

`while` **Loops** in MATLAB

Gerald W. Recktenwald
Department of Mechanical Engineering
Portland State University
`gerry@pdx.edu`

Loops in MATLAB

Repetition or Looping

A sequence of calculations is repeated until *either*

1. All elements in a vector or matrix have been processed

or

2. The calculations have produced a result that meets a predetermined termination criterion

Looping is achieved with `for` loops and `while` loops.

while loops (1)

while loops are most often used when an iteration is repeated until some termination criterion is met.

Syntax:

```
while expression
    block of statements
end
```

The *block of statements* is executed as long as *expression* is true.

expression is a MATLAB expression that evaluates to true or false

Example:

```
>> x = 3; y = 5;
>> x < y
ans =
    1
```

In MATLAB false and 0 are equivalent, and true and 1 are equivalent.

while loops (2)

More Examples

```
>> x = 3; y = 5;  
>> x > y  
ans =  
    0
```

```
>> x == 0  
ans =  
    0
```

```
>> abs(x-y) > 0  
ans =  
    1
```

Review: Relational Operators

Relational operators are used in comparing two values.

Operator	Meaning
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	equal to
~=	not equal to

The result of applying a relational operator is a logical value, i.e. the result is either *true* or *false*.

In MATLAB any nonzero value, including a non-empty string, is equivalent to *true*. Only zero is equivalent to *false*.

while loops (3)

Practice: What is the value of x at the end of this code snippet?

```
x = 10;
fprintf('\nAt start of loop, x = %12.8f\n',x);

while x > 0
    x = x - 3;
    fprintf('%12.8f\n',x);
end

fprintf('At end of loop, x = %12.8f\n',x);
```

Loop #	Value of x
before loop	10
1	7
after loop	

while loops (4)

Example: : Newton's method for evaluating \sqrt{x}

$$r_k = \frac{1}{2} \left(r_{k-1} + \frac{x}{r_{k-1}} \right)$$

```
r = ...           % initialize
rold = ...
while abs(rold-r) > delta
    rold = r;
    r = 0.5*(rold + x/rold);
end
```

Review: Logical Operators

Logical operators are used to combine logical expressions with logical “and” or logical “or”, or to reverse a logical value with “not”

Operator	Meaning
&	and
	or
~	not

Examples:

```
>> a = 2; b = 4;
>> aIsSmaller = a < b;
>> bIsSmaller = b < a;
>> bothTrue = aIsSmaller & bIsSmaller
bothTrue =
    0
```

```
>> eitherTrue = aIsSmaller | bIsSmaller
eitherTrue =
    1
```

```
>> ~eitherTrue
ans =
    0
```


while loops (5)

It is (almost) always a good idea to put a limit on the number of iterations to be performed by a `while` loop.

An improvement on the preceding loop,

```
maxit = 25;
it = 0;
while abs(rolld-r) > delta & it<maxit
    rold = r;
    r = 0.5*(rold + x/rold);
    it = it + 1;
end
```

`while` loops (6)

The `break` and `return` statements provide an alternative way to exit from a loop construct. `break` and `return` may be applied to `for` loops or `while` loops.

`break` is used to escape from an enclosing `while` or `for` loop. Execution continues at the end of the enclosing loop construct.

`return` is used to force an exit from a function. This can have the effect of escaping from a loop. Any statements following the loop that are in the function body are skipped.

The break command

Example: : Escape from a while loop

```
function k = breakDemo(n)
% breakDemo Show how the "break" command causes exit from a while loop.
%           Search a random vector to find index
%           of first element greater than 0.8.
%
% Synopsis: k = breakDemo(n)
%
% Input:    n = size of random vector to be generated
%
% Output:   k = first (smallest) index in x such that x(k)>0.8
x = rand(1,n);
k = 1;
while k<=n
    if x(k)>0.8
        break
    end
    k = k + 1;
end
fprintf('x(k)=%f   for k = %d   n = %d\n',x(k),k,n);

% What happens if loop terminates without finding x(k)>0.8 ?
```

The return command

Example: : Return from within the body of a function

```
function k = returnDemo(n)
% returnDemo Show how the "return" command causes exit from a function.
%           Search a random vector to find the index of the first
%           element greater than 0.8.
%
% Synopsis: k = returnDemo(n)
%
% Input:    n = size of random vector to be generated
%
% Output:   k = first (smallest) index in x
%           such that x(k)>0.8
x = rand(1,n);
k = 1;

while k<=n
    if x(k)>0.8
        return
    end
    k = k + 1;
end

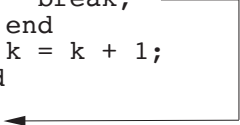
% What happens if loop terminates without finding x(k)>0.8 ?
```

Comparison of break and return

break is used to escape the current while or for loop.

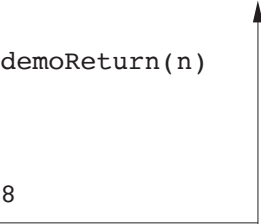
return is used to escape the current function.

```
function k = demoBreak(n)
...
while k<=n
    if x(k)>0.8
        break;
    end
    k = k + 1;
end
```



jump to end of enclosing
“while ... end” block

```
function k = demoReturn(n)
...
while k<=n
    if x(k)>0.8
        return;
    end
    k = k + 1;
end
```



return to calling
function