Introduction to Programming 2017-2018

Exercise 4

Question 1

This question deals with the conversion of temperatures from Celsius (C) to Fahrenheit (F) and vice versa.

a. Write a function called '**convC2F**' that gets a temperature/s in Celsius and returns its value in Fahrenheit (see formulas at the end of the question).

b. Write a function called '**tempConv'** that gets two input arguments:

   - First input argument: the temperatures to be converted (an array of numbers).
   - Second input argument: Type of conversion, which can get one of two values: the number 1 for converting from Celsius to Fahrenheit, or the number 2 for converting from Fahrenheit to Celsius.

   The function should return the converted temperatures (output) according to the type of conversion.
   In this function, use your function from section a. You may add your own functions as you like.

Notes: All the functions in the question (sections a & b) should be able to get scalars, vectors and matrices as the input temperature/s. The output array should have the same size and shape as the input.

Formulas:

$$C = \frac{(F-32)\cdot 5}{9} \qquad F = \frac{C\cdot 9}{5} + 32$$

Examples:

*convC2*F([10 , 35]) → [50 , 95]

tempConv(110, 2) → 43.33

Question 2

a. Write a function called **vecMean** that calculates the cumulative average of elements in a vector. Your function should get one input argument: a vector of numbers (row or column), and return one output argument: a vector (the same size as the input vector). Each element in the output vector should be the average of this element and all previous elements in the input vector (see example below).

b. Write a function called **matMean** that calculates the cumulative average of the rows or the columns of a matrix. Your function should get two input argument:

- First input argument: a matrix of numbers.
- Second input argument: Type of cumulative average to calculate - can get one of two values: the number 1 for calculating the cumulative average of the rows of the matrix, or the number 2 for calculating the cumulative average of the columns of the matrix.

The function should return one output argument: a matrix (the same size as the input matrix). Each element in the output matrix should be the cumulative average of this element and all previous elements in the same row or column of the input matrix (see example below). In this function, use your function from section 'a'.

Notes: For this question you should write your own algorithm for calculating the cumulative averages of the arrays - **do not** use built-in statistical MATLAB functions such as: mean, sum, cumsum, etc.

Examples:

vecMean([1, 2, 3, 4]) → [1   1.5   2   2.5]

myMat = $\begin{bmatrix} 1 & 1 & 7 \\ 5 & 2 & 5 \\ 3 & 3 & 3 \end{bmatrix}$

matMean(myMat,1) → $\begin{bmatrix} 1 & 1 & 3 \\ 5 & 3.5 & 4 \\ 3 & 3 & 3 \end{bmatrix}$ *matMean(myMat,2)* → $\begin{bmatrix} 1 & 1 & 7 \\ 3 & 1.5 & 6 \\ 3 & 2 & 5 \end{bmatrix}$

Question 2

Write a function called 'myDiff' that calculates the difference between adjacent elements of an input vector and finds the maximal difference.

Your function should get one input argument – a vector (row or column).

Your function should return two output arguments:

- The first output argument should be a vector (one element shorter than the input vector) of differences between adjacent elements of the input vector:
  [X(2)-X(1) , X(3)-X(2) , ... X(end)-X(end-1)]
- The second output argument should be the element with the largest absolute value in the differences vector (maximal difference).

— You may add your own functions as you like and call them from the main function.

— In your code **do not** use the built-in Matlab functions: diff, min, max, sort.

— Tip: you can use the Matlab 'abs' function to calculate absolute values.

Example:

[diffVec,maxDiff] = myDiff([1, 8, 11, 3])

diffVec = [7, 3, -8]   maxDiff = -8

General notes:

1. Document your functions using comments in the code. You can add comment throughout the code as you see necessary, but the following comments are required:
   - At the beginning of each function, explain what the function does and describe the input and output arguments.
   - Add comments in the code before loops and before logical conditions, and when creating new variables.
2. Make sure to submit all code files you write (both the required functions and any additional functions you write).

3. The scripts (.m files) you write should be sent by e-mail to:ex.matlab@gmail.com
   Files should be sent no later than **Monday (4.12.17) / Wednesday (6.12.17)** (according to your tirgul group) - **at midnight**.
4. The hard copy of your exercise should include printed copies of your scripts. Please staple all the sheets together and clearly write your name and ID number on all of them.
   The hard copies should be submitted at the beginning of your Tirgul group.

Good Luck!