Find all triplets with zero sum

Given an array of distinct elements. The task is to find triplets in array whose sum is zero.

Examples:

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

Input : arr[] = {1, -2, 1, 0, 5}
Output : 1 -2  1
```

We strongly recommend that you click here and practice it, before moving on to the solution.

Source : Google Interview

Method 1 (Simple : O(n^3))

The naive approach is that run three loops and check one by one that sum of three elements is zero or not if sum of three elements is zero then print elements other wise print not found.

```
// A simple C++ program to find three elements
// whose sum is equal to zero
#include<bits/stdc++.h>
using namespace std;

// Prints all triplets in arr[] with 0 sum
void findTriplets(int arr[], int n)
{
    bool found = true;
    for (int i=0; i<n-2; i++)
    {
        for (int j=i+1; j<n-1; j++)
        {
            for (int k=j+1; k<n; k++)
            {
                if (arr[i]+arr[j]+arr[k] == 0)
                {
                    cout << arr[i] << " "
                         << arr[j] << " "
                         << arr[k] <<endl;
                    found = true;
                }
            }
```

```
        }
    }

    // If no triplet with 0 sum found in array
    if (found == false)
        cout << " not exist "<<endl;

}

// Driver code
int main()
{
    int arr[] = {0, -1, 2, -3, 1};
    int n = sizeof(arr)/sizeof(arr[0]);
    findTriplets(arr, n);
    return 0;
}
```

Run on IDE

```
0 -1 1
2 -3 1
```

Time Complexity : $O(n^3)$

Auxiliary Space : O(1)

Method 2 (Hashing : $O(n^2)$)

We iterate through every element. For every element arr[i], we find a pair with sum “-arr[i]”. This problem reduces to pairs sum and can be solved in O(n) time using hashing.

```
Run a loop from i=0 to n-2
  Create an empty hash table
  Run inner loop from j=i+1 to n-1
      If -(arr[i] + arr[j]) is present in hash table
         print arr[i], arr[j] and -(arr[i]+arr[j])
      Else
         Insert arr[j] in hash table.
```

```
// C++ program to find triplets in a given
// array whose sum is zero
#include<bits/stdc++.h>
using namespace std;

// function to print triplets with 0 sum
void findTriplets(int arr[], int n)
{
    bool found = false;

    for (int i=0; i<n-1; i++)
    {
        // Find all pairs with sum equals to
        // "-arr[i]"
        unordered_set<int> s;
        for (int j=i+1; j<n; j++)
        {
            int x = -(arr[i] + arr[j]);
            if (s.find(x) != s.end())
            {
                printf("%d %d %d\n", x, arr[i], arr[j]);
```

```
                found = true;
            }
            else
                s.insert(arr[j]);
        }
    }

    if (found == false)
        cout << " No Triplet Found" << endl;
}

// Driver code
int main()
{
    int arr[] = {0, -1, 2, -3, 1};
    int n = sizeof(arr)/sizeof(arr[0]);
    findTriplets(arr, n);
    return 0;
}
```

Run on IDE

```
-1 0 1
-3 2 1
```

Time Complexity : $O(n^2)$

Auxiliary Space : O(n)

Method 3 (Sorting : $O(n^2)$)

The above method requires extra space. We can solve in O(1) extra space. The idea is based on method 2 of this post.

```
1. Sort all element of array
2. Run loop from i=0 to n-2.
     Initialize two index variables l=i+1 and r=n-1
4. while (l < r)
     Check sum of arr[i], arr[l], arr[r] is
     zero or not if sum is zero then print the
     triplet and do l++ and r--.
5. If sum is less than zero then l++
6. If sum is greater than zero then r--
7. If not exist in array then print not found.
```

```
// C++ program to find triplets in a given
// array whose sum is zero
#include<bits/stdc++.h>
using namespace std;

// function to print triplets with 0 sum
void findTriplets(int arr[], int n)
{
    bool found = false;

    // sort array elements
    sort(arr, arr+n);

    for (int i=0; i<n-2; i++)
    {
        // initialize left and right
        int l = i + 1;
```

```
        int r = n - 1;
        int x = arr[i];
        while (l < r)
        {
            if (x + arr[l] + arr[r] == 0)
            {
                // print elements if it's sum is zero
                printf("%d %d %d\n", x, arr[l], arr[r]);
                l++;
                r--;
                found = true;
            }

            // If sum of three elements is less
            // than zero then increment in left
            else if (x + arr[l] + arr[r] < 0)
                l++;

            // if sum is greater than zero than
            // decrement in right side
            else
                r--;
        }
    }

    if (found == false)
        cout << " No Triplet Found" << endl;
}

// Driven source
int main()
{
    int arr[] = {0, -1, 2, -3, 1};
    int n = sizeof(arr)/sizeof(arr[0]);
    findTriplets(arr, n);
    return 0;
}
```

Run on IDE

Output:

```
-3 1 2
-1 0 1
```

Time Complexity : $O(n^2)$

Auxiliary Space : O(1)

This article is contributed by **DANISH_RAZA**. If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

GATE CS Corner Company Wise Coding Practice

Arrays Hash Searching Sorting

Recommended Posts:

Maximum consecutive 1's in a binary array
Find a triplet that sum to a given value
Maximum sum of smallest and second smallest in an array
Constant time range add operation on an array
Check if reversing a sub array make the array sorted

Logged in as **Dragan Nikolic** (Logout)

3.5 Average Difficulty : **3.5/5.0**
Based on **6** vote(s)

Basic Easy Medium Hard Expert

Add to TODO List
Mark as DONE

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

Share this post!

Brij Raj Kishore • 2 days ago

In second method please correct the algorithm to
Run a loop from i=0 to n-1
Create an empty hash tab;e

To
Run a loop from i=0 to n-2
Create an empty hash table

GeeksforGeeks Mod ➚ Brij Raj Kishore • 2 days ago

Thanks for pointing this out. We have updated the same.