

MATLAB Marina: Iteration, for loops

Student Learning Objectives

After completing this module, one should:

1. Be able to use for loops to iteratively execute blocks of statements.
2. Be able to determine when to use array operations versus iteration.
3. Be able to explain the types of operations that require for loops.

Terms

iteration, loop, loop control variable, loop control array, nested loop, running sum, running product

MATLAB Functions, Keywords, and Operators

for, end

For Statements

Iterative program structures (`for`, `while`) allow programs to repetitively execute blocks of statements. By default, all program statements are executed once sequentially. A `for` statement (for loop) operates according to the flowchart of Figure 1a. The general syntax of a `for` statement is shown in Figure 1b.

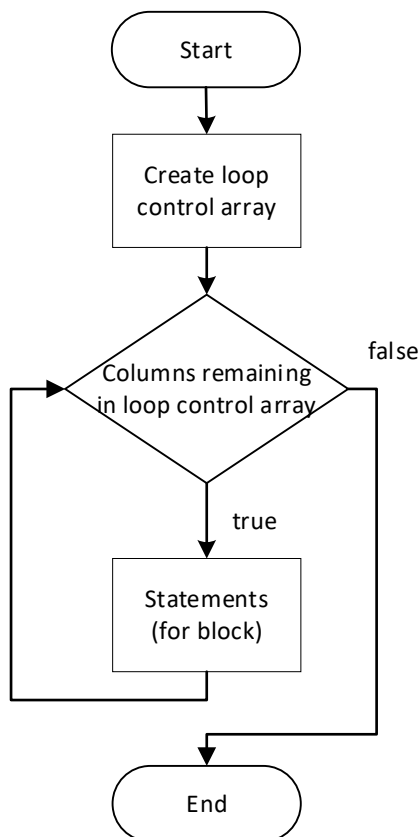


Figure 1a. for Statement Flowchart

```
for variable = expression
    statements to be executed
end
```

Figure 1b. General Form of for Statement

The `for` statement is characterized by a loop control expression and a block of statements to execute. The expression can either be an existing array or a MATLAB expression that results in an array. `For` statements are counter controlled loops and will execute the block of statements a fixed number of times. The number of times the statements are executed is known once the expression is evaluated (at the start of the loop). Note that `for` statements must be terminated with the `end` keyword.

The `for` statement operates as follows:

- The expression is evaluated resulting in an array (loop control array).
- Columns from the array are assigned to the variable (loop control variable) one column at a time.
- The statements between the `for` and `end` keywords are executed for the current value of the loop control variable.
- This is repeated until all the columns in the loop control array have been used.

If the loop control array is a 1D array, the loop control variable will get a scalar value for each loop iteration. For a 2D array, the loop control variable will get a column array for each loop iteration. If the expression results in a scalar, the loop statements are executed once.

Counting using a For Statement

The MATLAB programs of Figures 2a, 2b, and 2c show three ways of counting from 2 to 20 by 2s using a `for` loop.

```
% counting from 2 to 20 by 2s
clear; clc;
for count = 2:2:20
    fprintf('%d ',count);
end
fprintf('\n');
```

Figure 2a. MATLAB Program to Count from 2 to 20 by 2s

The expression in the program of Figure 2a is `2:2:20` which creates a 1D array using the colon operator. The loop counts from 2 to 20 as follows:

- The loop control variable `count` is set to 2 (first column of the loop control array) for the first iteration of the loop.
- The loop body is executed. In this case the loop body is a single print statement.

- For all subsequent iterations, the loop control variable `count` is set to the value in the next column of the loop control array and the loop body statement is executed.
 - This is continued until the last column (the value 20) of the array is used.
- Once the loop has finished, the loop control array will no longer exist although the loop control variable `count` will hold 20; the last column of the loop control array.

```
% counting using pre-existing array
clear; clc;
countValues = 2:2:20;
for count = countValues
    fprintf('%d ', count);
end
fprintf('\n');
```

Figure 2b. MATLAB Program to Count from 2 to 20 by 2s

The expression in the program of Figure 2b is the variable `countValues` which is an existing 1D array. The loop counts from 2 to 20 as follows:

- The loop control variable `count` is set to 2 (first column of the loop control array which is a copy of the `countValues` array) for the first iteration of the loop.
- The loop body is executed. In this case the loop body is a single print statement.
- For all subsequent iterations, the loop control variable `count` is set to the value in the next column of the loop control array and the loop body statement is executed.
- This is continued until the last column (the value 20) of the array is used.

Once the loop has finished, the loop control array will no longer exist. The `countValues` array will still exist as it was created separately. The loop control variable `count` will hold 20; the last column of the loop control array.

```
% counting by extracting count values from array
clear; clc;
countValues = 2:2:20;
for k = 1:1:length(countValues)
    count = countValues(k);
    fprintf('%d ', count);
end
fprintf('\n');
```


Figure 2c. MATLAB Program to Count from 2 to 20 by 2s

The expression in the program of Figure 2c is `1:1:length(countValues)` which creates a 1D array of the indices of the array `countValues`. The array `countValues` is created before the loop. The loop counts from 2 to 20 as follows:

- The loop control variable `k` is set to 1 (first column of the loop control array which is an array of the indices of the `countValues` array) for the first iteration of the loop.
- The loop body which is two statements is executed. The count is extracted from the `countValues` array using indexing and then displayed.

- For all subsequent iterations, the loop control variable `k` is set to the value in the next column of the loop control array (next array index) and the loop body statement is executed.
 - This is continued until the last column (the index 10) of the array is used.
- Once the loop has finished, the loop control array containing the array indices will no longer exist. The `countValues` array will still exist as it was created separately. The loop control variable `k` will hold 10; the last column of the loop control array.

Last modified April 6, 2021

 [MATLAB Marina](#) is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.