1 Not string but chars

In MATLAB, there is the type char but not the type string. What we call a string is really an array of chars.

Type each of the following statements in the Command Window and note the result.

```matlab
a = pi; % A numeric scalar
b = 'pi' % A char array. Use SINGLE quotes to enclose a char or multiple chars

% __________ b is an array, so one can use function length on it

c = length(b)

d = ['apple' ' b 'es'] % Vector concatenation. d should be the string 'apple pies'

e = [d; 'muffin']
% ________________

% __________ Note the two extra 'm's and one trailing space

e = [d; 'mmmuffins ']

% __________ b is an array, so one can use function length on it
[nr,nc]= size(e)

% ________________

d = e(1, 7:9)
% ________________

f = e(1, 7:10) = 'core'
% ________________

% ________________ e is a matrix, so one can use function size on it

% ________________

g = ones(2,3)*67;
% A NUMERIC 2-by-3 matrix, each component has the value 67

h = char(g)
% ________________

i = double(h)
% ________________

% ________________

% ________________ A random upper case letter
jj = char(floor(rand(1)*26) + 'A')

k = jj>='a' && jj<='z' % __________ True or false: character stored in jj is lower case

L = jj - 'A'
% ________________

M = jj - 'A' + 1
% ______ A convenient way to use letters as indices from 1 to 26.
% If jj is 'A', then M is 1; if jj is 'B', then M is 2; ...

% ________________

n = strcmp('abcd', 'ab') % ________________ strcmp compares the arguments

% ________________

o = 'abcd'=='ab'
% ERROR: attempted vectorized code on vectors of different lengths

% ________________

% ________________ Vectorized code--result is a vector
p = 'abcd'=='abcd'

% ________________

q = sum('abcd'=='abCd')
% ________________ The number of matches

r = sum('abcd'~='abCd')
% ________________ The number of mismatches

% ________________ A random string of 2 upper case letters
s = char(floor(rand(1,2)*26) + 'A')

% ________________

t = s>'D' & s<X' % ____ True or false: both letters in s are strictly between 'D' and 'X'

% ________________

% ________________

% ________________

% ________________

% ________________ Vectorized filtering and assignment
v = 'acdc'

% ________________

v(v=='c') = 'X'
```

1
w = 'acdc';
w(w=='c')= '' % ________________________ Vectorized filtering and deletion

2 My upper function

Implement the following function as specified. The only built-in function that you should use is char. *Hint:* do arithmetic on characters! See the creation of variables jj and L in §1 above.

function up = toUpper(cha)
% up is the upper case letter corresponding to lower case letter cha.
% If cha is not a lower case letter, do not capitalize and up is simply cha.

3 Counting a DNA pattern

Write a function countPattern(dna,p) to find out (and return) how many times a pattern p occurs in dna. Assume both parameters to be strings that contain the letters 'A', 'T', 'C', and 'G' only. Note that if p is longer than dna, then p appears in dna zero times. Use a loop to solve this problem.

(a) Version 1: Use the built-in function strcmp to compare two strings.

(b) Version 2: Do not use strcmp; instead use vectorized code and sum as demonstrated in Part 1 above to compare two strings.

4 Censor

Implement the following function as specified. Only these built-in functions are allowed: length, size, ones, char, lower, upper, strcmp

function D = censor(str, A)
% Replace all occurrences of string str in char matrix A with X's, regardless of case.
% Assume str is never split across two lines.
% D is A with X's replacing the censored string str.
% Example: A is ['Use MATLAB '; ... then D is ['Use MATXX '; ...'
in that lab.']['in that XXX.
]