Find the Missing Number

You are given a list of n-1 integers and these integers are in the range of 1 to n. There are no duplicates in list. One of the integers is missing in the list. Write an efficient code to find the missing integer.

```
Example:
I/P    [1, 2, 4, ,6, 3, 7, 8]
O/P    5
```


METHOD 1(Use sum formula)
Algorithm:

```
1. Get the sum of numbers
       total = n*(n+1)/2
2  Subtract all the numbers from sum and
   you will get the missing number.
```

Program:

```
#include<stdio.h>

/* getMissingNo takes array and size of array as arguments*/
int getMissingNo (int a[], int n)
{
    int i, total;
    total  = (n+1)*(n+2)/2;
    for ( i = 0; i< n; i++)
       total -= a[i];
```

```
    return total;
}

/*program to test above function */
int main()
{
    int a[] = {1,2,4,5,6};
    int miss = getMissingNo(a,5);
    printf("%d", miss);
    getchar();
}
```

Time Complexity: O(n)

METHOD 2(Use XOR)

```
1) XOR all the array elements, let the result of XOR be X1.
2) XOR all numbers from 1 to n, let XOR be X2.
3) XOR of X1 and X2 gives the missing number.
```

```
#include<stdio.h>

/* getMissingNo takes array and size of array as arguments*/
int getMissingNo(int a[], int n)
{
    int i;
    int x1 = a[0]; /* For xor of all the elemets in arary */
    int x2 = 1; /* For xor of all the elemets from 1 to n+1 */

    for (i = 1; i< n; i++)
        x1 = x1^a[i];

    for ( i = 2; i <= n+1; i++)
        x2 = x2^i;

    return (x1^x2);
}

/*program to test above function */
int main()
{
    int a[] = {1, 2, 4, 5, 6};
    int miss = getMissingNo(a, 5);
    printf("%d", miss);
```

Geometric Algorithms

Popular Posts

```
    getchar();
}
```

Time Complexity: O(n)

In method 1, if the sum of the numbers goes beyond maximum allowed integer, then there can be integer overflow and we may not get correct answer. Method 2 has no such problems.

Related Tpoics:

- Remove minimum elements from either side such that 2*min becomes more than max
- Divide and Conquer | Set 6 (Search in a Row-wise and Column-wise Sorted 2D Array)
- Bucket Sort
- Kth smallest element in a row-wise and column-wise sorted 2D array | Set 1
- Find the number of zeroes
- Find if there is a subarray with 0 sum
- Divide and Conquer | Set 5 (Strassen's Matrix Multiplication)
- Count all possible groups of size 2 or 3 that have sum as multiple of 3

Writing code in comment? Please use **ideone.com** and share the link here.

705 Subscribe

53 Comments GeeksforGeeks

Sort by Newest

Join the discussion…

Anubhav • 11 days ago
is it possible to do it using a hashmap..???
Will the worst case time be worse than what it is with the 2nd method or will it
• Reply • Share ›

Sourabh Upadhyay • a month ago
This can also be solved in O(n) by marking arr[arr[i]] = -arr[arr[i]] and finding th
found then return n.
• Reply • Share ›

mb1994 • a month ago
overflow can be avoided in the first algorithm.
instead of total=n*(n+1)/2:
add total=total+i+1 in the loop already used.
• Reply • Share ›

wrestler mb1994 • 6 days ago
Finally you are also storing total, so u r also not avoiding it
• Reply • Share ›

Recent Comments

Aman Hi, Why arent we checking for conditions...
Write a C program to Delete a Tree. · 18 minutes ago

kzs please provide solution for the problem...
Backtracking | Set 2 (Rat in a Maze) · 22 minutes ago

Sanjay Agarwal bool tree::Root_to_leaf_path_given_sum(tree...
Root to leaf path sum equal to a given number · 47 minutes ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...
Count trailing zeroes in factorial of a number · 48 minutes ago

newCoder3006 If the array contains negative numbers also. We...
Find subarray with given sum · 1 hour ago

newCoder3006 Code without using while loop. We can do it...
Find subarray with given sum · 1 hour ago

gandhi_rahul • 2 months ago

What if two numbers are missing? and can we generalize it for 'k' elements?

^ | v • Reply • Share ›

Meenal • 4 months ago

Why do we need n+1 in second loop, and not n i.e

for (i = 2; i <= n+1; i++)
x2 = x2^i;

Can anyone explain?

^ | v • Reply • Share ›

Sumit Khanna • 4 months ago

Another solution could be taking index=Arr[i] for i =0->n-1 ... and negating the v
with above procedure for all elements,,run a loop to check which element Arr[i

3 ^ | v • Reply • Share ›

Amit Baghel • 6 months ago

Visited :D

^ | v • Reply • Share ›

Kedar • 6 months ago

class example

{

public example()

{

int a[]={1,4,3,2,6};

int i,j,temp;

for(i=0;i<a.length;i++) for(j="0;j<a.length;j++)" if(a[i]<a[j])="" {="" temp="a[i];"
k="0;k<a.length;k++)" {="" if(a[k]!="k+1)" {="" system.out.print("the="" missin
break;="" }="" }="" }="" }="" class="" program="" {="" public="" static="" void="
example="" ex="new" example();="" }="" }="">

 • Reply • Share ›

vishal11 • 7 months ago

how can we find the missing nos if they are more than 1
Ex-{1,2,3,5,7}
output-4,6

4   • Reply • Share ›

guest ➚ vishal11 • 3 months ago

Will this help ?

```
void find_occ(int a[], int n)
{
int i = 0, t = 0;
while(i < n)
{
if((a[i] - 1 == i) || (a[i] > n)) {
i++;
}
else {
t = a[a[i] - 1];
a[a[i] - 1] = a[i];
a[i] = t;
}
}
```

```
for(i = 0; i < n; i++)
if(a[i] > i + 1)
cout << i + 1 << " missing" << endl;

}
```

Reply • Share ›

sandhanapandianrr • 9 months ago

```
#include<stdio.h>
#include<stdlib.h>

int main()
{

int a[] = {1, 2, 4, 5,6};
int i;
for(i=0;i<4;i++)
{
if(a[i+1]-a[i]!=1)
{
printf("%d", a[i]+1);
}
}
return 0;
}
```

5 Reply • Share ›

NK sandhanapandianrr • 4 months ago

Only if the input is sorted, XOR approach works on un-ordered array tc

Reply • Share ›

Kaustav Chatterjee • 10 months ago

```
#include
void quickSort(int[],int,int);
int partition(int[],int,int);
void exchange(int*,int*);
int GetMissingNumber(int[],int);
int main()
{
int i,n,b,a[100];
printf("Enter the no of elements of the array\n");
scanf("%d",&n);
printf("Enter the array\n");
for(i = 0;i<n;i++)
{
scanf("%d",&a[i]);
}
quickSort(a,0,n-1);
```

see more

• Reply • Share ›

Abhay • 10 months ago

Hashing can also be applied. Let array length be X. For each a[i], increase a[(a
C. Traverse the array again , to find the number which is less than C (Take C
which is same as n taken in queston), and return it's index. If no such quantity
returned

```
class Solver1
{
    public int SolverUtil(int a[])
    {
        int N=a.length+1;
```

```
for(int i=0;i<a.length;i++)
{
    a[(a[i]-1)%(a.length)]+=N;
}
int M=-1;
for(int j=0;j<a.length;j++)
{
    if(a[j]<N)
```

see more

• Reply • Share ›

Abhay ➚ Abhay • 10 months ago

The line "Take C larger than a.length, I have taken N, which is same as should be chosen only a.length.

Corrected code is:

```
class Solver1
{
    public int SolverUtil(int a[])
    {
        int N=a.length+1;
        for(int i=0;i<a.length;i++)
        {
            a[(a[i]-1)%(a.length)]+=a.length;
        }
        for(int j=0;j<a.length;j++)
        {

            if(a[j]<a.length)
                return j+1;
```

Reply • Share ›

Nagaraju • a year ago

Scan from left to right and make value at index a[i] to negative. Next pass is to non negative number and return index of the number. If we did not find any suc

```
[sourcecode language="JAVA"]
public static double getMissingValueMethod2(int[]a, int n){
for(int i =0; i < n-1; i++)
{
int t = Math.abs(a[i]);
if(t < n-1)
{
a[t] = -a[t];
}
}

for(int k =1; k < n-1; k++)
{
if(a[k] > 0) return k;

}
return n;
}
```

Reply • Share ›

Nagaraju • a year ago

```
public static double getMissingValueMethod2(int[]a, int n){
for(int i =0; i < n-1; i++)
{
int t = Math.abs(a[i]);
```

```
if(t < n-1)
{
a[t] = -a[t];
}
}

for(int k =1; k 0) return k;

}
return n;
}
```

Reply • Share ›

Chinmaya • a year ago

As the range of numbers is given,create a hash table. For every element in the
in the hash table to 1. Now look for 0 in the hash table and thus its index as the

Its complexity will be O(n).

Reply • Share ›

Shivam Maharshi • a year ago

Can we also do this with Binary Search? Time complexity will only be O(log(n)

Reply • Share ›

anonymous ➚ Shivam Maharshi • 5 months ago

Even if it WAS sorted. What would you be searching for?

Reply • Share ›

GeeksforGeeks ➚ Shivam Maharshi • a year ago

Binary Search can not be applied as the given array is not sorted.

Reply • Share ›

```
Yeah, didn't notice that.
```


Aman • 2 years ago

I am not getting how the logic of xor is working. Whats the mathematical or an
please help...

```
/* Paste your code here (You may delete these lines if not writing c
```


rohith ↗ Aman • 10 months ago

Same problem.... If you have found the answer for the logic behind usi
logic.
Thank you.

rohith ↗ rohith • 10 months ago

Oh u have replied :-P ..
Thank you..

kapil kumar chawala ↗ rohith • 9 months ago

Thanks, for the XOR logic, I got it...

Aman ↗ Aman • 2 years ago

i got it..
take xor to be (.) operator , its logic is..
if we take two sequences (a,b,c,d,e)& (a,b,c,d)

=(a.a).(b.b).(c.c).(d.d).e
=0.0.0.0.e
=0.e
=e

2 ^ | v • Reply • Share ›

Arpit • 2 years ago

```
public static void findMissing(Integer[] arr){
        int i=0, x1=arr[0],x2=1^2;
        for (i=1;i<arr.length;i++){
                x1 = x1 ^ arr[i];

                x2 = x2^(i+2);
        }


        System.out.println("Missing (No Duplicates)- "+ (x1^x2

}


public static void main(String[] args) {
        // TODO Auto-generated method stub
        try{
```

see more

^ | v • Reply • Share ›

cxin ➦ Arpit • 7 months ago

i like your code. the original post return wrong answer when missing 1

pr6989 • 2 years ago

```
 #include<stdio.h>
#include<stdlib.h>
using namespace std;
int main()
{
    int a[]={1,2,4,6,3,7,8};
    int size=sizeof(a)/sizeof(a[0]);
    int *arr=(int*)malloc(sizeof(int)*(size+1));
    int i;

    for(i=0;i<(size+1);i++)
    arr[i]=0;

    for(i=0;i<size;i++)
    arr[a[i]-1]++;

    for(i=0;i<(size+1);i++)
    {
```

see more

^ | v • Reply • Share ›

pr6989 ➚ pr6989 • 2 years ago

The above method should be O(n)

1.I'm creating a dynamic array "arr" of size= 1+size of given array and
2. Scan the given array and increment the count of the element in array
3. Scan the array "arr".Whichever index has value=0 that index+1 gives

^ | v • Reply • Share ›

Yogesh Batra pr6989 • 2 years ago

But you are using extra space O(n), it'll be an overhead. Why s solve the same thing without using extra space?

```
/* Paste your code here (You may delete these lines if
```

• Reply • Share ›

pr6989 • 2 years ago

How about this?

```
 #include<stdio.h>
#include<stdlib.h>
using namespace std;
int main()
{
    int a[]={1,2,4,6,3,7,8};
    int size=sizeof(a)/sizeof(a[0]);
    int *count=(int*)malloc(sizeof(int)*(size+1));
    int i;

    for(i=0;i<(size+1);i++)
    count[i]=0;

    for(i=0;i<size;i++)
    count[a[i]-1]++;
```

see more

• Reply • Share ›

Shyam • 2 years ago

@geeksforgeeks

these methods do not work if the missing element in the list is n... (i.e) if numb
missing then incorrect answer of 0 will be obtained unless we give the upper li

 • Reply • Share ›

kartik Shyam • 2 years ago

@shyam: Take a closer look at the programs. They all take size of arra
the size is n, then sum of n+1 elements is considered. In second meth
to make sure that the last missing case is handled. You could try runni
inputs.

 • Reply • Share ›

Meenal kartik • 4 months ago

Why do we need n+1 in second case. Can you please explain l

 • Reply • Share ›

neilljohnson • 3 years ago

```
>>> a=range(1,101,1)
>>> a
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24
35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55,
66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86,
97, 98, 99, 100]
>>> import random
>>> random.shuffle(a)
>>> a
[24, 3, 2, 60, 90, 95, 8, 40, 49, 53, 58, 97, 66, 7, 82, 6, 81, 22, 72, 12, 91, 76, 1
15, 26, 86, 67, 98, 99, 65, 79, 94, 30, 80, 17, 44, 43, 29, 33, 73, 35, 61, 19, 32,
28, 71, 70, 89, 39, 42, 9, 63, 52, 45, 56, 96, 34, 64, 25, 31, 74, 13, 10, 50, 37, 1
41, 75, 51, 46, 100]
>>> a.pop()
```

100
>>>
>>> a
[24, 3, 2, 60, 90, 95, 8, 40, 49, 53, 58, 97, 66, 7, 82, 6, 81, 22, 72, 12, 91, 76, 1

see more

Reply • Share ›

wgpshashank • 3 years ago

we are doing xor two time because its becomes duplicted aray & only one elei
xor=x1^x2 will give us..missing number ..isn't it

Reply • Share ›

Venki • 4 years ago

In method 2, why do we need two loops? We need one extra XOR operation w

Further we can optimize the XOR technique. Assuming N as power of 2, then
N = N. It can be done in O(1) time.

Reply • Share ›

ravikant • 4 years ago

can somebody please how the XOR logic works ????

Reply • Share ›

Meenakshi ravikant • 3 years ago

refer the below link

http://www.rawkam.com/?p=48

Reply • Share ›

Suesh PV • 4 years ago

Given an array of n-2 elements, how do you find the two missing elements.

kartik ➚ Suesh PV • 4 years ago

The trick used @http://geeksforgeeks.org/?p=2457 can also be used h
XOR the result with all numbers from 1 to n.

^ | v • Reply • Share ›

Jithan • 4 years ago

Solution can be calculated in O(logn). If the a[n/2] = n/2, then the missing num
Repeat the process to find the number. Similar to binary search.

^ | v • Reply • Share ›

kartik ➚ Jithan • 4 years ago

@Jitin: The solution that your are suggesting works for sorted array on

In the above given solutions, array is not assumed to be sorted.

^ | v • Reply • Share ›

Jithan ➚ kartik • 4 years ago

My bad! :)

^ | v • Reply • Share ›

GeeksforGeeks • 4 years ago

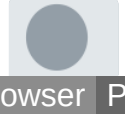
TJ & rv_10987: We have added a new method (please see method 2) that doe
problem.

^ | v • Reply • Share ›

TJ • 4 years ago

@rv can you please explain your solution?

^ | v • Reply • Share ›



rv_10987 • 4 years ago

a[a[i]]*=-1;
Now traverse the array,print the index value of the only positive element..!!

• Reply • Share ›

Load more comments

Subscribe Add Disqus to your site