Topics: Iteration using `while`, 1-dimensional array

Reading (ML): Sec 4.1, revisit Sec 2.1–2.4, 2.8 for discussion on 1-d array (exclude matrix and matrix operations)

Iteration

Important features:

- Task can be accomplished if some step is repeated a number of times
- Must be able to quantify success ⇒ ____________________
- Must have a starting point
- Must keep track of progress ⇒ ____________________

Syntax of the `while` Loop

```
while expression
    statements to execute if
    expression evaluates to true
end
```

Example 1: Average

Write a program that prompts the user for 10 numbers and then print the average. Use only scalar variables.

Pattern for doing something n times

```
i = 1;
while i<=n
   % do something
   % ...
   i = i + 1;
end
```

```
% Average 10 numbers from user input

n = 10;     % number of data values
total = 0;  % current sum (initialized to zero)
i = 1;      % initialize counter
while (i<=n)
  % read and process input value
    num = input('Enter a number: ');
    total = total + num;
  % update
    i = i + 1;
end
ave = total/n  % average of n numbers
```

Example 2: Running average

Write a program that repeatedly: (a) prompts the user for a number; (b) prints the average of previously entered numbers. The user enters 10 numbers in total. Again use only scalar variables.

```
% Running average of 10 numbers from user input

n = 10;      % number of data values
total = 0;  % current sum (initialized to zero)
i = 1;      % initialize counter
while (i<=n)
  % process input
    num = input('Enter a number: ');
    total = total + num;
    runningAve = total/i;  % running average
    disp(['Running average is ' num2str(runningAve)])
  % update
    i = i + 1;
end
```

Example 3: Indefinite iteration

What if the total number of entries is not known in advance? Write another program for calculating running averages. The user enters -9999 to indicate the end of data entry.

Pattern for doing something an indefinite number of times

```
% initialization
% ...
while not stopping signal
   % do something
   % ...
   % update status (variables)
   % ...
end
```

```
% Running averages numbers from user input
% User terminates input by entering -9999

endSignal = -9999;  % Ending signal from user
total = 0;  % current sum (initialized to zero)
i = 0;      % number of data entries so far
num = input('Enter a number (-9999 to quit): ');
while (num ~= endSignal)
  % process data
    i = i + 1;
    total = total + num;
    disp(['current average is ' num2str(total/i)])
  % update
    num = input('Enter a number (-9999 to quit): ');
end
```

1-Dimensional Array: Vector

An array is a *named* collection of data values organized into rows and/or columns. A 1-d array is a row or a column, also known as a *vector*. An *index* identifies the position of a value in the vector.

Suppose vector **v** is a collection of 4 values, i.e., vector **v** has 4 cells.

The *i*th value can be accessed as **v(i)**.

Assign a value of 9 to into the 4th cell of vector **v**: **v(4) = 9**.

Copy the value in the 4th cell to the 2nd cell of vector **v**: **v(2) = v(4)**.

Copy the value in the current cell to the next cell of vector **v**: **v(i+1) = v(i)**.

Array Initialization

MATLAB function **zeros**: **vecA = zeros(1,5)**
MATLAB function **ones**: **vecB = ones(1,5)**
"Manual": **vecC(5) = 10**

Can you write a program for calculating an average (Example 1) that stores all the data entered by the user?