Programming Assignments

Introduction to Programming with MATLAB

Lesson 7

- Unless otherwise indicated, you may assume that each function will be given the correct number of inputs and that those inputs have the correct dimensions. For example, if the input is stated to be three row vectors of four elements each, your function is not required to determine whether the input consists of three two-dimensional arrays, each with one row and four columns.
- Unless otherwise indicated, your function should not print anything to the Command Window, but your function will not be counted incorrect if it does.
- Note that you are not required to use the suggested names of input variables and output variables, but you must use the specified function names.
- Finally, read the instructions on the web page on how to test your functions with the auto-grader program provided, and what to submit to Coursera to get credit.
- Note that starred problems, marked by ***, are harder than usual, so do not get discouraged if you have difficulty solving them.

1. Write a function called **`integerize`** that takes as its input a matrix **`A`** of non-negative integers of type **`double`**, and returns the name of the "smallest" unsigned integer class to which **`A`** can be accurately converted. If no such class exists, the string 'NONE' is returned. For example, if the largest element of A is 14, then the function would return **`'uint8'`**, but if the largest integer in **`A`** is 1e20, then the function would return 'NONE'.

2. Write a function called **`May2015`** that returns a struct vector (row or column) whose elements correspond to the days of May, 2015. Each struct should contain three fields with these (exact) field names: "month", "date", and "day" (all lower case).

   - The month field must contain the string **`'May'`** (uppercase 'M').
   - The date field must contain a scalar of type **`double`** that equals the date (1 through 31).
   - The day field must contain the three-letter abbreviation of the day chosen from this list: 'Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'.

   For example, here is a call of the function followed by a command that shows the eleventh element of the struct array that is returned by the function:

```
>> m = May2015;
>> m(11)
ans =
    month: 'May'
     date: 11
      day: 'Mon'
```

3. Write a function called **June2015** that returns a cell array of dimensions 30-by-3, whose rows correspond to the days of June, 2015. The three elements of each row must be set as follows:

   - The first element refers to the string **'June'** (uppercase 'J').
   - The second element refers to a scalar of type **double** that equals the date (1 through 30).
   - The third element refers to the three-letter abbreviation of the day chosen from this list: 'Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'.

   For example, here is a call of the function followed by a command that shows the eleventh element of the cell array that is returned by the function:

```
>> m = June2015;
>> m(11,:)
ans =
    'June'    [11]    'Thu'
```

4. Write a function called **codeit** that takes a string **txt** as input and returns another string **coded** as output. The function reverses the alphabet: It replaces each 'a' with 'z', each 'b' with 'y', each 'c' with 'x', etc. The function must work for uppercase letters the same way, but it must not change any other characters. Note that if you call the function twice like this **str = codeit(codeit(txt))** then **str** and **txt** will be identical.

5. Each number on older telephone keypads, except 0 and 1, corresponds to three uppercase letters as shown in this list:

   2 ABC, 3 DEF, 4 GHI, 5 JKL, 6 MNO, 7 PRS, 8 TUV, 9 WXY

   A phone-number specification can include uppercase letters other than Q and Z, digits, the # and * signs, spaces, parentheses and dashes. Write a function called **dial** that takes as its input argument a string of any length that includes only these characters and returns as its output argument a string containing the corresponding telephone number with only digits, spaces, and the # and * signs. Specifically, it does not change the digits, the # and * signs, or the spaces, but it replaces each parenthesis with a space and each dash with a space, and it converts each uppercase letter to a digit according to the list shown above. Here is the input and output for one example of a call of the function:

```
Input:    '1 (FUN) DOG-4-YOU #2'
Output:   '1  386  364 4 968 #2'
```

   Note that in this example there are three spaces in the input string and seven spaces in the output string. If the input does not meet the phone-number specification given above, the function returns an empty array. You are not allowed to use the built-in function **strrep**.

6. Write a function called **`replace`** that takes three inputs: a cell vector (row or column) of strings and two characters: **`c1`** and **`c2`**. The function returns the cell vector unchanged except that each instance of **`c1`** in each string is replaced by **`c2`**. You are not allowed to use the built-in function **`strrep`**.

7. Write a function called **`roman`** that takes a string representing an integer between 1 and 20 inclusive using Roman numerals and returns the Arabic equivalent as a number of type **`uint8`**. If the input is illegal or its value is larger than 20, **`roman`** returns 0 instead. The rules for Roman numerals can be found here: http://en.wikipedia.org/wiki/Roman_numerals. Use the definition at the beginning of the page under the "Reading Roman Numerals" heading. In order to have unambiguous one-to-one mapping from roman to Arabic numbers, consider only the shortest possible roman representation as legal. Therefore, only three consecutive symbols can be the same (**IIII** or **VIIII** are illegal, but **IV** and **IX** are fine). Also, a subtractive notation cannot be followed by an additive one using the same symbols making strange combinations, such as **IXI** for 10 or **IXX** for 19, illegal also.
8. Write a function called **`censor`** whose first input argument is a cell vector (row or column) of strings and whose second input argument is a string. It removes each element of the cell vector whose string is either identical to the second input argument or contains the second argument as a substring. The resulting cell vector is returned as the only output argument. For example, suppose each element of the first argument refers to one line of the first verse of the US national anthem and the second argument is **`'the'`**. Then the function would return a cell vector with a single element containing the string: **`'Oh, say does that star-spangled banner yet wave'`**.