

## Reorder an array according to given indexes

Given two integer arrays of same size, "arr[]" and "index[]", reorder elements in "arr[]" according to given index array. It is not allowed to given array arr's length.

Example:

```
Input: arr[] = [10, 11, 12];  
       index[] = [1, 0, 2];  
Output: arr[] = [11, 10, 12]  
        index[] = [0, 1, 2]  
  
Input: arr[] = [50, 40, 70, 60, 90]  
       index[] = [3, 0, 4, 1, 2]  
Output: arr[] = [40, 60, 90, 50, 70]  
        index[] = [0, 1, 2, 3, 4]
```

Expected time complexity  $O(n)$  and auxiliary space  $O(1)$

**We strongly recommend you to minimize your browser and try this yourself first.**

A **Simple Solution** is to use an auxiliary array temp[] of same size as given arrays. Traverse the given array and put all elements at their correct place in temp[] using index[]. Finally copy temp[] to arr[] and set all values of index[i] as i.

```

// C++ program to sort an array according to given
// indexes
#include<iostream>
using namespace std;

// Function to reorder elements of arr[] according
// to index[]
void reorder(int arr[], int index[], int n)
{
    int temp[n];

    // arr[i] should be present at index[i] index
    for (int i=0; i<n; i++)
        temp[index[i]] = arr[i];

    // Copy temp[] to arr[]
    for (int i=0; i<n; i++)
    {
        arr[i] = temp[i];
        index[i] = i;
    }
}

// Driver program
int main()
{
    int arr[] = {50, 40, 70, 60, 90};
    int index[] = {3, 0, 4, 1, 2};
    int n = sizeof(arr)/sizeof(arr[0]);

    reorder(arr, index, n);

    cout << "Reordered array is: \n";
    for (int i=0; i<n; i++)
        cout << arr[i] << " ";

    cout << "\nModified Index array is: \n";
    for (int i=0; i<n; i++)
        cout << index[i] << " ";

    return 0;
}

```

Output:

```

Reordered array is:
40 60 90 50 70
Modified Index array is:
0 1 2 3 4

```

Thanks to [gccode](#) for suggesting above solution.

We can solve it **Without Auxiliary Array**. Below is algorithm.

- 1) Do following for every element arr[i]
  - a) While index[i] is not equal to i
    - (i) Store array and index values of the target (or correct) position where arr[i] should be placed. The correct position for arr[i] is index[i]
    - (ii) Place arr[i] at its correct position. Also update index value of correct position.
    - (iii) Copy old values of correct position (Stored in step (i)) to arr[i] and index[i] as the while loop continues for i.

Below is C++ implementation of above algorithm.

```

// A O(n) time and O(1) extra space C++ program to
// sort an array according to given indexes
#include<iostream>
using namespace std;

// Function to reorder elements of arr[] according
// to index[]
void reorder(int arr[], int index[], int n)
{
    // Fix all elements one by one
    for (int i=0; i<n; i++)
    {
        // While index[i] and arr[i] are not fixed
        while (index[i] != i)
        {
            // Store values of the target (or correct)
            // position before placing arr[i] there
            int oldTargetI = index[index[i]];
            char oldTargetE = arr[index[i]];

            // Place arr[i] at its target (or correct)
            // position. Also copy corrected index for
            // new position
            arr[index[i]] = arr[i];
            index[index[i]] = index[i];

            // Copy old target values to arr[i] and
            // index[i]
            index[i] = oldTargetI;
            arr[i] = oldTargetE;
        }
    }
}

// Driver program
int main()
{
    int arr[] = {50, 40, 70, 60, 90};
    int index[] = {3, 0, 4, 1, 2};
    int n = sizeof(arr)/sizeof(arr[0]);

    reorder(arr, index, n);

    cout << "Reordered array is: \n";
    for (int i=0; i<n; i++)
        cout << arr[i] << " ";

    cout << "\nModified Index array is: \n";
    for (int i=0; i<n; i++)
        cout << index[i] << " ";
    return 0;
}

```

Output:

```

Reordered array is:
40 60 90 50 70
Modified Index array is:
0 1 2 3 4

```