

MATH 3341: Introduction to Scientific Computing Lab

Libao Jin

University of Wyoming

February 12, 2020





Lab 03: Functions and Control Flows



The background features a large, faint watermark of the University of Wyoming seal. The seal is circular with a rope-like border. Inside the border, the words "UNIVERSITY OF WYOMING" are written in an arc at the top. In the center is an open book with a quill pen resting on it. Below the book, the word "EQUALITY" is written in an arc, and the year "1886" is at the bottom.

Anonymous Functions



An *anonymous function* is a function that does not have a function name, but is associated with a variable whose data type is `function_handle`. Anonymous functions can accept inputs and return outputs, just as standard functions do. To define the function $f(x) = x^2 + 1$ we use `f = @(x) x.^2 + 1`, where the inputs (parameters) are defined by the @ symbol in front of the list of variables in parenthesis.



Examples

- $f(y) = \sin(y)$
`f = @(y) sin(y)`
- $g(x, y) = x^2 + y^2 - 1$
`g = @(x, y) x.^2 + y.^2 - 1.`
- $h(z) = e^{\sin z} = e^{f(z)}$
`h = @(z) exp(f(z)).`



The background features a large, faint watermark of the University of Wyoming seal. The seal is circular with a rope-like border. Inside the border, the words "UNIVERSITY OF WYOMING" are written in an arc at the top. In the center is an open book with a quill pen resting on it. Below the book, the word "EQUALITY" is written in an arc, and the year "1886" is at the bottom.

Function Files



“Fun is where you find it. Look closely, and you can find it in functions.” Defining functions can save you from writing the same code over and over again. Here is the syntax to define a function:

```
function [output_args] = functionName(input_args)
% FUNCTIONNAME Summary of the function
% Details of the function goes here such as syntax, etc.

% function body goes here
end
```



Example: sumProd

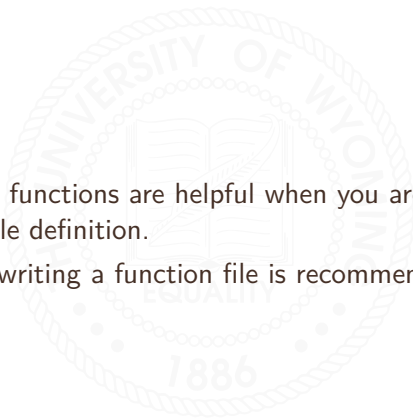
```
function [summation, product] = sumProd(x)
% SUMPROD Calculate the summation and product of
% all elements in x
% Syntax:
%   [summation, product] = sumProd(x)
%   summation = sumProd(x)

% Initialize variables summation and product
summation = 0;
product = 1;
for i = 1:length(x)
    summation = summation + x(i);
    product = product * x(i);
end
```



Which do I use?

- Anonymous functions are helpful when you are using functions with a simple definition.
- Otherwise, writing a function file is recommended.





Branching: **if-else** and **switch**



One of the keys to designing intelligent programs is to give them the ability to make decision. MATLAB provides the `if` and `switch` statements to implement decisions. The `if` comes in two forms: `if` and `if else`. The `if` statement directs a program to execute a statement or statement block if a test condition is true and to skip that statement or block if the condition is false.



Syntax

Run `help if` in the Command Window:

if Conditionally execute statements.

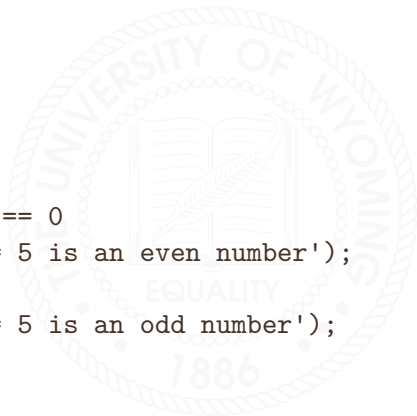
The general form of the if statement is

```
if expression
    statements
elseif expression
    statements
else
    statements
end
```



Examples

```
% Example 1
n = 5;
if mod(n, 2) == 0
    disp('n = 5 is an even number');
else
    disp('n = 5 is an odd number');
end
```



Examples

```
function is_leap_year = isLeapYear(year)

if mod(year, 400) == 0
    is_leap_year = true;
elseif mod(year, 4) == 0 && mod(year, 100) ~= 0
    is_leap_year = true;
else
    is_leap_year = false;
end

end
```



Examples

Make it shorter:

```
function is_leap_year = isLeapYear(year)

if mod(year, 400) == 0 || (mod(year, 4) == 0 && mod(year, 100) > 0)
    is_leap_year = true;
else
    is_leap_year = false;
end

end
```



switch statement

Run `help switch` in the Command Window:

`switch` Switch among several cases based on expression.
The general form of the switch statement is:

```
switch switch_expr
  case case_expr1,
        statements
  case {case_expr2, case_expr3, ..., case_exprN}
        statements
  ...
  otherwise,
        statements
end
```



Examples

```
function dayOfWeek1(d)

switch d
    case {'Monday', 'Tuesday', 'Wednesday', ...
          'Thursday', 'Friday'}
        fprintf('%s is weekday.\n', d)
    otherwise
        fprintf('%s is weekend.\n', d)
end

end
```



Examples

```
function dayOfWeek2(d)

switch d
    case {'Monday', 'Tuesday', 'Wednesday', ...
          'Thursday', 'Friday'}
        fprintf('%s is weekday.\n', d)
    case {'Saturday', 'Sunday'}
        fprintf('%s is weekend.\n', d)
    otherwise
        fprintf('Error!\n')
end

end
```



Relational Operators

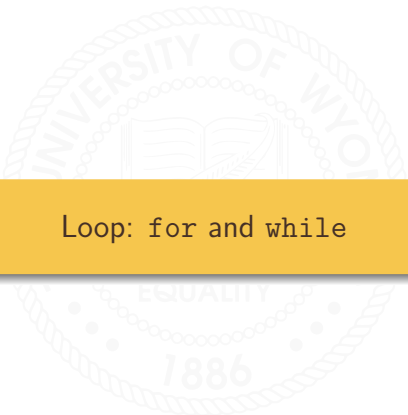
| Symbol | Meaning |
|--------------------|--------------------------|
| <code>==</code> | equal to |
| <code>~=</code> | not equal |
| <code>></code> | greater than |
| <code><</code> | less than |
| <code>>=</code> | greater than or equal to |
| <code><=</code> | less than or equal to |



Logical Operators

| Symbol | Meaning |
|-------------------------|---------------------------|
| <code>&</code> | element-wise logical AND |
| <code> </code> | element-wise logical OR |
| <code>&&</code> | short-circuit logical AND |
| <code> </code> | short-circuit logical OR |
| <code>~</code> | logical NOT |



The background features a large, faint watermark of the University of Wyoming seal. The seal is circular with a rope-like border. Inside the border, the words "UNIVERSITY OF WYOMING" are written in an arc at the top. In the center is an open book with a quill pen resting on it. Below the book, the word "EQUALITY" is written in an arc, and the year "1886" is at the bottom.

Loop: for and while



Question: What should we do if we want to `disp('Repeating is BORING!')` for 100 times?

```
disp('Repeating is BORING!')  
disp('Repeating is BORING!')  
disp('Repeating is BORING!')  
...  
disp('Repeating is BORING!')
```



Better Approach: Using for or while Loop

Run `help for` in the Command Window:

`for` Repeat statements a specific number of times.

The general form of a `for` statement is:

```
for loop_counter = expr
    statements
end
```



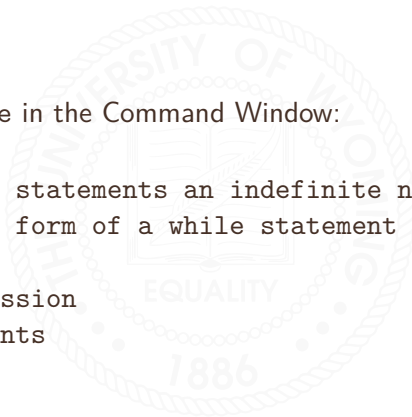
Better Approach: Using for or while Loop

Run `help while` in the Command Window:

`while` Repeat statements an indefinite number of times.

The general form of a while statement is:

```
while expression
    statements
end
```



Problem solved!

Using for loop:

```
for i = 1:100
    disp('Repeating is BORING!')
end
```

Using while loop:

```
i = 1;
while i <= 100
    disp('Repeating is BORING!')
    i = i + 1;
end
```

