ans1 = 'a' > 'b'
ans2 = base=='a'
ans3 = base==letter18
blk = char('one row', 'two rows')
ans4 = blk==0'   % character-by-character comparison
ans5 = blk(1,:)==blk(2,:)  % character-by-character comparison