9.1 Use a `for` loop to sum the elements in the following vector:

$$x = [1,\ 23,\ 43,\ 72,\ 87,\ 56,\ 98,\ 33]$$

Check your answer with the `sum` function.

9.4 Use a `while` loop to create a vector of the squares of the numbers 1 through 5.

9.6 A Fibonacci sequence is composed of elements created by adding the two previous elements. The simplest Fibonacci sequence starts with 1, 1, and proceeds as follows:

$$1,\ 1,\ 2,\ 3,\ 5,\ 8,\ 13,\ \ldots$$

However, a Fibonacci sequence can be created with any two starting numbers. Fibonacci sequences appear regularly in nature. For example, the shell of the chambered nautilus (see **Figure P9.6**) grows in accordance with a Fibonacci sequence.

Figure P9.6
Chambered nautilus. (Colin Keates © Dorling Kindersley, Courtesy of the natural history museum, London.)

Prompt the user to enter the first two numbers in a Fibonacci sequence, and the total number of elements requested for the sequence. Find the sequence and store it in an array by using a `for` loop. Now plot your results on a `polarplot` graph. Use the element number for the angle, and the value of the element in the sequence for the radius.

9.13 Edmond Halley (the astronomer famous for discovering Halley's comet) invented a fast algorithm for computing the square root of a number, *A*. Halley's algorithm approximates $\sqrt{A}$ as follows:
Start with an initial guess x_1. The new approximation is then given by

$$\begin{aligned} Y_n &= \frac{1}{A}x_n^2 \\ x_{n+1} &= \frac{x_n}{8}\Big(15 - y_n\big(10 - 3y_n\big)\Big) \end{aligned}$$

These two calculations are repeated until some convergence criterion,

$$\varepsilon,$$

is met.

$$|x_{n+1} - x_n| \le \varepsilon$$

Write a MATLAB® function called `my_sqrt` that approximates the square root of a number. It should have two inputs, the initial guess and the convergence criterion. Test your function by approximating the square root of 5 and comparing it to the value calculated with the built-in MATLAB® function, `sqrt`.

NOTE: There is a typo in the textbook: there should be 3 inputs to the function:

1. the number you want to square root
2. the initial guess
3. the convergence criteria

Extra Credit

9.21 Consider the following method to approximate the mathematical constant, ***e***. Start by generating *K* uniform random integers between 1 and *K*. Compute *J*, the number of integers between 1 and *K*, which were never generated. We then approximate ***e*** by the ratio

$$\frac{K}{J}$$

Consider the following example for $K = 5$. Assume that the following five integers are randomly generated between 1 and 5.

$$1 \quad 1 \quad 2 \quad 3 \quad 2$$

The number of times the integers are generated is given by

Integers	1	2	3	4	5
Number of instances	2	2	1	0	0

In this example, there are two integers, namely 4 and 5, which were never generated. This means that $J = 2$. Consequently, ***e*** is approximated by

$$\frac{5}{2} = 2.5$$

Write a function called `eapprox` that takes the value of *K* as input, and which then approximates ***e*** using the method described above. Test your function several times with different values of *K*, and compare the result to the value of ***e*** calculated using the built-in MATLAB® function.

```
exp(1)
```

HINT

Use the randi function to create an array of random integers.