Mixed bags

Working with strings and cell arrays

CS112 Scientific Computation
Department of Computer Science
Wellesley College

Strings

Program input/output
month = input('Enter a month: ');
disp(['There are ' num2str(numDays(month)) ' days in that month']);
myTitle = get(handles.titleBox, 'String');
set(handle.sourceToggle, 'String', 'consume');

Text labels and graph properties
plot(xcoords, ycoords, 'g:*', 'Linewidth', 2);
title('golden ratio data');

Cell arrays of strings
names={'Varitek' 'Ortiz' 'Ramirez' 'Drew' ... 'Crisp' 'Pedroia'};

Reading/Writing image files
im = imread('albright.jpg');
imwrite(im, 'myImage.jpg');
What lies beneath - the ASCII code

ASCII code, established by the American Society for Communication and Information Interchange, is a numerical code for representing letters, digits, punctuation, and control signals.

original ASCII code represents characters using a 7-bit code (numbers from 0 to 127).

new international standard - Unicode - has thousands of characters, allows representation of multiple languages.

MATLAB character strings

Character strings in MATLAB are stored in special numerical vectors of ASCII values.

The `double` and `char` functions translate between a character string and its ASCII values:

```matlab
>> string1 = 'violet';
>> numcode = double(string1)
numcode =
    118   105   111  108  101  116
>> char(numcode)
ans =
    violet
```
String processing freebies

>> string1 = 'spring break';
ans =

>> length(string1)
ans =

>> string1(6)
ans =

>> string1(3:6)
ans =

>> string1{[6 3 11 8]}
ans =

>> string1(8:12) = 'fling'
ans =

>> string1(20) = '*';

Conditional expressions & strings

>> string1 == 'g'
ans =

0 0 0 0 0 1 0 0 0 0 1

ans

>> string1(string1 == 'f') = 'b'

string1 =

>> string1 == 'CS112'

* remember strcmp
More fun with strings

```matlab
>> string2 = 'to be or not to be';
>> find(string2 == ' ')
ans =

>> string2Letters = string2(string2 ~= ' ')
string2Letters =

>> string2(string2 == ' ') = []
string2 =
```

String processing newbies

```matlab
>> lower('To Be Or Not To Be') converts letters to lower case
ans =

>> upper(ans) converts letters to upper case
ans =

>> strrep('to be or not to be', 'be', 'play') replaces occurrences of 2nd input string with 3rd input string, in the 1st input string...
ans =

>> strfind(ans, 'ay') finds all occurrences of second input string in first input string... returns indices of first character
ans =
```
Time out exercises

What actions are performed by the following statements?

```matlab
newString = ' '; 
for letter = string1 
    newString = [letter newString];
end
```

What test is performed by the following function?

```matlab
function answer = test(str) 
str = str(str ~= ' '); 
str = lower(str); 
answer = all(str == str(end:-1:1));
``` 

>> answer = test(‘Murder for a jar of red rum’)

Collecting strings with cell arrays

We have used a cell array to store a collection of strings

```matlab
>> myPets = {'mona' 'cleo' 'trot'};
```

We can access the contents of individual locations of a cell array using an index placed inside curly braces:

```matlab
>> myPets{1} 
ans =
mona
```

```matlab
for index = 1:length(myPets) 
    disp(myPets{index});
end
```

Exercise: Write a function with a single input that is a cell array, which prompts the user for a string and determines whether the user’s string is contained in the input cell array
Collecting multiple types of data

The *real power* of cell arrays is that they allow us to store multiple types of data in one structure:

```matlab
>> myCell = {'Ellen' 3.14159 [2 5 1 7] [1 2; 3 4]}
myCell =
    'Ellen'  [3.14159]    [1x4 double]    [2x2 double]
```

Create a cell array from scratch with the `cell` function:

```matlab
>> newCell = cell(1,3);
>> newCell{1} = 'Sohie';
>> newCell{2} = 'SCI E127';
>> newCell{3} = sohieImage;
```

Accessing the contents of cell arrays

Contents of individual locations of a cell array can be accessed with an index surrounded by curly braces:

```matlab
>> myCell = {'Ellen' 3.14159 [2 5 1 7] [1 2; 3 4]};
>> disp([myCell{1}'s favorite number is ' num2str(myCell{2})])
ans =
>> myCell{3}(2)
ans =
>> sum(sum(myCell{4}))
ans =
```
Into thin air…

```matlab
mountains = {{'Everest' 'K2' 'Kanchenjunga' 'Lhotse I' 'Makalu I' ...
 'Lhotse II' 'Dhaulagiri' 'Manaslu I' 'Cho Oyu' ...
 'Nanga Parbat' 'Annapurna'}...
 {'Himalayas' 'Karakoram' 'Himalayas' 'Himalayas'
 'Himalayas' 'Himalayas' 'Himalayas' 'Himalayas' ...
 'Himalayas' 'Himalayas' 'Himalayas'}...
 {'Nepal-China' 'Kashmir' 'Nepal-India' 'Nepal-China' ...
 'Nepal-China' 'Nepal-China' 'Nepal' 'Nepal' 'Nepal-China' ...
 'Kashmir' 'Nepal'}...
 [29028 28250 28208 27923 27824 27560 26810 ...
 26760 26750 26660 26504]}.
mount = input('Enter the name of a mountain: ', 's');
```

Exercise:

Write a loop that prints all of the information about the user’s input mountain.