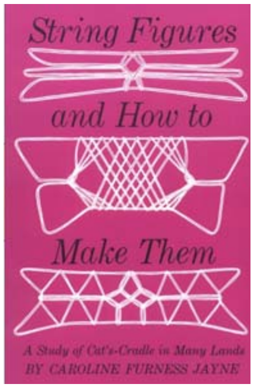

String Figures
and How to
Make Them
A Study of Cat's-Cradle in Many Lands
BY CAROLINE FURNESS JAYNE

More on Vectors
&
Strings

Lecture 12 (Feb 28)
CS100M – Spring 2008

Topics for Today

- Vector-related functions
  - length, zeros, ones, std
  - Revisit: rand, randn, max
- String related functions
  - isletter, isspace, lower, upper, ischar
- Row and column vectors
- Strings

Special Functions for Creating Vectors

- Some vectors are used so often that there are special functions for creating them

```
zeros(1, 5)  % A vector of length 5 holding all zeros
                                          0    0    0    0    0

ones(1, 3)   % A vector of length 3 holding all ones
                                                  1    1    1

rand(1, 4)   % A vector of length 4 holding random numbers
                          0.9501   0.2311   0.6068   0.4860
```

Why the extra arguments?

- Matlab (= Matrix Laboratory) uses matrices (2D arrays) as its default
  - Thus, zeros(3, 4) produces a 3-by-4 matrix of zeros

```
0 0 0 0
0 0 0 0
0 0 0 0
```

  - zeros(1, 5) produces a 1-by-5 matrix (i.e., a single row of a matrix; also called a *row vector*)

```
0 0 0 0 0
```

  - zeros(5, 1) produces a 5-by-1 matrix (i.e., a single column of a matrix; also called a *column vector*)

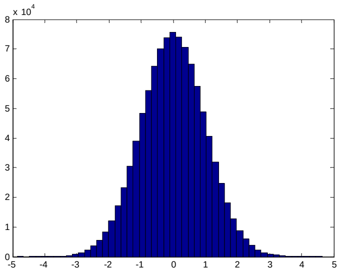
```
0
0
0
0
0
```

Row and Column Vectors

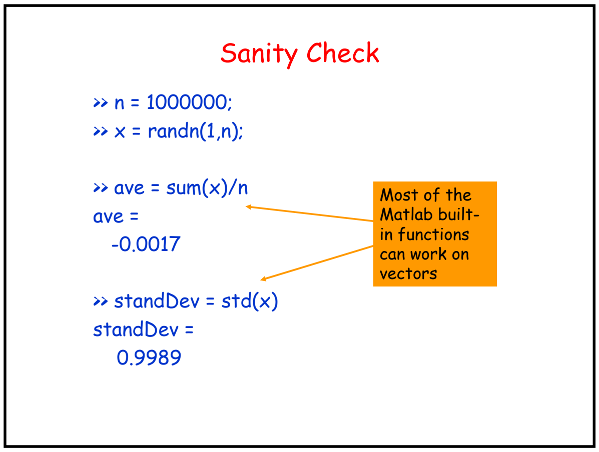
```
>> v = [1 2 3]

v =
     1     2     3

>> v = [1 ; 2 ; 3]

v =
     1
     2
     3
```

Note the semicolons

Normal Distribution with Zero Mean and Unit STD



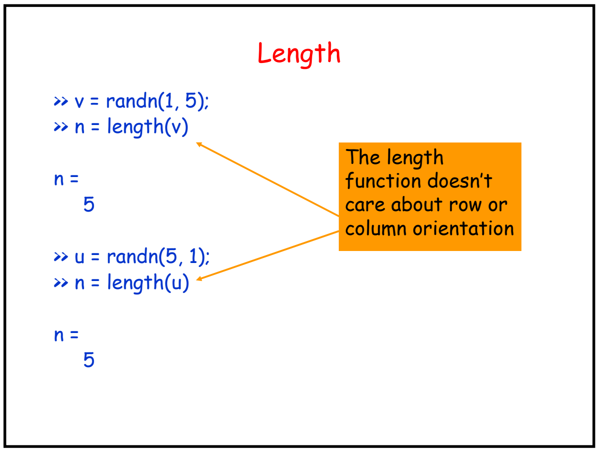
Distribution of randn(1,1000000)



Sanity Check

```
>> n = 1000000;
>> x = randn(1,n);

>> ave = sum(x)/n
ave =
   -0.0017

>> standDev = std(x)
standDev =
    0.9989
```

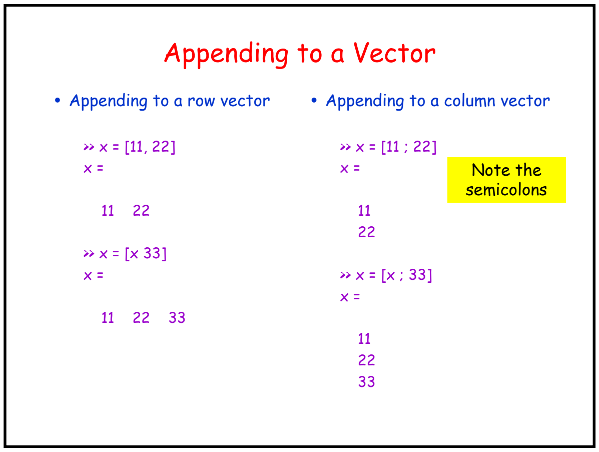
Most of the Matlab built-in functions can work on vectors



Length

```
>> v = randn(1, 5);
>> n = length(v)

n =
     5

>> u = randn(5, 1);
>> n = length(u)

n =
     5
```

The length function doesn't care about row or column orientation



Appending to a Vector

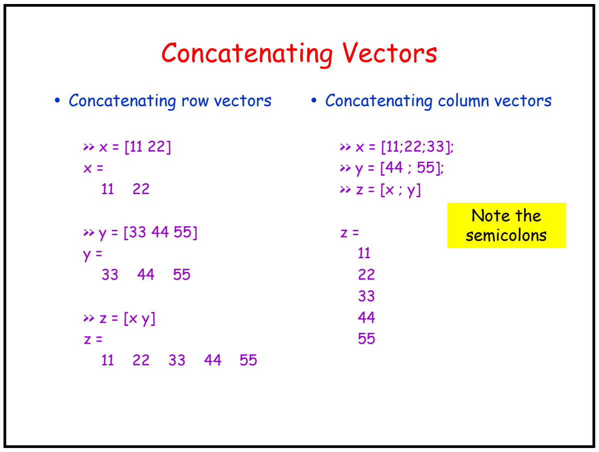
- Appending to a row vector

```
>> x = [11, 22]
x =

   11    22

>> x = [x 33]
x =

   11    22    33
```

- Appending to a column vector

```
>> x = [11 ; 22]
x =

   11
   22

>> x = [x ; 33]
x =

   11
   22
   33
```

Note the semicolons



Concatenating Vectors

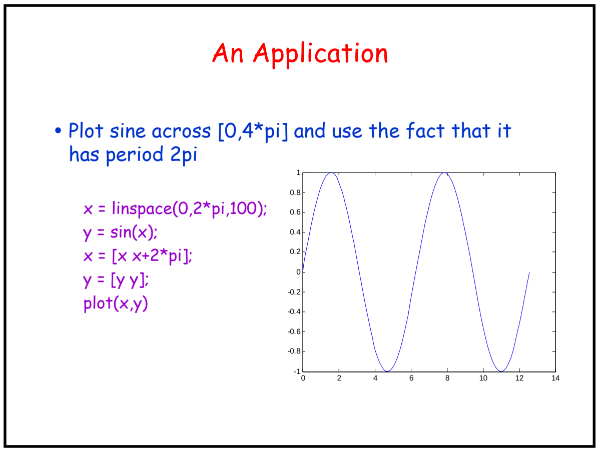
- Concatenating row vectors

```
>> x = [11 22]
x =
   11    22

>> y = [33 44 55]
y =
   33    44    55

>> z = [x y]
z =
   11    22    33    44    55
```

- Concatenating column vectors

```
>> x = [11;22;33];
>> y = [44 ; 55];
>> z = [x ; y]

z =
   11
   22
   33
   44
   55
```

Note the semicolons

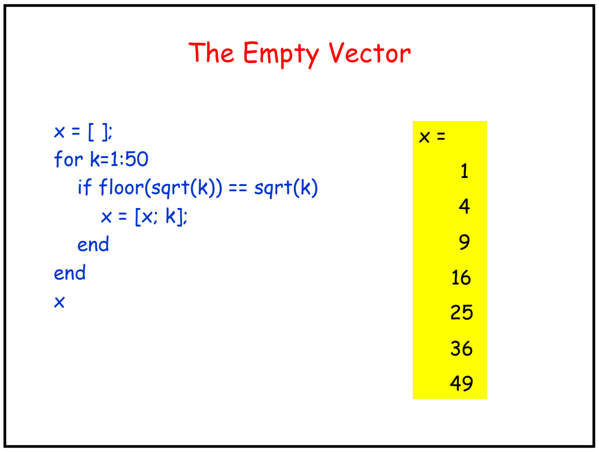


An Application

- Plot sine across [0,4*pi] and use the fact that it has period 2pi

```
x = linspace(0,2*pi,100);
y = sin(x);
x = [x x+2*pi];
y = [y y];
plot(x,y)
```

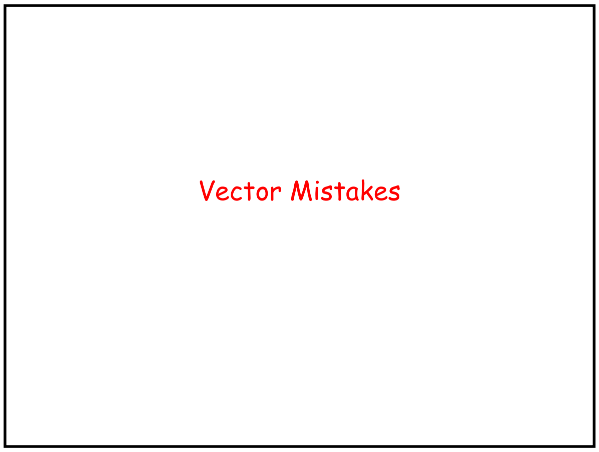
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
2
4
6
8
10
12
14



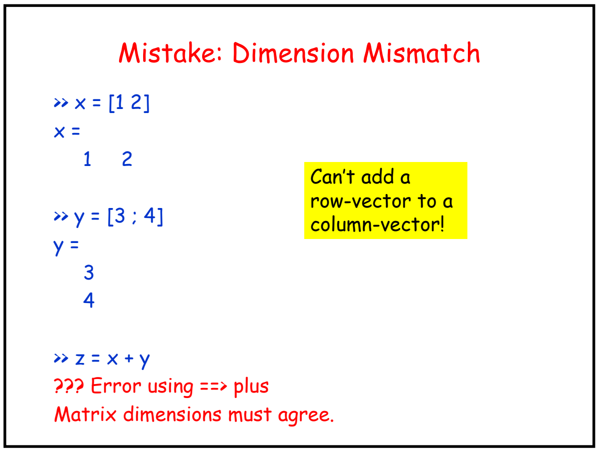
The Empty Vector

```
x = [ ];
for k=1:50
   if floor(sqrt(k)) == sqrt(k)
      x = [x; k];
   end
end
x
```

```
x =
     1
     4
     9
    16
    25
    36
    49
```



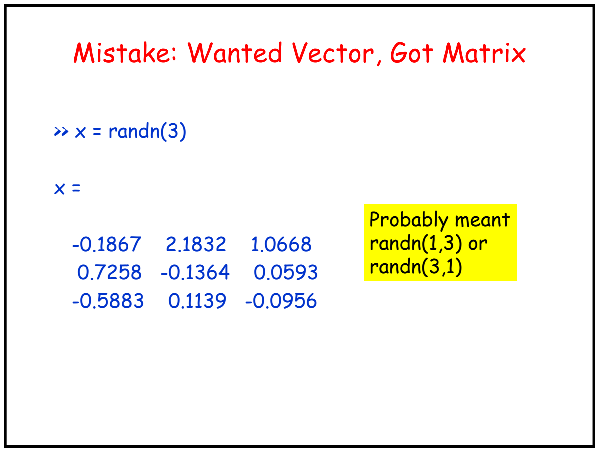
Vector Mistakes



Mistake: Dimension Mismatch

```
>> x = [1 2]
x =
    1    2

>> y = [3 ; 4]
y =
    3
    4

>> z = x + y
??? Error using ==> plus
Matrix dimensions must agree.
```

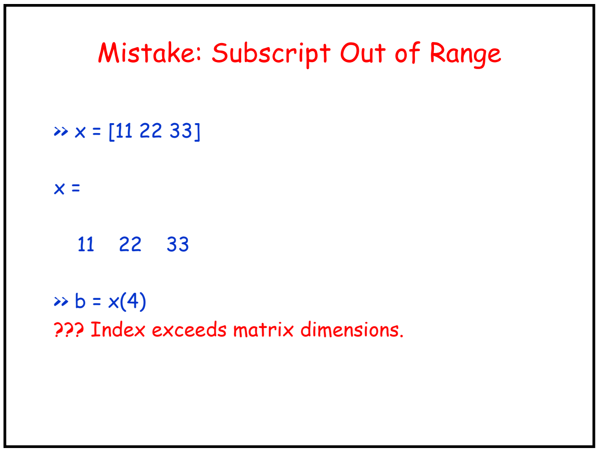
Can't add a row-vector to a column-vector!



Mistake: Wanted Vector, Got Matrix

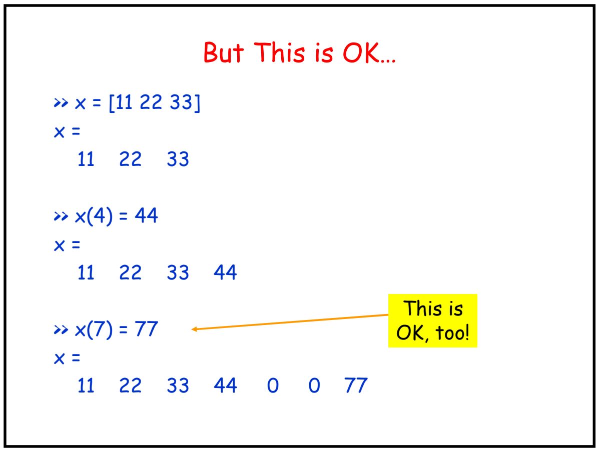
```
>> x = randn(3)

x =

  -0.1867    2.1832    1.0668
   0.7258   -0.1364    0.0593
  -0.5883    0.1139   -0.0956
```

Probably meant randn(1,3) or randn(3,1)



Mistake: Subscript Out of Range

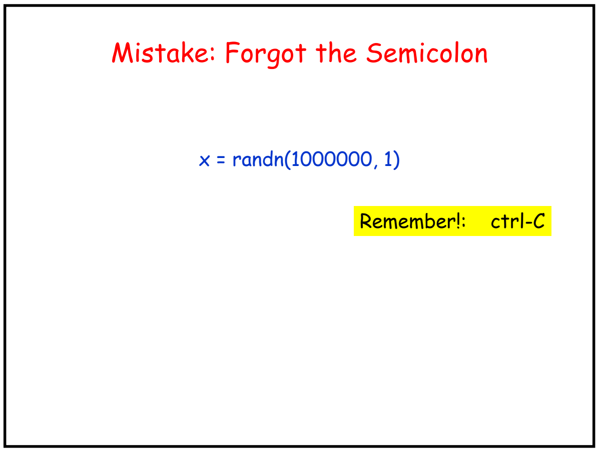
```
>> x = [11 22 33]

x =

   11    22    33

>> b = x(4)
??? Index exceeds matrix dimensions.
```



But This is OK…

```
>> x = [11 22 33]
x =
   11    22    33

>> x(4) = 44
x =
   11    22    33    44

>> x(7) = 77
x =
   11    22    33    44     0     0    77
```

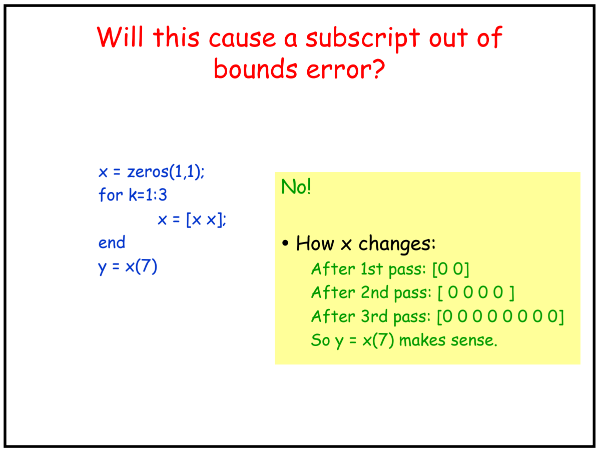
This is OK, too!



Mistake: Forgot the Semicolon

```
x = randn(1000000, 1)
```

Remember!:   ctrl-C

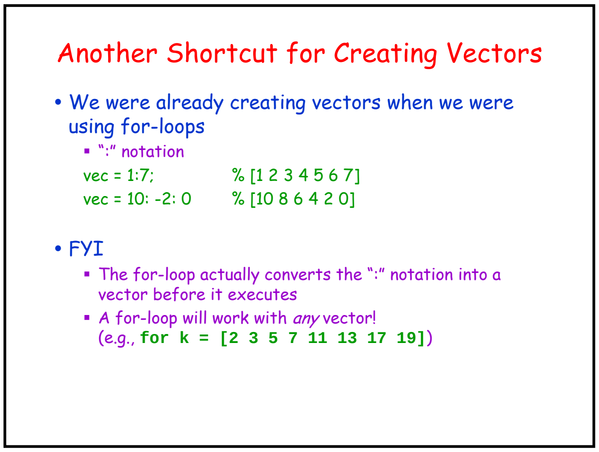


Will this cause a subscript out of bounds error?

```
x = zeros(1,1);
for k=1:3
        x = [x x];
end
y = x(7)
```

No!

- How x changes:
  - After 1st pass: [0 0]
  - After 2nd pass: [0 0 0 0]
  - After 3rd pass: [0 0 0 0 0 0 0 0]
  - So y = x(7) makes sense.

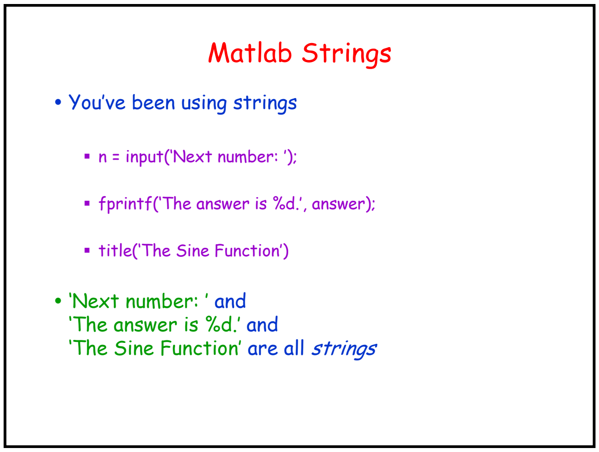


Another Shortcut for Creating Vectors

- We were already creating vectors when we were using for-loops
  - ":" notation

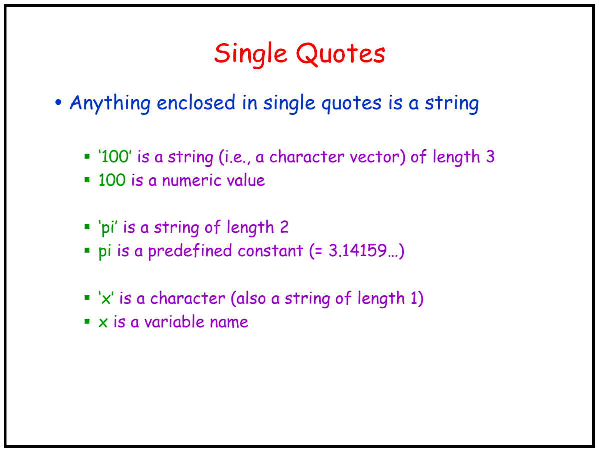
```
vec = 1:7;           % [1 2 3 4 5 6 7]
vec = 10: -2: 0      % [10 8 6 4 2 0]
```

- FYI
  - The for-loop actually converts the ":" notation into a vector before it executes
  - A for-loop will work with *any* vector!
    (e.g., `for k = [2 3 5 7 11 13 17 19]`)



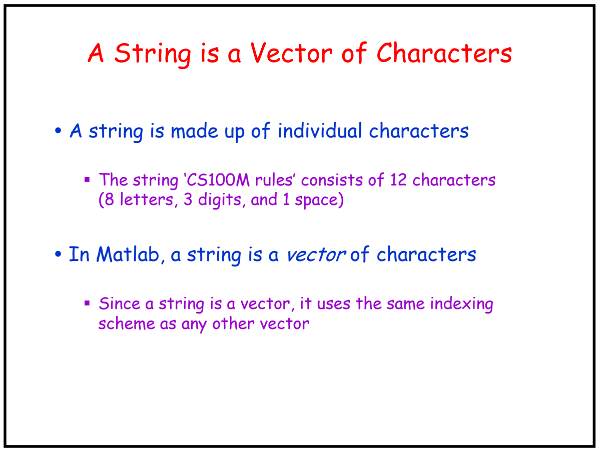
Matlab Strings

- You've been using strings
  - n = input('Next number: ');
  - fprintf('The answer is %d.', answer);
  - title('The Sine Function')
- 'Next number: ' and
  'The answer is %d.' and
  'The Sine Function' are all *strings*



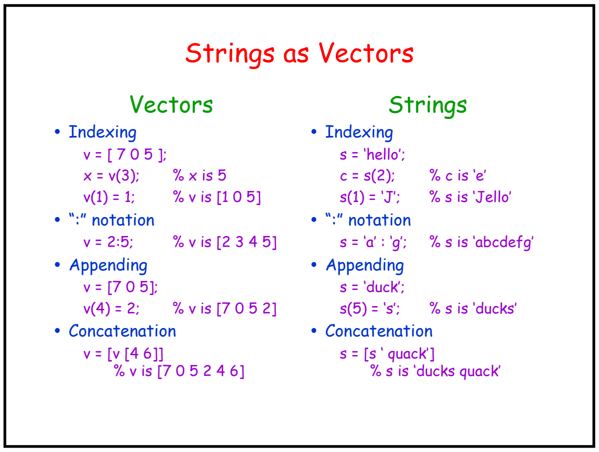
Single Quotes

- Anything enclosed in single quotes is a string
  - '100' is a string (i.e., a character vector) of length 3
  - 100 is a numeric value
  - 'pi' is a string of length 2
  - pi is a predefined constant (= 3.14159…)
  - 'x' is a character (also a string of length 1)
  - x is a variable name



A String is a Vector of Characters

- A string is made up of individual characters
  - The string 'CS100M rules' consists of 12 characters (8 letters, 3 digits, and 1 space)
- In Matlab, a string is a *vector* of characters
  - Since a string is a vector, it uses the same indexing scheme as any other vector



Strings as Vectors

Vectors

- Indexing

```
v = [ 7 0 5 ];
x = v(3);      % x is 5
v(1) = 1;      % v is [1 0 5]
```

- ":" notation

```
v = 2:5;       % v is [2 3 4 5]
```

- Appending

```
v = [7 0 5];
v(4) = 2;      % v is [7 0 5 2]
```

- Concatenation

```
v = [v [4 6]]
      % v is [7 0 5 2 4 6]
```

Strings

- Indexing

```
s = 'hello';
c = s(2);      % c is 'e'
s(1) = 'J';    % s is 'Jello'
```

- ":" notation

```
s = 'a' : 'g';     % s is 'abcdefg'
```

- Appending

```
s = 'duck';
s(5) = 's';    % s is 'ducks'
```

- Concatenation

```
s = [s ' quack']
      % s is 'ducks quack'
```

Some Useful String Functions

```
str = 'CS100M rules';

isletter(str)       % [ 1 1 0 0 0 1 0 1 1 1 1 1 ]
isspace(str)        % [ 0 0 0 0 0 0 1 0 0 0 0 0 ]

s = lower(str);     % s is 'cs100m rules'
s = upper(str);     % s is 'CS100M RULES'

ischar(str);        % Is str a char array?  1 (= true)
```

ASCII

(American Standard Code for Information Interchange)

ASCII Code	Character	ASCII Code	Character
48	'0'	97	'a'
49	'1'	98	'b'
50	'2'	99	'c'
51	'3'	…	…
…	…	122	'z'
65	'A'	…	…
66	'B'	127	DEL
67	'C'		
…	…		
90	'Z'		
…	…		

Characters ↔ ASCII Code

```
str = 'CS100M';       % Vector (1D array) of characters

code = double(str);   % Converts each character to a number;
                      % code is a standard Matlab vector

s = char(code);       % Converts a vector of numbers into
                      % a string (i.e., a vector of characters)
```

Character Arithmetic

- You can do "math" with characters

```
'd' – 'a'        % Produces 3
'9' – '8'        % Produces 1
'a' < 'd'        % Produces 1 (= true)
'd' < 'b'        % Produces 0 (= false)
'Z' < 'b'        % Produces 1 (= true)
                 % Because 90, the ASCII code for 'Z',
                 % is less than 98, the ASCII code for 'b'

'a' + 2          % Produces 99
char('a'+2)      % Produces 'c'
```

Example: toUpper

- Goal: Write toUpper( ), our own version of Matlab's upper( ), a function to convert a string to all uppercase
  - We want to do this without using Matlab's function upper( )

- Function header

```
function str = toUpper(str)
% Post: Convert string so all letters are upper case
% Pre: Input is a string
```

- Idea: Note that 'a' – 'A' has the same value as 'b' – 'B' which has the same value as 'c' – 'C', etc.
  - All we have to do is subtract the right number from a lowercase letter and we'll have the equivalent uppercase letter

Example: Capitalize First Letters

- Goal:
  - Write a function to capitalize just the first letter of each word in a string
  - Assume the string consists entirely of letters and spaces

- Function header

```
function result = capitalize(str)
% Post: Convert string so each word has just first letter capitalized
% Pre: Input string consists entirely of letters & spaces
```

Post = What is supposed to have happened when function is done (i.e., what the function does)

Pre = What assumptions are being made when function starts