Pattern for doing something \( n \) times

\[
\begin{align*}
n &= \_\_\_\_ \\
\text{for } k &= 1:n \\
% \text{code to do} \\
% \text{that something} \\
\text{end}
\end{align*}
\]

Syntax of the \texttt{for} loop

\[
\text{for } <\text{var}> = <\text{start value}>:<\text{incr}>:<\text{end bound}> \\
\begin{align*}
\text{statements to be executed repeatedly} \\
\text{end}
\end{align*}
\]

Loop header specifies all the values that the index variable will take on, one for each pass of the loop. E.g., \( k = 3:1:7 \) means \( k \) will take on the values 3, 4, 5, 6, 7, one at a time.

\textbf{for} loop examples

\[
\begin{align*}
\text{for } k &= 2:0.5:3 \quad \text{disp}(k) \\
\text{end} \\
\text{for } k &= 1:4 \quad \text{disp}(k) \\
\text{end} \\
\text{for } k &= 0:-2:-6 \quad \text{disp}(k) \\
\text{end} \\
\text{for } k &= 0:-2:-7 \quad \text{disp}(k) \\
\text{end} \\
\text{for } k &= 5:2:1 \quad \text{disp}(k) \\
\text{end}
\end{align*}
\]

In the loop body, never change the value of the loop variable

\[
\begin{align*}
n &= \_\_\_\_ \\
\text{for } k &= 1:n \\
\% \text{code to do} \\
\% \text{that something} \\
\text{end}
\end{align*}
\]
Another way to estimate $\pi$

$$1 + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \cdots \approx \frac{\pi^2}{6}$$

Write a script to sum 100 terms of the above series and then calculate $\pi$.

### Drawing ASCII diagrams

Print these diagrams on the Command Window:

```
*******
*******
*******
*******
```

Printing is done left to right and top to bottom.

What is a simpler (sub)problem?

→ Print just one row of asterisks.

What is an even simpler (sub)problem?

→ Print just one single asterisk!

---

```matlab
function printRepeatChar(theChar,n)
% Print to the Command Window the
% character in variable theChar n times.
% n is non-negative integer.
% Add a linebreak.

for k= 1:n
    fprintf('%c', theChar)
end
fprintf('
')
end
```

---

**Keyword to indicate this is a function file (not a script)**

```matlab
function printRepeatChar(theChar,n)
% Print to the Command Window the
% character in variable theChar n times.
% n is non-negative integer.
% Add a linebreak.

for k= 1:n
    fprintf('%c', theChar)
end
fprintf('
')
end
```

---

**Function name, same as the file name**

```matlab
function printRepeatChar(theChar,n)
% Print to the Command Window the
% character in variable theChar n times.
% n is non-negative integer.
% Add a linebreak.

for k= 1:n
    fprintf('%c', theChar)
end
fprintf('
')
end
```

---

**Parameters, also called input parameters**

```matlab
function printRepeatChar(theChar,n)
% Print to the Command Window the
% character in variable theChar n times.
% n is non-negative integer.
% Add a linebreak.

for k= 1:n
    fprintf('%c', theChar)
end
fprintf('
')
end
```

---

**Function header**

```matlab
function printRepeatChar(theChar,n)
% Print to the Command Window the
% character in variable theChar n times.
% n is non-negative integer.
% Add a linebreak.

for k= 1:n
    fprintf('%c', theChar)
end
fprintf('
')
end
```
Function comments, or specifications
(VERY IMPORTANT to have this comment)

```matlab
function printRepeatChar(theChar, n)
% Print to the Command Window the
% character in variable theChar n times.
% n is non-negative integer.
% Add a linebreak.

for k = 1:n
    fprintf('%c', theChar)
end
fprintf('
')
```

Function body

Calling a function (invoking a function)

- I have a function with this header:
  ```matlab
  function printRepeatChar(theChar, n)
  ```
- I will call function printRepeatChar like this:
  ```matlab
  printRepeatChar('*', 8)
  ```
- The function header tells you everything you need to know about how to call the function.

Drawing ASCII diagrams

```
******
******
******
******

% printRectangle
for r = 1:4
    % Print rth row
    printRepeatChar('*', 7)
end
```

Given this function:

```matlab
function m = convertLength(ft, in)
% Convert length from feet (ft) and inches (in) to meters (m).
.
How many proper calls to convertLength are shown below?
```

```
f = ...; n = ...;
d = convertLength(f, n);
d = convertLength(f*12+n);
d = convertLength(f+n/12);
x = min(convertLength(f, n), 1);
y = convertLength(pi*(f+n/12)^2);
```

```
A: 1    B: 2    C: 3    D: 4    E: 5 or 0
```

General form of a user-defined function

```matlab
function [out1, out2, ...] = functionName(in1, in2, ...)
% 1-line comment to describe the function
% Additional description of function

Executable code that at some point assigns
values to output parameters out1, out2, ...
```

- `in1, in2, ...` are defined when the function begins execution. Variables `in1, in2, ...` are called function parameters and they hold the function arguments used when the function is invoked (called).
- `out1, out2, ...` are not defined until the executable code in the function assigns values to them.

My twinkling stars ...

- What is the algorithm? First, what are the individual tasks?
Implement this function

```matlab
function r = randReal(lo,hi)
% r is a random real number in the
% interval (lo,hi)
```

Implement this function

```matlab
function [x,y] = randPt(lo,hi)
% x and y are random real numbers in the
% interval (lo,hi)
```