Find the two repeating elements in a given array

You are given an array of n+2 elements. All elements of the array are in range 1 to n. And all elements occur once except two numbers which occur twice. Find the two repeating numbers.

For example, array = {4, 2, 4, 5, 2, 3, 1} and n = 5

The above array has n + 2 = 7 elements with all elements occurring once except 2 and 4 which occur twice. So the output should be 4 2.

Method 1 (Basic)
Use two loops. In the outer loop, pick elements one by one and count the number of occurrences of the picked element in the inner loop.

This method doesn't use the other useful data provided in questions like range of numbers is between 1 to n and there are only two repeating elements.

```
#include<stdio.h>
#include<stdlib.h>
void printRepeating(int arr[], int size)
{
  int i, j;
  printf(" Repeating elements are ");
  for(i = 0; i < size; i++)
    for(j = i+1; j < size; j++)
      if(arr[i] == arr[j])
        printf(" %d ", arr[i]);
}

int main()
{
  int arr[] = {4, 2, 4, 5, 2, 3, 1};
  int arr_size = sizeof(arr)/sizeof(arr[0]);
```

Google™ Custom Search

GeeksforGeeks

53,522 people like GeeksforGeeks.

Interview Experiences
Advanced Data Structures
Dynamic Programming
Greedy Algorithms
Backtracking
Pattern Searching
Divide & Conquer
Mathematical Algorithms
Recursion
Geometric Algorithms

```
  printRepeating(arr, arr_size);
  getchar();
  return 0;
}
```

Time Complexity: O(n*n)
Auxiliary Space: O(1)

Method 2 (Use Count array)
Traverse the array once. While traversing, keep track of count of all elements in the array using a temp array count[] of size n, when you see an element whose count is already set, print it as duplicate.

This method uses the range given in the question to restrict the size of count[], but doesn't use the data that there are only two repeating elements.

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

void printRepeating(int arr[], int size)
{
  int *count = (int *)calloc(sizeof(int), (size - 2));
  int i;

  printf(" Repeating elements are ");
  for(i = 0; i < size; i++)
  {
    if(count[arr[i]] == 1)
      printf(" %d ", arr[i]);
    else
     count[arr[i]]++;
  }
}

int main()
{
  int arr[] = {4, 2, 4, 5, 2, 3, 1};
  int arr_size = sizeof(arr)/sizeof(arr[0]);
  printRepeating(arr, arr_size);
  getchar();
  return 0;
}
```

Time Complexity: O(n)

Popular Posts

- All permutations of a given string
- Memory Layout of C Programs
- Understanding “extern” keyword in C
- Median of two sorted arrays
- Tree traversal without recursion and without stack!
- Structure Member Alignment, Padding and Data Packing
- Intersection point of two Linked Lists
- Lowest Common Ancestor in a BST.
- Check if a binary tree is BST or not
- Sorted Linked List to Balanced BST

Auxiliary Space: O(n)

Method 3 (Make two equations)
Let the numbers which are being repeated are X and Y. We make two equations for X and Y and the simple task left is to solve the two equations.
We know the sum of integers from 1 to n is n(n+1)/2 and product is n!. We calculate the sum of input array, when this sum is subtracted from n(n+1)/2, we get X + Y because X and Y are the two numbers missing from set [1..n]. Similarly calculate product of input array, when this product is divided from n!, we get X*Y. Given sum and product of X and Y, we can find easily out X and Y.

Let summation of all numbers in array be S and product be P

X + Y = S – n(n+1)/2
XY = P/n!

Using above two equations, we can find out X and Y. For array = 4 2 4 5 2 3 1, we get S = 21 and P as 960.

X + Y = 21 – 15 = 6

XY = 960/5! = 8

X – Y = sqrt((X+Y)^2 – 4*XY) = sqrt(4) = 2

Using below two equations, we easily get X = (6 + 2)/2 and Y = (6-2)/2
X + Y = 6
X – Y = 2

Thanks to geek4u for suggesting this method. As pointed by Beginer , there can be addition and multiplication overflow problem with this approach.
The methods 3 and 4 use all useful information given in the question 🙂

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

/* function to get factorial of n */
int fact(int n);

void printRepeating(int arr[], int size)
{
  int S = 0;  /* S is for sum of elements in arr[] */
  int P = 1;  /* P is for product of elements in arr[] */
  int x,  y;   /* x and y are two repeating elements */
  int D;      /* D is for difference of x and y, i.e., x-y*/
  int n = size - 2,  i;

  /* Calculate Sum and Product of all elements in arr[] */
  for(i = 0; i < size; i++)
  {
    S = S + arr[i];
    P = P*arr[i];
  }

  S = S - n*(n+1)/2;  /* S is x + y now */
  P = P/fact(n);         /* P is x*y now */

  D = sqrt(S*S - 4*P); /* D is x - y now */

  x = (D + S)/2;
  y = (S - D)/2;

  printf("The two Repeating elements are %d & %d", x, y);
}

int fact(int n)
{
   return (n == 0)? 1 : n*fact(n-1);
}

int main()
{
  int arr[] = {4, 2, 4, 5, 2, 3, 1};
  int arr_size = sizeof(arr)/sizeof(arr[0]);
  printRepeating(arr, arr_size);
  getchar();
  return 0;
}
```

Time Complexity: O(n)
Auxiliary Space: O(1)

Method 4 (Use XOR)
Thanks to neophyte for suggesting this method.
The approach used here is similar to method 2 of this post.
Let the repeating numbers be X and Y, if we xor all the elements in the array and all integers from

1 to n, then the result is X xor Y.

The 1's in binary representation of X xor Y is corresponding to the different bits between X and Y. Suppose that the kth bit of X xor Y is 1, we can xor all the elements in the array and all integers from 1 to n, whose kth bits are 1. The result will be one of X and Y.

```
void printRepeating(int arr[], int size)
{
  int xor = arr[0]; /* Will hold xor of all elements */
  int set_bit_no;  /* Will have only single set bit of xor */
  int i;
  int n = size - 2;
  int x = 0, y = 0;

  /* Get the xor of all elements in arr[] and {1, 2 .. n} */
  for(i = 1; i < size; i++)
    xor ^= arr[i];
  for(i = 1; i <= n; i++)
    xor ^= i;

  /* Get the rightmost set bit in set_bit_no */
  set_bit_no = xor & ~(xor-1);

  /* Now divide elements in two sets by comparing rightmost set
   bit of xor with bit at same position in each element. */
  for(i = 0; i < size; i++)
  {
    if(arr[i] & set_bit_no)
      x = x ^ arr[i]; /*XOR of first set in arr[] */
    else
      y = y ^ arr[i]; /*XOR of second set in arr[] */
  }
  for(i = 1; i <= n; i++)
  {
    if(i & set_bit_no)
      x = x ^ i; /*XOR of first set in arr[] and {1, 2, ...n }*/
    else
      y = y ^ i; /*XOR of second set in arr[] and {1, 2, ...n } */
  }

  printf("\n The two repeating elements are %d & %d ", x, y);
}


int main()
{
  int arr[] = {4, 2, 4, 5, 2, 3, 1};
  int arr_size = sizeof(arr)/sizeof(arr[0]);
  printRepeating(arr, arr_size);
  getchar();
  return 0;
}
```

Method 5 (Use array elements as index)

Thanks to Manish K. Aasawat for suggesting this method.

```
Traverse the array. Do following for every index i of A[].
{
check for sign of A[abs(A[i])] ;
if positive then
   make it negative by   A[abs(A[i])]=-A[abs(A[i])];
else  // i.e., A[abs(A[i])] is negative
   this   element (ith element of list) is a repetition
}
```

```
Example: A[] =  {1, 1, 2, 3, 2}
i=0;
Check sign of A[abs(A[0])] which is A[1].  A[1] is positive, so make it negative.
Array now becomes {1, -1, 2, 3, 2}

i=1;
Check sign of A[abs(A[1])] which is A[1].  A[1] is negative, so A[1] is a repetition.

i=2;
Check sign of A[abs(A[2])] which is A[2].  A[2] is  positive, so make it negative. '
Array now becomes {1, -1, -2, 3, 2}

i=3;
Check sign of A[abs(A[3])] which is A[3].  A[3] is  positive, so make it negative.
Array now becomes {1, -1, -2, -3, 2}

i=4;
Check sign of A[abs(A[4])] which is A[2].  A[2] is negative, so A[4] is a repetition.
```

Note that this method modifies the original array and may not be a recommended method if we are not allowed to modify the array.

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

void printRepeating(int arr[], int size)
{
  int i;

  printf("\n The repeating elements are");

  for(i = 0; i < size; i++)
  {
    if(arr[abs(arr[i])] > 0)
      arr[abs(arr[i])] = -arr[abs(arr[i])];
    else
      printf(" %d ", abs(arr[i]));
  }
}

int main()
{
  int arr[] = {1, 3, 2, 2, 1};
  int arr_size = sizeof(arr)/sizeof(arr[0]);
  printRepeating(arr, arr_size);
  getchar();
  return 0;
}
```

The problem can be solved in linear time using other method also, please see this and this comments

Please write comments if you find the above codes/algorithms incorrect, or find better ways to solve the same problem.

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

11 Tweet 0 2

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

96 Comments GeeksforGeeks

Sort by Newest

nathan • 3 months ago

nice solutions

• Reply • Share ›

smith • 5 months ago

#include<stdio.h>

void swap(int *a,int *b)

{

int t=*a;

*a=*b;

*b=t;

}

int obs(int x)

{

return x>0?x:-x;

}

see more

• Reply • Share ›

smith • 5 months ago

n fifth method,how to solve if 0 is present in array?

• Reply • Share ›

smith • 5 months ago

in fifth method,how to solve if 0 is present in array?

• Reply • Share ›

GCODE smith • 3 months ago

Elements must be from 0 to n-1 in that case,to use array as index.

1 • Reply • Share ›

jasleen • 10 months ago

In the second method, whenever we refer to count[a[i]], count[a[i]-1] should be where nth element is repeated.
Correct me if I'm wrong..

• Reply • Share ›

sonali gupta • 11 months ago

```
#include<stdio.h>
#include<conio.h>
int main()
{
int a[] = {4, 2, 4, 5, 2, 3, 1},i ,b[10];
for(i=0;i<=6;i++)
b[i]=0;
for(i=0;i<=6;i++)
b[a[i]]++;
for(i=0;i<=6;i++)
{if(b[i]>1)
printf("%d ",i);
}
getch();
return 0;
```

,

1 ^ | ⌄ • Reply • Share ›

shahid Rauf ➦ sonali gupta • 8 months ago

hey

can u send me solution of this task:

if we take 3*3 array and take number by rand()%8 and store these num

repeated number and romove of these number and add again non repe

please send me complete code in c language on this ID: shahidrauf.14

^ | ⌄ • Reply • Share ›

vikasnitt ➦ sonali gupta • 9 months ago

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

^ | ⌄ • Reply • Share ›

Vikas Gupta ➦ sonali gupta • 9 months ago

This is same as method 2.

^ | ⌄ • Reply • Share ›

sonali gupta • 11 months ago

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

#include<stdio.h>

#include<conio.h>

int main()

{

int a[] = {4, 2, 4, 5, 2, 3, 1},i ,b[10];

for(i=0;i<=6;i++)

b[i]=0;

```
b[a[i]]++;
for(i=0;i<=6;i++)
{if(b[i]>1)
printf("%d ",i);
}
getch();
return 0;
}
```

Reply • Share ›

krishnx • a year ago

```
 #include<iostream>
#include<map>

using namespace std;

void findRepeatingElements(int a[], int);

int main()
{
        int a[] = {4, 2, 4, 5, 2, 3, 1};
        int length = sizeof(a)/sizeof(a[0]);

        findRepeatingElements(a, length);

        return 0;
}

void findRepeatingElements(int a[], int length)
```

see more

Reply • Share ›

GeeksforGeeks • a year ago

Could you please provide an example array for which it produced wrong outpu

Reply • Share ›

Sandeep Jain • a year ago

Please take a closer look at the question. The example provided by doesn&#0
says "You are given an array of n+2 elements. All elements of the array are in

Reply • Share ›

Kingshuk Chatterjee • a year ago

Method 5 will bomb if the numbers are greater than the array size. For instanc
ArrayIndexOutOfBoundsException {1, 23, 2, 3, 2}; Why? Because, the array s
element.

Reply • Share ›

Chandra Kanta Mohapatra • a year ago

case 5 is not satisfying to all the inputs..gives wrong output..

Reply • Share ›

nathan • a year ago

in method 5, what if the duplicate repeat 3 times or more, the same result will
right???

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

Reply • Share ›

sush • a year ago

2nd method will give wrong output for case {5, 2, 4, 5, 2, 3, 1}
size of count array should be 'size-1' instead of 'size-2'

Reply · Share ›

Rajath · 2 years ago

With the below input I am getting ans as 7 and 3
3, 4, 2, 3, 4, 6,1

Where is it goign wrong ?

Reply · Share ›

Steve · 2 years ago

Input:{5,3,4,2,2,1,1}
Can you explain the answer like you did for the example in case 5

Reply · Share ›

aa · 2 years ago

@acolyte : can u further explain ur cycle detection algorithm to find duplicates

Reply · Share ›

Sourabh Goyal · 2 years ago

In method 5 suppose array A has 10 elements indexed form 0 to 9,then if valu
size of array A must be atleast the maximum of all values stored in the array.
the problem with hashing in O(n) time.

1 · Reply · Share ›

Varadharajan · 2 years ago

The neophyte method works awesome!!!!
Good use of xor logic :)
Great...

Reply · Share ›

Shahid · 2 years ago

Hi ,

In step "i=3 ; list now becomes {-1,-1,2,3,2} and A[3] is not repeated
now list becomes {-1,-1,-2,3,2}"
according to you
i= 3 and a[a[3]] is nagative so it should be repeated??

Actually I got the solutions but I think there is some mistake in this step..

^ | ⌄ • Reply • Share ›

GeeksforGeeks ➜ Shahid • 2 years ago

@Shahid:
The explanation was correct, but it was following the convention that ar
modified the example to make more comprehensive. Now indexes star
doubts.

^ | ⌄ • Reply • Share ›

Bala ➜ GeeksforGeeks • a year ago

Suppose if i have an array like this Arr={100,2,1,4,5}.

Here when i=0,
arr[arr[0]] = arr[100] = ?
total size of the array here is only 5. we will get array index out o

^ | ⌄ • Reply • Share ›

GeeksforGeeks ➜ Bala • a year ago

Please take a closer look at the problem statement. It is

^ | ⌄ • Reply • Share ›

akshaybhogra • 2 years ago

I dont understand how the following line works in method 4

```
/* Get the rightmost set bit in set_bit_no */
set_bit_no = xor & ~(xor-1);
```

^ | ˅ • Reply • Share ›

Vijay Ekkaladevi • 3 years ago

Why not insert the array elements into a binary search tree, during the insertio
element repeating element is caught. It works for characters as well.

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

^ | ˅ • Reply • Share ›

GeeksforGeeks ➜ Vijay Ekkaladevi • 2 years ago

Thanks for suggesting a new method. This method works. The worst c
O(nLogn) (if we use self balancing BST). Time complexity is better tha
in O(n) though.

^ | ˅ • Reply • Share ›

alaska • 3 years ago

i still don't get what two numbers can be used to identify and element

^ | ˅ • Reply • Share ›

GeeksforGeeks ➜ alaska • 3 years ago

@alaska: Try to run it for some examples and add some printf stateme
will all make sense to you. We have also improved the language of the
comprehensive.

^ | ˅ • Reply • Share ›

jigard • 3 years ago

i have done simple python implementation of this problem

def two_reapeat(list):
k = [1 for i in range(max(list)+1)]
for i in list:

if(k[i] == -1):
print "Rapeated number",i
else:
k[i]= -k[i]

Reply · Share ›

Swaroop · 3 years ago

S = S - n*(n+1)/2; /* S is x + y now */
P = P/fact(n); /* P is x*y now */

D = sqrt(S*S - 4*P); /* D is x - y now */
Can someone explain how to understand the equation for the variable "D"?
I am finding difficult to understand this.

Reply · Share ›

Adam Chu Swaroop · 3 years ago

x+y=S
x*y=P
D^2 = (x-y)^2 = x^2+y^2-2*x*y = (x+y)^2 - 4*x*y = S*S -4*P
D = sqrt(S*S-4*P)

Reply · Share ›

Swaroop Adam Chu · 3 years ago

Thanks Adam.

Reply · Share ›

mahe · 3 years ago

Method two is modified to terminate the loop if required two numbser are ident
loop will be avoided

```
 void printRepeating(int arr[], int size)
{
        int i, j=0;

        printf(" Repeating elements are ");

        for(i = 0; i < size; i++)
        {
                if(count[arr[i]] == 1)
                {
                        printf(" %d ", arr[i]);
                }
                else
```

see more

^ | ⌄ • Reply • Share ›

Nishant • 3 years ago

check method 4 for following input
{3,1,2,4,8,6,5,7,8,10,13,9,11,14,12,18,16,17,12,15}

^ | ⌄ • Reply • Share ›

Venki ➦ Nishant • 3 years ago

Thanks @Nishant, there is an error in the code. We will correct it.

The first for loop should start at 1, in lieu of 0, as the variable xor alreac

^ | ⌄ • Reply • Share ›

Venki • 3 years ago

Method 1 complexity to be n2.

^ | ⌄ • Reply • Share ›

```
class Program
    {
        static void Main(string[] args)
        {
            Program p = new Program();
            int[] nArray = { 4, 2, 4, 5, 2, 3, 1 };
            p.FindTwoRepeatingElement(nArray);
            Console.ReadLine();


        }

        public void FindTwoRepeatingElement(int[] nPlus2)
        {
            int[] countArray = new int[nPlus2.Length - 2];

            for (int i = 0; i < nPlus2.Length; i++)
            {
                if (countArray[nPlus2[i]-1] == 1)
```

see more

^ | v • Reply • Share ›

poonam ➦ poonam • 3 years ago

I am using "countArray[nPlus2[i]-1]" so that it wont show u array out of

Logic is:- if size of array is say n+2=7 so the size of count array will b

0,1,2,3,4. So as per Method 2 it will give error arr[i]=5..like count[arr[i]]=

not 5...it give u error....

To avoid that i have substracted 1 from the index of count array.

I hope u guys can understand it now...

Thanks

^ | v • Reply • Share ›

GeeksforGeeks → poonam • 3 years ago

@poonam:

Thanks for sharing code. Could you please add few word about the ap

^ | v • Reply • Share ›

Ravi → GeeksforGeeks • 3 years ago

If the value is **not set**(for the corresponding index in the array), **repeated**

^ | v • Reply • Share ›

GeeksforGeeks → Ravi • 3 years ago

@Ravi: This method looks same as Method 2 given in t

^ | v • Reply • Share ›

Ravi → GeeksforGeeks • 3 years ago

yeah... but i merely explained the logic... 8-)

^ | v • Reply • Share ›

sandy • 3 years ago

I am NOT sure somebody have suggested this method..

```
int arr[9] = {1,7, 9,10,4,5,4,8,9};
       int asize = sizeof(arr)/sizeof(arr[0]);
       int i,j;

       for (i=0;i<asize;i++)
       {
               for (j=(i+1);j<asize;j++)
               {
                       if ( (arr[i] - arr[j]) == 0)
                       {
```

```
                                printf("\nthe repeated element is in 
                                        " the number is %d", i,j, arr[
                        }

                }
        }
```

Reply • Share ›

GeeksforGeeks ➦ sandy • 3 years ago

@sandy:
This method is same as Method 1 of the post.

Reply • Share ›

Prateek Caire • 3 years ago

here is another O(n) solution

```
TRE()
        for each i from 0 to n-1
                while(a[i] != i+1)
                        if(a[i] != a[a[i]-1])
                                swap(i, a[i]-1)
                        else
                                print a[i]
                                a[i] = -1
                                break
```

1 • Reply • Share ›

GeeksforGeeks ➦ Prateek Caire • 3 years ago

@Prateek Caire:

Please check if your method works for the input given by Jason Huang earlier and it didn't word for the above mentioned input.

Reply • Share ›

Load more comments

Subscribe Add Disqus to your site