Strings and vectorized code

Given strings **`a='hello'`** and **`b='jello'`** …

- No. of characters that are the same in **`a`** and **`b`**: **`sum(a==b)`**
- No. of characters that are different in **`a`** and **`b`**: **`sum(a~=b)`**

Vectorized (logical) operations – 1d

```
a= [4 2 6 1 3];
b= [5 3 6 5 3];
Relational ops:
   L= a==b
   M= a>b
Arithmetic ops:
   c= a-b
Extraction:
   d= a(a==b)
   e= b(b>3 & rem(b,2)==1)
```

Vectorized (logical) operations – 1d

```
a= [4 2 6 1 3];
b= [5 3 6 5 3];
```

Relational ops:

```
L= a==b
M= a>b
```

L: [0 0 1 0 1]
M: [0 0 0 0 0]
} Type: Logical

Arithmetic ops:

```
c= a-b
```

[-1 -1 0 -4 0] Type: double

Extraction:

```
d= a(a==b)
```

[6 3]

```
e= b(b>3 & rem(b,2)==1)
```

[5 5]

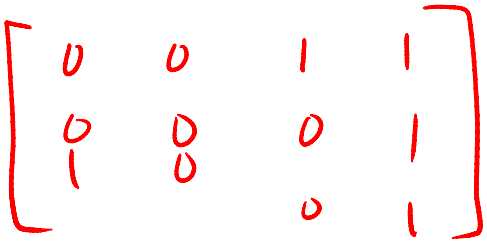
b>3: for vectorized code
rem(b,2)==1: odd

Vectorized (logical) operations – 2d

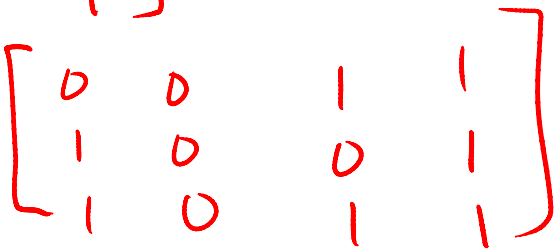
```
m= [ 2 3  5 7; …
    -2 1  0 7; …
     5 2 -1 8]

L= m>3

P= m>3 | m<0

a= m(m>4)

b= (m>4).*m
```

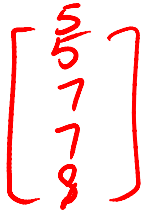
Vectorized (logical) operations – 2d

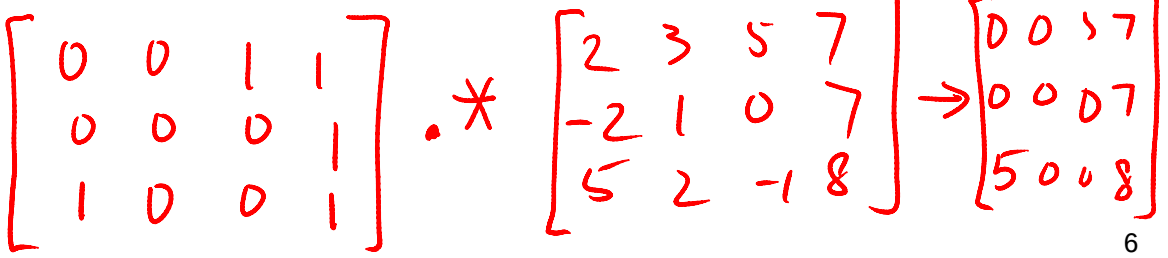
```
m= [ 2 3  5 7; …
    -2 1  0 7; …
     5 2 -1 8]

L= m>3

P= m>3 | m<0

a= m(m>4)

b= (m>4).*m
```

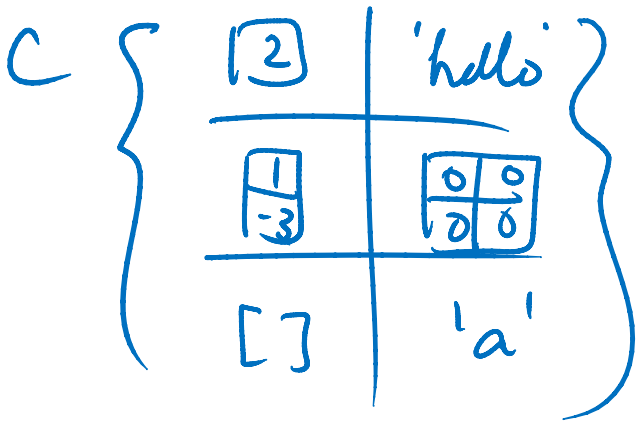


vectorized code








C
2
'hello'
1
-3
0
0
0
0
[]
'a'

ASCII characters

(American Standard Code for Information Interchange)

ascii code	*Character*	*ascii code*	*Character*
:	:	:	:
:	:	:	:
65	‘A’	48	‘0’
66	‘B’	49	‘1’
67	‘C’	50	‘2’
:	:	:	:
90	‘Z’	57	‘9’
:	:	:	:

Character vs ASCII code

```
str= 'Age 19'
    %a 1-d array of characters
code= double(str)
    %convert chars to ascii values
str1= char(code)
    %convert ascii values to chars
```

Arithmetic and relational ops on characters

- `'c'-'a'` gives 2
- `'6'-'5'` gives 1
- `letter1='e'; letter2='f';`
- `letter1-letter2` gives -1

- `'c'>'a'` gives true
- `letter1==letter2` gives false

- `'A' + 2` gives 67
- `char('A'+2)` gives 'C'

Example: toUpper

Write a function toUpper(cha) to convert character cha to upper case if cha is a lower case letter. Return the converted letter. If cha is not a lower case letter, simply return the character cha.

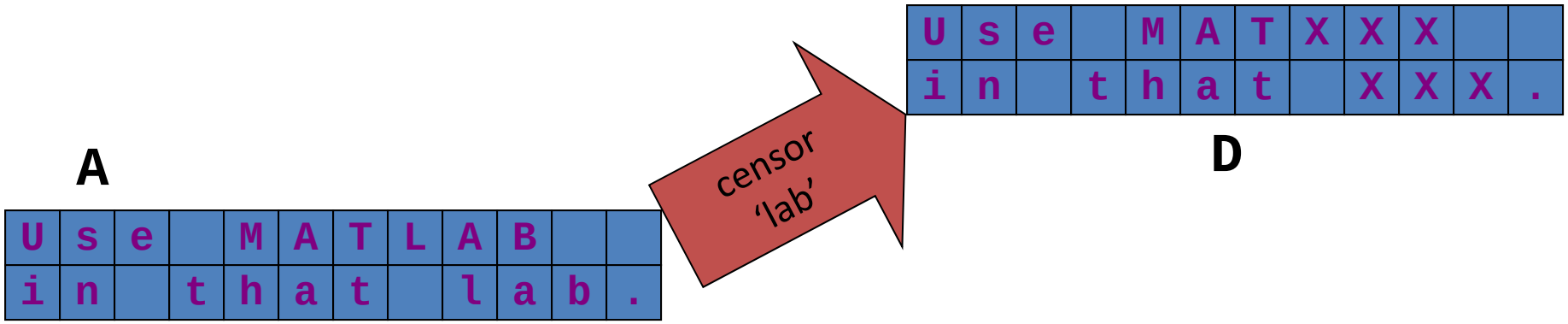
Hint: Think about the distance between a letter and the base letter 'a' (or 'A'). E.g.,

Of course, do not use Matlab function upper!

```
function up = toUpper(cha)
% up is the upper case of character cha.
% If cha is not a letter then up is just cha.
```

Example: censoring words

```
function D = censor(str, A)
% Replace all occurrences of string str in
% character matrix A with X's, regardless of
% case.
% Assume str is never split across two lines.
% D is A with X's replacing str.
```



```
function D = censor(str, A)
% Replace all occurrences of string str in character matrix A,
% regardless of case, with X's.
% A is a matrix of characters.
% str is a string.  Assume that str is never split across two lines.
% D is A with X's replacing the censored string str.

D= A;
B= lower(A);
s= lower(str);
ns= length(str);
[nr,nc]= size(A);

% Build a string of X's of the right length




% Traverse the matrix to censor string str
```