Islamic University of Technology (IUT)

Organization of Islamic Cooperation (OIC)

Department of Electrical and Electronic Engineering (EEE)

EEE 4416: Simulation Lab
Lab – 07 Assignment

Exercise - 01

While writing a doc file, you may have noticed that if you provide a space before a comma, it will issue a warning. Again, if you don't provide a space after a comma, it'll issue a warning. For example –

"Logan , you still have time."

Here, we placed a space before the comma. So, Microsoft Word is showing a warning. The proper way of writing would be –

"Logan, you still have time."

Now, say you want to write a program that will automatically solve this problem, i.e., it'll remove the space before a comma if the user mistakenly places any. That way, you won't have to worry about that anymore while writing a doc. Write a function named 'comma_check' that will take a sentence as input and provide the corrected sentence as output.

Test Case – 01:

- Input: "Logan , you still have time."
- Output: "Logan, you still have time."

Test Case – 02:

- Input: "Evil is evil. Lesser , greater , middling – makes no difference."
- Output: "Evil is evil. Lesser, greater, middling – makes no difference."

Test Case – 03:

- Input: "He is the white wolf, the king in the North."
- Output: "He is the white wolf, the king in the North."

- In this case, the input string doesn't have any issues. It's been written properly. So, no change.

Asif Newaz
Lecturer, Department of EEE, IUT

Exercise – 02

Problem statement:

Capitalize the 1st letter of each of the string. Other letters should be made lowercase.

Look at the test cases carefully.

Test Case – 01:

- Input: “pandas”
- Output: “Pandas”

Test Case – 02:

- Input: “Cats and Dogs”
- Output: “Cats and dogs”

***Test Case – 03:**

- Input: “763potus”
- Output: “763Potus”

****Test Case – 04:**

- Input: {'**GldosgGf', 'asbd', '$$$123', 'ASDR'}
- Output: {'**Gldosggf', 'Asbd', '$$$123', 'Asdr'}

Hint:

- Similar to ‘regexprep’, there is a ‘regexp’ function. It returns the index(position) of your desired match.
- You can use regexp find the locations of digits, whitespace, etc.

- Test case 4 is a bit tricky. See if you can solve it.

Exercise – 03

Problem statement: Iccanobif Sequence

This is quite similar to the Fibonacci sequence with a single difference.

$$Fib\ (i)\ =\ Fib\ (i-1)\ +\ fib\ (i-2)$$
$$Icc\ (i)\ =\ reverse\ (\ Icc\ (i-1)\ )\ +\ reverse\ (\ Icc\ (i-2)\ )$$

Sequence: Icc = [1, 1, 2, 3, 5, 8, 13, 39, 124, … …]

Here,

- icc(8) = 39 = rev(icc(7)) + rev(icc(6)) = 31 + 8 =39
- icc(9) = 124 = rev(icc(8)) + rev(icc(7)) = 93 + 31 =124

Your task is to write a function named 'Iccanobif' that will take an integer 'n' as input and provide the n-th number in the Iccanobif sequence.

If the input is anything other than an integer, e.g., a negative number or a string, print out an error message saying – "Input must be an integer".

- ❖ Plot the points in a figure and see if you can visualize the trend.
- ❖ Is vectorization possible for this problem?
- ❖ How to make your code memory efficient?

Exercise – 04*

Problem statement:

Refer back to exercise-01 in this week's lecture. You created a function called 'max_out' that takes a matrix and an integer 'n' as input, calculates the maximum of each n-by-n block in the matrix starting from (1, 1), and places those values in a separate matrix whose size is $\frac{1}{n}$ of its original size.

Let's try to refine the function even more.

Create a function named 'mat_out' that will take a matrix, an integer, and a string.

- The string should be 'max', 'min', or 'mean'. You should perform a similar operation on the matrix.
- Any other string other than above mentioned should return an error message.
- The integer(a) is the box size that you need to slice.

There is a catch. If your input matrix size is 10 and a=2, then the output matrix size would be 5x5. But if a=3, the remainder is no longer 0. In that case, the operation should be performed on the remaining slice. The output matrix size would be 4-by-4.

Test Case – 01:

- Input: mat= randi (400, 16), a=3, str = 'mean'

Test Case – 02:

- Input: mat= magic (20), a=5, str = 'min'

Test Case – 03:

- Input: mat= magic (10), a=5, str = 'mmn'
- Output: 'Unrecognized input arguments'

Test Case – 04:

- Input: mat= magic (20), a=3, str = 'min'

Test Case – 05:

- Input: mat= randi (400, 16, 23), a=3, str='max'

Hint: One way to solve this problem is to pad your matrix (you already know the function) to the desired size so that it is divisible by a.

Exercise - 05

Problem Statement:

Given a cell array of names, perform the following –

I. Extract the middle name.
II. Return only the 1st name.
III. Return 1st name and last name.
IV. Append the abbreviated form of the name in the end after a comma and a space.

Test Case – 01:

- Input: {“Harry James Potter”, “Eddard Ned Stark”, “ Charles Francis Xavier”}
- Output:
  I. {“James”, “Ned”, “Francis”}
  II. {“Harry”, “Eddard”, “Charles”}
  III. {“Harry Potter”, “Eddard Stark”, “Charles Xavier”}
  IV. {“Harry James Potter, HJP”, “Eddard Ned Stark, ENS”, “Charles Francis Xavier, CFX”}

⇨ Look out! The third input contains some leading spaces. This is a common scenario in text data processing where the input string contains leading and trailing spaces, usually due to recording errors.
⇨ You can easily remove them using the “strtrim” function or a regular expression (test both approaches).

```
⇨ strtrim(str)        % this will remove both leading and trailing spaces
```

- regexprep(str, '^\s+', '')

This will remove only leading spaces. Here,

- \s catches the space characters
- + indicates one or more space characters
- ^ indicates the start of the string

Exercise - 06

Problem Statement: Handling a 3D matrix

A 3D matrix is given as input. Perform the following tasks –

i. Extract all the elements from the 2nd column from the 3rd layer.
ii. Extract all the elements from the 3rd row from the 1st layer.
iii. Change the diagonal elements of the 2nd layer to 0.
iv. Change the 18th element of the matrix to nan.
v. Add another layer to the original matrix with all elements equal to 1000.

Test Case:

- Input:
  - a(:, :, 1) = magic(6)
  - a(:, :, 2) = randi(100,6)
  - a(:, :, 3) = spiral(6)
  - a(:, :, 4) = eye(6)

Exercise - 07

Problem statement:

Suppose you have a 3D array C of size M×N×P (e.g., an image stack or volumetric data). You need to compute the arithmetic mean of each 2D "page" (along the 3rd dimension) and return a 1×P vector slice_means.

Implement both of the following:

i. Loop + Pre-allocation
ii. Fully Vectorized (no explicit loops)

Finally, measure the execution time of each approach using tic/toc and compare.

Exercise – 08

Problem statement: Maximum Slice Sum in a 3D Array

You are given a 3D array A of size m × n × p. Your task is to:

- For each slice along the 3rd dimension (i.e., A(:, :, k)), compute the sum of all elements in that slice.
- Identify the slice index k_max that has the maximum sum.

Return:

i. The maximum sum,
ii. The corresponding slice index k_max,
iii. The slice itself as a 2D matrix. Return only the first one for duplicates

Write a function named 'slice_3d' that will take a 3D matrix as input and will provide 3 outputs.

Test Case – 01:

- Input: cat (3, [1 2; 3 4], [5 5; 5 5], [0 1; 0 1])
- Output:
  i. maxSum = 20
  ii. k_max = 2
  iii. slice =

```
[5  5
 5  5]
```

Test Case – 02:

- Input: cat(3, eye(4), spiral(4), magic(4), zeros(4))
- Output:
  iv. maxSum = 136
  v. k_max = [2, 3]
  vi. slice = spiral (4)

❖ *'cat'* stands for concatenation. It provides another way of creating a multi-dimensional matrix.

Exercise – 09*

Problem statement:

From the following passage, write a code to extract all the digits (individually as well as whole numbers).

Passage: 'On **June 28, 1914**, a quiet summer day in Sarajevo, Gavrilo Princip made a choice that echoed through history. By **July 28, 1914**, exactly **30 days** later, Austria-Hungary officially declared war on Serbia, marking the beginning of World War I. As autumn set in, soldiers counted the days: **100 days** of brutal trench warfare by **November 11, 1914**. The first winter dragged into **December 25, 1914**—a Christmas Day declared a temporary truce in pockets, where German and British troops shared bread and songs for just **one day**.

Throughout **1915**, battles raged on for **365 days** straight, leading into **1916's** infamous Somme offensive, which began on **July 1, 1916**, and lasted **141 days**, dragging into the chilling dawn of **November 18, 1916**. Then, on **March 8, 1917**, Russia began its upheaval, just **50 days** before the tsar abdicated on **April 27, 1917**.

World War I finally ended on **November 11, 1918**, exactly **1,572 days** after the assassination in Sarajevo. Only **21 years** later, Europe plunged back into conflict. On **September 1, 1939**, Germany invaded Poland, igniting World War II—**766 days** after the signing of the Munich Agreement in **September 1938**. By **May 10, 1940**, just **252 days** later, Germany had swept through the Low Countries.

In **1941**, on **June 22**, Operation Barbarossa was launched, beginning a brutal Eastern Front. The war in Europe raged for **1,594 days** until **May 8, 1945**—Victory in Europe Day. Meanwhile, in the Pacific, WWII spanned **2,194 days** from **December 7, 1941** (Pearl Harbor) until **September 2, 1945**, the official end on the USS Missouri.

Each day—365 days in 1942, 365 in 1943, another 366 in leap-year 1944, and the final 244 days of 1945—brought another statistic, another record: battles lasted 78 days, sieges endured for 872 days, victory celebrations lasted for weeks, and mourning for years.

Through those **5,768 days** from Sarajevo to Missouri, lives were measured in dates: **July 14, 1916**, **April 12, 1945**, **September 1, 1939**, and beyond. Each calendar turned—**365 days**, **366**, **365**, and **etc.**—etched into history. And through every year, every month counted in days, human resilience, hope, and loss were written in the language of dates.'

Output:

i. [2, 8, 1, 9, 1, 4, … …]
ii. [28, 1914, 28, 1914, 30, … …]

Hint:

- Check the 'regexp' documentation. You have seen in class how to match a pattern or position. Find out how to extract the matched pattern.

Asif Newaz
Lecturer, Department of EEE, IUT

Exercise – 10

Problem Statement: Write an anonymous function that removes all the spaces from a character array.

Test Case – 01:
- Input: 'H e l l o W o r l d'
- Output: 'HelloWorld'

Test Case – 02:
- Input: “Cats and Dogs”
- Output: “Catsanddogs”

Test Case – 03:
- Input: “what door?”
- Output: “whatdoor?”

Test Case – 04:
- Input: “Hodor”
- Output: “Hodor”

Exercise – 11

Problem Statement: Write an anonymous function that keeps only alphabetic letters (both lowercase and uppercase) and removes everything else (digits, punctuation, etc.) from a character array.

Test Case – 01:
- Input: ‘Abc123 def-456* *’
- Output: ‘Abcdef’

Test Case – 02:
- Input: ‘Cats and Dogs’
- Output: “CatsandDogs”

Test Case – 03:
- Input: “AI@2025: Transforming! The_Future...”
- Output: “AITransformingTheFuture”

Test Case – 04:
- Input: “Autobots! Roll out”
- Output: “AutobotsRollout”

- ❖ MATLAB's regexprep automatically handles both char and string inputs as long as s is a single string/character vector, not a cell or array of strings.