**FINAL LAB EXAM QUESTIONS**

**QUESTION NUMBER-1(ARRAYS)**

# **Program to find repeating element in an array (duplicate elements)**

****Algorithm****

1. Declare the array and input the array elements
2. Start traversing the array and check if the current element is already encountered
3. If it is already encountered, print the element as a repeating element and continue
4. If not, move to the next element of the array and continue

# **Program to remove duplicate elements in an array**

Program to remove duplicate elements in an array (sorted and unsorted array cases) is discussed here. Given an array, all the duplicate elements of the array are removed.

For example, consider the array.

## ****case 1:**** ****Remove duplicates from sorted array****

****Input:**** arr = {1, 2, 3, 4, 4}

****Output:**** arr = {1, 2, 3, 4}

## ****case 2:**** ****Remove duplicates from unsorted array****

****Input:**** arr = {9, 2, 7, 4, 7}

****Output:**** arr = {9, 2, 7, 4}

## ****Algorithm to remove duplicate elements in an array (sorted array)****

* Input the number of elements of the array.
* Input the array elements.
* Repeat from i = 1 to n
* - if (arr[i] != arr[i+1])
* - temp[j++] = arr[i]
* - temp[j++] = arr[n-1]
* Repeat from i = 1 to j
* - arr[i] = temp[i]
* return j.

**QUESTION NUMBER-2 (2-D ARRAYS)**

# Program to add two matrices

* (RECURSION)- TOWERS OF HANOI.

## Algorithm to add two matrices

* Input matrix 1 and matrix 2.
* If the number of rows and number of columns of matrix 1 and matrix 2 is equal,
* for i=1 to rows[matrix 1]
* for j=1 to columns [matrix 1]
* Input matrix 1 [i,j]
* Input matrix 2 [i,j]
* matrix 3 [i,j]= matrix 1 [i,j]+ matrix 2 [i,j];
* Display matrix 3 [i,j];

**QUESTION NUMBER -3(STACKS)**

* [Sorting a stack using a temporary stack](https://www.faceprep.in/sort-a-stack-using-temporary-stack-and-recursion/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)
* [Infix to postfix using a stack](https://www.faceprep.in/procoder/knowledgebase/infix-to-postfix-conversion-using stack/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)
* [Infix to prefix using a stack](https://www.faceprep.in/procoder/knowledgebase/infix-to-prefix-conversion-using-stack/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)

**QUESTION NUMBER-4( QUEUES)**

* [Queue using a linked list](https://www.faceprep.in/procoder/knowledgebase/queue-data-structure-insertions-and-deletion/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)
* [Circular queue using arrays and Linked list](https://www.faceprep.in/procoder/knowledgebase/circular-queue-using-arrays-and-linked-lists/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)
* [Implement stack using a queue](https://www.faceprep.in/procoder/knowledgebase/implementing-stacks-using-queues/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)
* [Implement queue using a stack](https://www.faceprep.in/procoder/knowledgebase/implementing-queues-using-stacks/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)

**QUESTION NUMBER-5 (LINKED LIST)**

* [Remove duplicates from a linked list](https://www.faceprep.in/remove-duplicates-from-a-linked-list/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)
* [Reverse a linked list](https://www.faceprep.in/procoder/knowledgebase/reversing-a-linked-list/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)
* [Circular queue using arrays and Linked list](https://www.faceprep.in/procoder/knowledgebase/circular-queue-using-arrays-and-linked-lists/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)

**QUESTION NUMBER - 6( TREES)**

* [Height of a binary tree](https://www.faceprep.in/find-the-height-of-a-binary-tree/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)
* [Find kth maximum value in a binary search tree](https://www.faceprep.in/program-to-find-the-kth-maximum-element-in-a-binary-search-tree/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)

**QUESTION NUMBER -7 (SEARCHING AND SORTING)**

* [Bubble Sort](https://www.faceprep.in/bubble-sort-in-c/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)
* [Selection Sort](https://www.faceprep.in/selection-sort-in-c/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)
* [Insertion Sort](https://www.faceprep.in/insertion-sort-in-c/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)
* [Merge Sort](https://www.faceprep.in/merge-sort-in-c/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)
* [Quick Sort](https://www.faceprep.in/quick-sort-algorithm-in-c/" \t "https://www.faceprep.in/data-structures/data-structures-programs/_blank)