GeeksforGeeks
A computer science portal for geeks

Practice IDE Q&A GeeksQuiz

Rearrange an array so that arr[i] becomes arr[arr[i]] with O(1) extra space

Given an array *arr[]* of size *n* where every element is in range from *0* to *n-1*. Rearrange the given array so that *arr[i]* becomes *arr[arr[i]]*. This should be done with *O(1)* extra space.

Examples:

```
Input: arr[]  = {3, 2, 0, 1}
Output: arr[] = {1, 0, 3, 2}

Input: arr[] = {4, 0, 2, 1, 3}
Output: arr[] = {3, 4, 2, 0, 1}

Input: arr[] = {0, 1, 2, 3}
Output: arr[] = {0, 1, 2, 3}
```

If the extra space condition is removed, the question becomes very easy. The main part of the question is to do it without extra space.

The credit for following solution goes to Ganesh Ram Sundaram . Following are the steps.

1) Increase every array element arr[i] by (arr[arr[i]] % n)*n.
2) Divide every element by n.

```
Let us understand the above steps by an example array {3, 2, 0, 1}
In first step, every value is incremented by (arr[arr[i]] % n)*n
After first step array becomes {7, 2, 12, 9}.
The important thing is, after the increment operation
of first step, every element holds both old values and new values.
Old value can be obtained by arr[i]%n and new value can be obtained
by arr[i]/n.

In second step, all elements are updated to new or output values
by doing arr[i] = arr[i]/n.
After second step, array becomes {1, 0, 3, 2}
```

Following is C++ implementation of the above approach.

```
#include <iostream>
using namespace std;

// The function to rearrange an array in-place so that arr[i]
// becomes arr[arr[i]].
void rearrange(int arr[], int n)
{
    // First step: Increase all values by (arr[arr[i]]%n)*n
    for (int i=0; i < n; i++)
        arr[i] += (arr[arr[i]]%n)*n;

    // Second Step: Divide all values by n
    for (int i=0; i<n; i++)
        arr[i] /= n;
}

// A utility function to print an array of size n
void printArr(int arr[], int n)
{
    for (int i = 0; i < n; i++)
        cout << arr[i] << " ";
    cout << endl;
}


/* Driver program to test above functions*/
int main()
{
    int arr[] = {3, 2, 0, 1};
    int n = sizeof(arr)/sizeof(arr[0]);

    cout << "Given array is \n";
    printArr(arr, n);

    rearrange(arr, n);

    cout << "Modified array is \n";
    printArr(arr, n);
    return 0;
}
```

Output:

```
Given array is
3 2 0 1
Modified array is
1 0 3 2
```

Time Complexity: O(n)

Auxiliary Space: O(1)

The only problem with above solution is, it may cause overflow.

This article is contributed by **Himanshu Gupta**. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

74 Comments Category: Arrays

Related Posts:

- Longest Span with same Sum in two Binary arrays
- Count Inversions of size three in a give array
- Find the subarray with least average
- Count triplets with sum smaller than a given value
- Find zeroes to be flipped so that number of consecutive 1's is maximized
- Reorder an array according to given indexes
- Find maximum value of Sum(i*arr[i]) with only rotations on given array allowed
- Find maximum average subarray of k length

(Login to Rate and Mark)

4 Average Difficulty : **4/5.0**
Based on **2** vote(s)

□ Add to TODO List

□ Mark as DONE

Writing code in comment? Please use code.geeksforgeeks.org, generate link and share the link here.