Islamic University of Technology (IUT)

Organization of Islamic Cooperation (OIC)
Department of Electrical and Electronic Engineering (EEE)

EEE 4416   Marks: 40
Lab Quiz (Matlab)   Time: 2.00 hrs

Question - 01

From left-to-right, if no digit is exceeded by the digit to its left, it is called an **increasing number**. For example - 2344568.

Similarly, if no digit is exceeded by the digit to its right, it is called a **decreasing** number. For example - 964221.

A bouncy number is **any non-negative integer that is neither increasing nor decreasing**. For example, 155349.

Given a number or an array, determine whether it's a bouncy number or not. The output should be logical True or False.

Test Case	Case - 01	Case - 02
Input	2989457	[3456, 321, 13245, 789465]
Output	1	[0, 0, 1, 1]

Question – 02

Given an array, write a function to find the summation the following way -

y = x (1) – x (2) - x (3) + x (4) - x (5) – x(6) + x(7) – x(8) – x(9) + x(10) - … … …

Test Case	Case - 01	Case - 02	Case - 03
Input	[2,5,4,6,1]	[10, 5, 5, 10, 5, 5, 10, 5, 5, 10]	[3,-3]
Output	-2	10	6

Question – 03

An array is given (even no. of inputs). Encode it in the following way –

Input: [3, 2, 1, 5, 4, 1]

Output: [2,2,2, 5, 1,1,1,1]

Explanation: Look at the input array elements pair-wise.

- 1st pair is (3, 2) == three, 2's.
- 2nd pair is (1, 5) == one, 5.
- 3rd pair is (4,1) == four, 1's.

If there are odd number of elements in the array, return the output – 'invalid'.

Test Case	Case - 01	Case - 02	Case - 03
Input	[2, 9]	[1, 2, 1, 3, 1, 4, 7,0]	[3,6,7]
Output	[9, 9]	[2,3,4,0,0,0,0,0,0,0]	'invalid'

Question – 04

Write a function that will take two string inputs. The output will be the number of times the 2nd string appeared in the 1st string. Ignore the case.

Test Case	Input - 1	Input -2	Output
Case - 01	'Knock, knock. Penny!!! … knock, knock. Penny!!! … knock, knock. Penny!!!'	'knock'	6
Case - 02	'The piano does not murder the player if it does not like the music'	'Not'	2
Case - 03	'the wild hunt'	'witcher'	0

Question – 05

Write a function to find the longest run of consecutive numbers in an array or a matrix.

- If your input is a matrix, your output by default should be a row vector operating column-wise. If the user gives a 2nd input as 2, the operation should be row-wise and the output should be a column vector. If the 2nd input is 1, then the default column-wise operation. Anything other than 1 and 2 should return an error message. If no 2nd input is given, perform default column-wise operation.
- You have to give the user the flexibility to provide 1 or multiple inputs.
- If there are more than one longest occurrences, provide only the first consecutive occurrence.

Test Case	Case - 01	Case - 02	Case - 03	Case - 04	Case - 05
Input - 1	[2, 9, 9, 0, 5, 9]	[1 1 1 2 2 2 3 3 3]	[1, 1, 4, 2; 1, 4, 4, 4; 5, 6, 4, 2; 5, 8, 4, 2]	[1, 1, 4, 2; 1, 4, 4, 4; 5, 6, 4, 2; 5, 8, 4, 2]	[1, 1, 4, 2; 1, 4, 4, 4; 5, 6, 4, 2; 5, 8, 4, 2]
Input -2				3	2
Output	9	1	[1, 1, 4, 2]	'invalid'	[1; 4; 5; 5]

Question – 06

Write a function to check if the parenthesis sequence is balanced in a given string.

Test Case	Input - 1	Output
Case - 01	'(()()())'	1
Case - 02	'(()()()()()()()()())'	1
Case - 03	'((()())'	0
Case - 04	')()('	0

Question - 07

Given an array of integers, sort the numbers based on their score. The score is the summation of all digits in that number.

For example, for n = 38, its score is = 3+8 =11.

If two numbers have the same score, place them in accordance with their original index in the given array. Hence, in test case – 02, 5 and 14 both have same weight but 5 appeared prior to 14 in the original array. So, in the output, it is placed before 14.

Test Case	Case - 01	Case - 02	Case - 03
Input	[99, 1000, 237, 6 , 72001]	[5, 7 ,9, 11, 14 ,29]	[3,-3, -33, -22]
Output	[1000, 6, 72001, 237, 99]	[11, 5, 14, 7, 9, 29]	[3, 3, 22, 33]

Question - 08

Given a matrix of size (m,n) - Find the final state of the matrix assuming the matrix as an abelian sandpile.

Think of the elements in the matrix as a pile of grains. By final state it means, all the elements in the matrix should be <4. Any elements greater than 4 should be evenly distributed around it i.e. distribute four sand grains, one each to the elements above, below, left and right. Continue doing this until all the piles(elements) in the matrix is less than 4.

Test Case	Case - 01	Case - 02	Case - 03
Input	[0 0 0 0; 0 0 4 0; 0 0 0 0]	[0 0 0 0; 0 4 4 0; 0 0 0 0]	[0, 0, 1, 0; 2, 0, 9, 0; 0, 3, 0, 0]
Output	[0 0 1 0; 0 1 0 1; 0 0 1 0]	[0 1 1 0; 1 1 1 1; 0 1 1 0]	[0, 0, 3, 0; 2, 2, 1, 2; 0, 3, 2, 0]