# Indian Institute of Information Technology Allahabad

# ITP ARRAY Assignment-'1'

**NITYA GUPTA** IEC2021071 **VAIBHAV PANDEY** IEC2021072 **BHAVIKA LONGWANI** IEC2021073 **SHIVANI PAL** IEC2021074

***Abstract*:**

This paper contains an analysis concerning the task of writing a C program to count the total number of duplicate elements and delete all duplicate elements from an array.

## I. INTRODUCTION:

To count and delete absolute duplicate components in a given array, we want 'three for loops'. Run an external loop from 0 to measure, loop structure should look like for(i=0; i<size; i++). This loop is utilized to choose every component of the array and check the next resulting component for duplicate components utilizing one more settled loop. Now, inside the second for loop, add an 'if statement' to check repeating elements.Then to delete duplicate elements we use another nested 'for loop'.

This program requests that the client enter Array Size and array components. Then, it will count the all out number of duplicate components present in this exhibit utilizing For Loop.

We pronounced 1 One Dimensional Arrays arr[] of size 10 and furthermore proclaimed i to emphasize the Array components. The underneath printf explanation requests that the User enter the cluster arr[] size (Number of components an Array can hold). Also, a 'for loop' articulation will dole out the client entered qualities to the array.

Underneath For loop will assist with repeating every cell present in the arr[8] cluster. Condition (i<Size) in the for loop guarantees that the compiler does not surpass as far as possible. The C Programming scanf proclamation inside the for loop will store the client entered qualities in each individual exhibit component, for example:- arr[0], arr[1], arr[2], arr[3], arr[4].

## II. LOGIC:

## To check for the duplicates:

1.To include complete duplicate components in a given cluster we really want three loops. Run an external loop from 0 to estimate. Loop structure should look like for(i=0; i<size; i++).

2. This loop is utilized to choose every component of the cluster and check next

ensuing components for copies of components utilizing one more settled loop.
Run one more inward loop to see the first copy of the current exhibit component. Run an internal loop from i+1 such that the loop construction looks like for(j=i+1; j<size; j++). Presently, we run a loop from i+1. Since we really want to look for copy components in the next ensuing components, from the current component.
3. Inside inward loop ,check for copy component. In the event that a copy component is found, increase copy count. Which is if(arr[i] == arr[j]) then, at that point, count++.

## To Remove Duplicate Elements from an Array:

This segment will talk about the eliminating or cancellation of the copy components from an exhibit in the C programming language. At the point when a similar number of components happens in an arranged or unsorted cluster, then, at that point, the components of the exhibit are known as the copy components. Furthermore we want to erase these copy components or the same number from an exhibit to make the resultant cluster composed of interesting components.

For instance, there is a whole number sort exhibit arr[10] that contains { 6, 7, 3, 3, 7, 9} components. In this exhibit, 7 and 3 happen twice. Subsequently, these components are copy components. In this way, subsequent to erasing the copy components from a cluster arr[], we get 6, 7, 3,9 components.

# III. PSEUDOCODE:

Step 1: Input the size of an array from the user and store into the size variable.

Step 2: Use a for loop to read the elements of an array and store in the arr[i] variable.

Step 3: To get the duplicate elements from an array we need to use two for loops. Where the first loop starts from 0 to size. And the structure of the loop is: for (i = 0; i < size; i++).

Another loop selects each element of the array and compares it with the corresponding element to get the duplicate elements. And the structure of the inner loop is: for (j = i + 1; j < size; j++) and the code to find the subsequent same element is: if (arr[i] == arr[j]).

Step 4: If any duplicate element is encountered, delete the duplicate element from an array and the size of array is decremented by 1 such that, size = size - 1.

Step 5: After that, print the unique elements of an array , the code for algorithm is:

# IV. ACTUAL CODE:

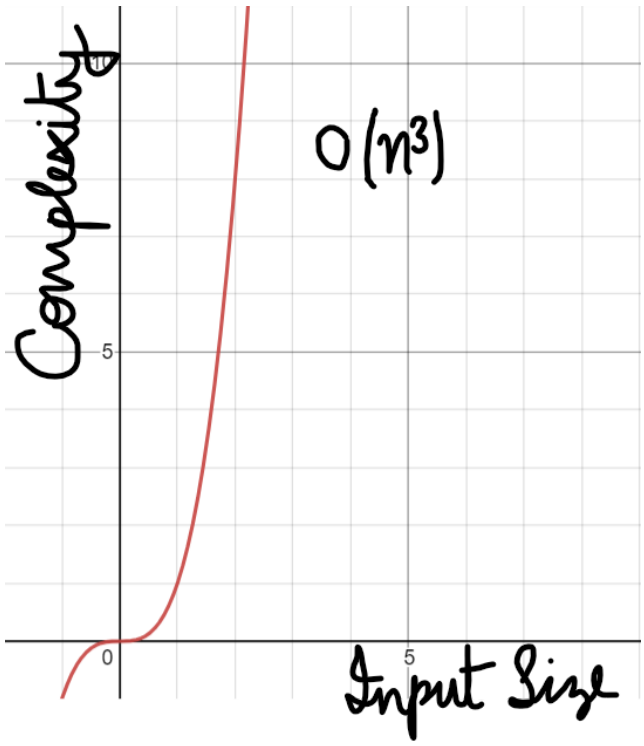
```c
#include <stdio.h>

int main()
{
    int i, j, k, SIZE, COUNT = 0;
    int arr[10];
    printf("Enter the SIZE of the array\n");
    scanf("%d", &SIZE);

    printf("Enter %d elements for the array\n", SIZE);
    for (i = 0; i < SIZE; i++)
    {
        scanf("%d", &arr[i]);
    }
    for (i = 0; i < SIZE; i++)
    {
        for (j = i + 1; j < SIZE; j++)
        {
            if (arr[i] == arr[j])
            {
                COUNT++;
                for (k = j; k < SIZE; k++)
                {
                    arr[k] = arr[k + 1];
                }
                SIZE--;
                j--;
            }
        }
    }

    printf("The number of duplicate elements are: %d\n", COUNT);

    printf("The total elements after deleting duplicate elements are: ");
    for (i = 0; i < SIZE; i++)
    {
        printf("%d\t", arr[i]);
    }

    return 0;
}
```

# V. OUTPUT:

```
Enter the SIZE of the array
8
Enter 8 elements for the array
1 2 3 2 2 3 3 4
The number of duplicate elements are: 4
The total elements after deleting duplicate elements are: 1     2       3       4
```

# VI. TIME COMPLEXITY:

**Time complexity** is the computational complexity that describes the amount of computer time it takes to run an algorithm.




The time complexity of this algorithm is $O(n^3)$.

# VII. CONCLUSION:

Here we have concluded the working of a C program for counting the duplicate elements of an array and deleting the duplicate elements of the same array.

# VII. REFERENCES:


C program to delete duplicate elements from array - Codeforwin

C Program to Delete Duplicate Elements from an Array (tutorialgateway.org)