MATLAB Tutorial 03: Loops & Functions

ENME 303 Computational Methods for Engineers

August Phelps
Slides adapted from Parham Oveissi (2023)

Last week

- Exporting from MATLAB
- Plotting

Agenda

- Loops
- If/Else Statements
- Functions

Loops (for loop)

- The for loop is a loop that executes a block of statements a specified number of times.

```
for  k = i : T : n
    commands f (k)
end
```

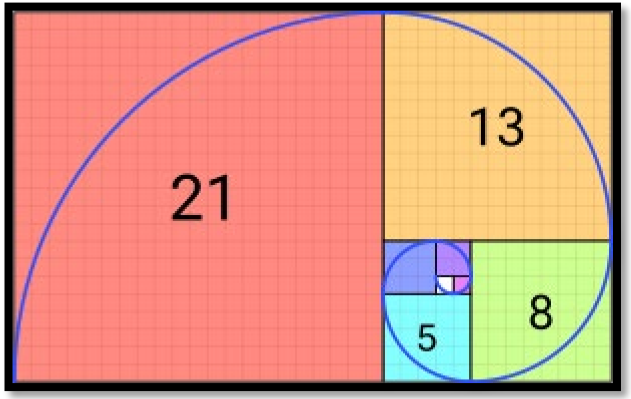
```
for i = 0:10
    disp(i)
end
```

```
for $k_1 = i_1 : T_1 : n_1$
    for $k_2 = i_2 : T_2 : n_2$
    .
    .
    commands f $(k_1, k_2)$
    .
    .
    end
end
```

For Loop Example: Golden Ratio

- Fibonacci Sequence :
  - 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, …
  - $x(1) = 1,\ \ x(2) = 2,\ x(n) = x(n-1) + x(n-2)$ for $n \geq 3$

- Golden Ratio:
  - $\frac{x(n)}{x(n-1)}$



if, else, elseif

```
if  logical expression
      commands
end
```

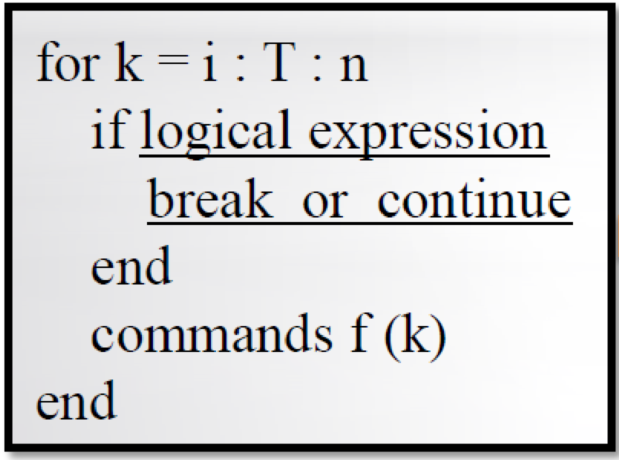
```
N = input('Enter a Number: ');
flag = 0;
if  N > 70
      flag = 1;
end
fprintf('flag is: %0.0f \n', flag);
```

```
if  logical expression
      commands 1
else
      commands 2
end
```

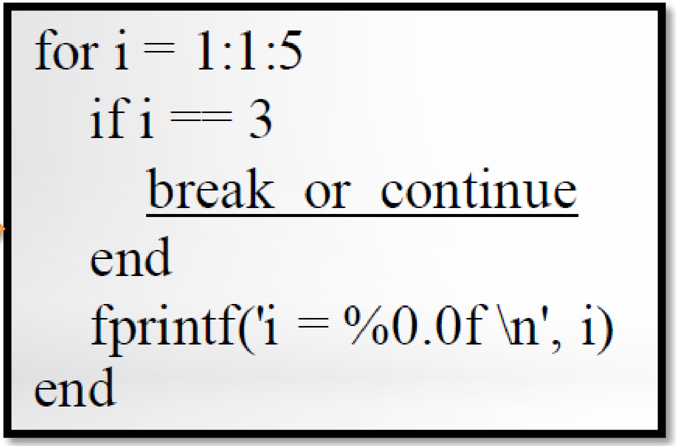
```
N = 10;
if   rem(N,2) == 0
      fprintf('Number is even \n');
else
      fprintf('Number is odd \n');
end
```

```
if  logical expression 1
      commands 1
elseif  logical expression 2
      commands 2
.
.
elseif  logical expression n-1
      commands n-1
else
      commands n
end
```

break and continue



```
for k = i : T : n
    if logical expression
        break  or  continue
    end
    commands f (k)
end
```



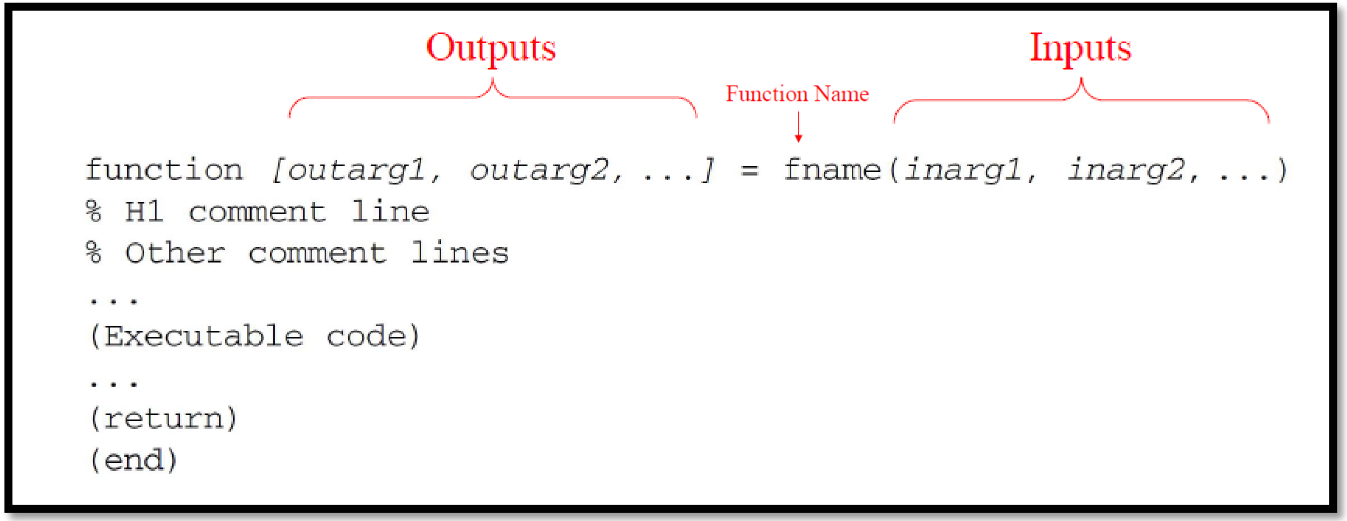
```
for i = 1:1:5
    if i == 3
        break  or  continue
    end
    fprintf('i = %0.0f \n', i)
end
```

Ways to Define Functions in MATLAB

- Function in a Script File
  - Function with One Output
  - Function with Multiple Outputs
- Function at the end of a Script File
- Multiple Functions in a Function File
- Anonymous Functions

Function in a Script

- Each ordinary MATLAB function should be placed in a file with the same name (including capitalization) as the function along with the file extension ".m". For example, if a function is named My_fun, that function should be placed in a file named My_fun.m.



```
function [outarg1, outarg2, ...] = fname(inarg1, inarg2, ...)
% H1 comment line
% Other comment lines
...
(Executable code)
...
(return)
(end)
```

Function in a Script

- A function is invoked by naming it in an expression together with a list of actual arguments. A function can be invoked by typing its name directly in the Command Window or by including it in a script file or another function.

```
function [outarg1, outarg2, ...] = fname(inarg1, inarg2, ...)
% H1 comment line
% Other comment lines
...
(Executable code)
...
(return)
(end)
```

```
>> fname (x ,y, …)
```

Function with One Output

Define a function in a file named calculateAverage.m

```
function ave = calculateAverage(x)
    ave = sum(x(:))/numel(x);
end
```

Invoking the function in a script saved in the same directory as the function file.

```
z = 1:99;
ave = calculateAverage(z)
```

```
ave =
    50
```

Function with Multiple Outputs

Define a function in a file named stat.m

```
function [m,s] = stat(x)
    n = length(x);
    m = sum(x)/n;
    s = sqrt(sum((x-m).^2/n));
end
```

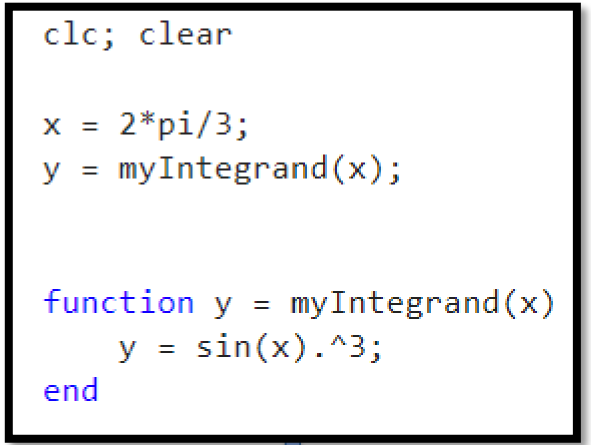
Invoking the function in a script saved in the same directory as the function file.

```
values = [12.7, 45.4, 98.9, 26.6, 53.1];
[ave,stdev] = stat(values)
```

```
ave =
   47.3400
stdev =
   29.4124
```

Function at the end of a Script File

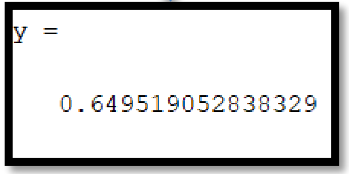
Defining and invoking the function in the same script.



```
clc; clear

x = 2*pi/3;
y = myIntegrand(x);


function y = myIntegrand(x)
    y = sin(x).^3;
end
```



```
y =

   0.649519052838329
```

Multiple Functions in a Function File

Define two functions in a file named stat2.m, where the first function calls the second.

Note that function avg is a local function. Local functions are only available to other functions within the same file.

```
function [m,s] = stat2(x)
    n = length(x);
    m = avg(x,n);
    s = sqrt(sum((x-m).^2/n));
end

function m = avg(x,n)
    m = sum(x)/n;
end
```

```
values = [12.7, 45.4, 98.9, 26.6, 53.1];
[ave,stdev] = stat2(values)
```

```
ave =
   47.3400
stdev =
   29.4124
```

Anonymous Functions

Anonymous functions allow you to define a function without creating a program file, as long as the function consists of a single statement. A common application of anonymous functions is to define a mathematical expression, and then evaluate that expression over a range of values.

fcn = @ (x) f(x)   fcn = @ (x, y, ...) ([f_1(x, y, ...); f_2(x, y, ...); ...])

```
sqr = @(x) x.^2;
```

```
a = sqr(5)
```

```
a =
  25
```

Function Practice: Absolute Value

- Create a function that calculates the absolute value of a variable
- Your function call should look like this:

```
function out = absoluteValue(a)
```

$$x = \begin{cases} x, & x \geq 0 \\ -x, & x < 0 \end{cases}$$

Function Example: Bisection Method

- Let’s create an algorithm to find a zero using bisection
- Start problem with pseudocode, then write in MATLAB

```
function [c, error] = bisectionMethod(f,a0,b0,numIterations,errorMax)
```

Thanks!