

CS1132 Lab Exercise 3

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.

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

c= length(b)           % ----- b is an array, so one can use function length on it

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

e= [d; 'muffin']       % -----

e= [d; 'mmmuffins ' ] % Note the two extra 'm's and one trailing space

[nr,nc]= size(e)       % ----- e is a matrix, so one can use function size on it

f= e(1, 7:9)           % ----- Accessing a subarray

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

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

h= char(g)             % -----

i= double(h)           % -----

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

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

L= strcmp('abcd', 'ab') % ----- strcmp compares the arguments

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

n= 'abcd'=='abCd'     % ----- Vectorized code--result is a vector

o= sum('abcd'=='abCd') % ----- The number of matches

p= sum('abcd'~= 'abCd') % ----- The number of mismatches
```

2 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.

3 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 MATXXX '; ...
%             'in that lab. ']             'in that XXX. ']
```

4 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 variable `jj` 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.
```