A Quick Introduction to Loops in MATLAB

Loops are used to repeat sequences of calculations. In MATLAB, loops can be implemented with a `for ...end` construct or a `while ...end` construct. In terms of their ability to repeat a series of calculations, `for` loops and `while` loops are equivalent.

`for` Loops

`for` loops are often used when a sequence of operations is to be performed a predetermined number of times. For example computing the average of a list of numbers requires adding up a known number of values.

Syntax

Loop counter incremented by one:

```
for i = startValue:endValue
  x = ...
  y = ...
  .
  .
  .
end
```

`i` is the *loop counter*. On the first pass through the loop, `i` is set to *`startValue`*. On the second pass through the loop `i` is set to *`startValue`*+1. The MATLAB statements between the `for` and the `end` are evaluated until `i`>*`endValue`*

Example 1 Print the square root of the first n integers

```
n = 5;
for i=1:n
  fprintf('%6d  %8.4f\n',i,sqrt(i));
end
```

See pp. 102–105 for a description of the `fprintf` statement.

□

Loop counter incremented by specified amount:

```
for i = startValue:increment:endValue
  x = ...
  y = ...
  .
  .
  .
end
```

The *`increment`* can be any positive or negative number

Example 2 Print the square root of the even integers up to n

```
n = 10;
for i=2:2:n
  fprintf('%6d  %8.4f\n',i,sqrt(i));
end
```

What happens when `n` = 9 or `n` = 11? □

Increments can be positive

```
for i = 0:2:10
  ...
end
```

or negative

```
for i = 5:-1:-5
  ...
end
```

The *startValue*, *increment*, and *endValue* parameters do not need to be integers

Example 3 Print the sine and cosine of a list of angles

```
for a=0:pi/6:pi
  d = a*180/pi;   %  convert to degrees
  fprintf('%8.3f %8.1f %9.4f %9.4f\n',a,d,sin(a),cos(a));
end
```

You could add a title row to this table by inserting

```
fprintf(' a (rad)   d (deg)   sin(a)    cos(a)\n')
```

before the start of the `for` loop. ——— □

Pre- and Post-loop Processing

Many loops involve manipulating quantities that are defined before the loop begins.

Example 4 Compute the sum of the first n integers

```
n = 10;
s = 0;
for i=1:n
  s = s + i;
end
```

The variable `s` must exist, and have a meaningful value before the loop begins. Otherwise the expression `s + i` cannot be evaluated.

The expression `s = s + i` is not a mathematical equation, it is an assignment. Mentally replace the "=" sign with an assignment arrow like "←".

$$\texttt{s} = \texttt{s} + \texttt{i} \qquad \text{means} \qquad \texttt{s} \leftarrow \texttt{s} + \texttt{i}$$

The statement `s = 0` is called an *initialization* of `s` because it gives `s` its initial value before the loop starts. ——— □

Loops can involve many repetitions, so printing during each pass through a loop is often impractical and undesirable. In some cases, a message or other clean-up work is done after the loop is finished.

Example 5 Compute the average of a list of numbers

```
n = 500;
x = rand(1,n);
s = 0;
for i=1:n
  s = s + x(i);
end
xbar = s/n;
```

The expression `x = rand(1,n)` creates a row vector of n pseudo-random numbers. The expression `s = s + x(i)` adds the ith element of `x` to the sum. As in Example 4, an initial value of `s` must be assigned before the loop starts. The average value (`xbar`) can only be computed after the loop is finished. □

