I. Introduction to For Loops

Loops allow the repetition of segments of code to build up a calculation. They are one of the most useful tools in programming. This section demonstrates the basic syntax and nature of these loops.

Exercise: Create and run the following script. Notice the semicolons have been left off on purpose so it will display what is happening. What does it do?

```
% Program LoopTest.m
% This is a quick script file to look at what a simple for loop does.
% This loop simply prints the loop index value
for k=1:5
   k
end
```

Modify the code in the LoopTest.m script and test each of the following pieces of code. Notice only a small part of the code needs to be changed for each step.

1. For loop with simple indexing: Set up the simple loop below in your script, save it and then run the file by typing "LoopTest" at the command prompt.

```
x=5;
for k=1: x
    k
end
```

What are the different values taken on by k?

2. A for loop with step indexing: Now modify the loop by changing the part in red below. Run your loop again. What values of k does it take on? Why isn't 14 one of the values you see?

```
x = 14
for k=1:3:x
    k
end
```

3. A for loop with vector indexing: Modify the loop again with the values shown in red below. How does the loop perform in this case?

```
x = [200,400,260,600]
for k=x
    k
end
```

4. A for loop with 2-D array indexing: Add the values shown in red below to the x vector, be sure to include the semicolon as shown. What happens to the loop index in this case?

```
x=[200,400,260,600 ; 300,500,120,200]
for k=x
    k
end
```

Overview of For Loops in MATLAB

1. **English:**
   For an each successive column in the given input vector or matrix execute the following statements.

2. MATLAB General Structure

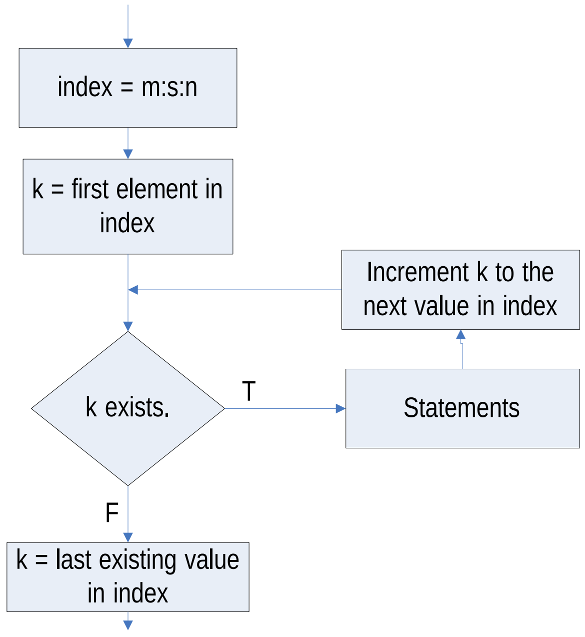
```
for k = m:s:n
        statements (that vary with k)
end
```

where:

k =	loop variable
m =	start
s =	increment
n =	stop

or k can equal any vector or 2D matrix for each iteration of the loop k is equal to the next column in the vector or matrix. Loop exits after last column is applied.

3. **Flow Chart**


index = m:s:n
k = first element in index
Increment k to the next value in index
k exists.
T
Statements
F
k = last existing value in index

4. **MATLAB Example (using fill logic):**
   A function that will test each element in a vector for its sign and return the square root for positive numbers and NaN for anything else.

```
function y=root2(x)
% lots of comments
for k = 1:length(x)
        if x(k)>=0
                y(k)=x(k)^0.5
        else
                y(k) = NaN
        end
end
```

Download and execute this example. This example is using the "for" loop to demonstrate "Fill Logic". Each iteration of the loop fills in another number in the output vector. Notice carefully how k is used to index:

- The input vector in the conditionals
- The input on the RHS of calculation
- The output on the LHS of calculation

In the conditional labs it was noticed that if statements will not operate on a vector in an element-by-element fashion. Nesting the "if" statement inside a for loop creates an element-by-element calculation.

Notes:

- Notice that in both the English sentence and in the loop there is an "if" contained inside the "loop".
- For loops always start with the word "for" and end with the word "end"
- Any number of statements may be contained in the loop.
- Indentation is not required but makes the program more readable and is completed automatically by the MATLAB editor when typing initially. When editing click the first indent icon (smart indent) on the home tab to get this indenting for edited programs.

II. Problem: Creating an element-by-element if Logic

The goal of this exercise is to modify the leap day function developed earlier so that it can take a vector of years as the input and return a vector of 1s and 0s (where 1s represent leap years). It will have the same input and output variables, but it will be able to take an input of a vector and output a vector of results.

Your tasks

a. Download and fill out the first page of the program development worksheet.

b. Nest the leap year conditional code inside a loop and add indexing as is done in the square root example in the previous section. Note for this transformation will require:
   i. Using a for loop.
   ii. Using the length function to determine the total number of loops.
   iii. Indexing the input vector for the logical test.
   iv. Indexing the output vector when assigning the 1 or 0.

c. Test the resulting code on a sample vector.

III. Two-dimensional Fill Loop

Loops can also be used for a fill type problem involving matrices. For a 2D matrix there will be two indices (e.g. the matrix M is addressed M(i, j) where i and j are integer index values). To fill a 2D matrix two loops are required, one inside the other (this is called “nested” loops).

To see how this works try out the following problem. ➔ **Important:** To be successful with this problem it is critical that you complete the hand solution very carefully and note in detail the exact steps you take including their order

<u>Problem: Set Up/ Introduction</u>

1. Problem Statement:
Develop a function that will fill in an n x m array where each cell is the sum of its row index and its column index (e.g. M(2, 3) = 2 + 3 = 5)

2. Inputs: (full name, variable to be used, units)

Variable Name	Description	Units or Values	Input Source*
n	*Number of rows in output array*	*NA*	*Command Line*
m	*Number of columns in output array*	*NA*	*Command Line*

* Possible sources: command line, file, interactive input

3. Output: (full name, variable to be used, units)

Variable Name	Description	Units or Values	Output type*
M	*The n x m output array of index sums*	*NA*	*Command Line*

* Possible types: command line, file, display

4. Solution Steps:
   a. Perform calculation on test case(s)
      (select a test case if one is not given, avoid answers of 0 or 1)
   b. Identify the steps/equations

<u>Hand calculate a test case for a 2 x 3 array (i.e., n = 2, m = 3</u>) – do this slowly and jot notes on the exact process you go through (e.g., what order do you go, what calc.s do you complete, when …)

		Column Index		
		1	2	3
Row Index	1			
	2			

Consider the steps you took to fill in this array including the order of calculations you completed (number them). List steps or draw a flow chart of what you had to do.

5. Write and test a program to complete this problem.

IV. **Aside: A Quiz Program** (optional but helpful and potentially fun).
This is an example multi-loop program
Download the files quiz1.m and A.mat to your working directory
Run quiz1 (i.e., type quiz1 and press enter in the command window. The take the quiz
Open the m-file in the MATLAB editor window and examine the code.
Can you figure out the loops the program is using to run this quiz?

V. Assignment (due next <u>regular</u> lab)
Complete a Program Development Worksheet (Setup, Coding & Validation) for each of the following:
   a. Write a program that takes an arbitrary vector of real numbers and returns a vector with negative numbers squared and positive numbers halved.
   b. Prepare a program to return the Julian Date (i.e., the number of days since the beginning of the year) given the current date (name of month, date in month, year). This program can be done with or without loops (but will require conditionals). It must consider leap days (use previous leap day function). To handle the months try a switch-case structure and switch based on only the first three letters of the month (e.g. if the variable month contains the name of the month compare month(1:3) to 'Jan', 'Feb', 'Mar' …
   c. Convert the grading program developed in class to handle vectors (i.e. make it so a vector of scores can be input and the program will return a vector of grades).
   d. Prepare a program that will return the terms of a Fibonacci Sequence given the first two values and the number of terms. In a Fibonacci Sequence each successive term is the sum of the two terms that come before it. For example the 6 term series with initial terms of 1 and 1 is:

      1, 1, 2, 3, 5, 8