Sort an array of 0s, 1s and 2s

Given an array A[] consisting 0s, 1s and 2s, write a function that sorts A[]. The functions should put all 0s first, then all 1s and all 2s in last.

Example
Input = {0, 1, 1, 0, 1, 2, 1, 2, 0, 0, 0, 1};
Output = {0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 2, 2}

The problem is similar to our old post Segregate 0s and 1s in an array, and both of these problems are variation of famous Dutch national flag problem.

The problem was posed with three colours, here `0′, `1′ and `2′. The array is divided into four sections:

1. a[1..Lo-1] zeroes (red)
2. a[Lo..Mid-] ones (white)
3. a[Mid..Hi] unknown
4. a[Hi+1..N] twos (blue)

The unknown region is shrunk while maintaining these conditions

```
1. Lo := 1; Mid := 1; Hi := N;
2. while Mid <= Hi do
    1. Invariant: a[1..Lo-1]=0 and a[Lo..Mid-1]=1 and a[Hi+1..N]=2; a[Mid..Hi] are
       unknown.
    2. case a[Mid] in
        ■ 0: swap a[Lo] and a[Mid]; Lo++; Mid++
        ■ 1: Mid++
```

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

- 2: swap a[Mid] and a[Hi]; Hi–

— Dutch National Flag Algorithm, or 3-way Partitioning —

Part way through the process, some red, white and blue elements are known and are in the “right” place. The section of unknown elements, a[Mid..Hi], is shrunk by examining a[Mid]:

```
|
0 0 0 1 1 1 ? ? ? ? 2 2 2
^     ^     ^
|     |     |
Lo    Mid   Hi
```

Examine a[Mid]. There are three possibilities: a[Mid] is (0) red, (1) white or (2) blue.
Case (0) a[Mid] is red, swap a[Lo] and a[Mid]; Lo++; Mid++

```
0 0 0 0 1 1 1 ? ? ? 2 2 2
^     ^   ^
|     |   |
Lo    Mid Hi
```

Case (1) a[Mid] is white, Mid++

```
0 0 0 1 1 1 1 ? ? ? 2 2 2
^       ^   ^
|       |   |
Lo      Mid Hi
```

Case (2) a[Mid] is blue, swap a[Mid] and a[Hi]; Hi–

```
0 0 0 1 1 1 ? ? ? 2 2 2 2
^     ^   ^
|     |   |
Lo    Mid Hi
```

Geometric Algorithms

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

Continue until Mid>Hi.

```
#include<stdio.h>

/* Function to swap *a and *b */
void swap(int *a, int *b);

void sort012(int a[], int arr_size)
{
   int lo = 0;
   int hi = arr_size - 1;
   int mid = 0;

   while(mid <= hi)
   {
      switch(a[mid])
      {
         case 0:
           swap(&a[lo++], &a[mid++]);
           break;
         case 1:
           mid++;
           break;
         case 2:
           swap(&a[mid], &a[hi--]);
           break;
      }
   }
}

/* UTILITY FUNCTIONS */
void swap(int *a, int *b)
{
  int temp = *a;
  *a = *b;
  *b = temp;
}

/* Utility function to print array arr[] */
void printArray(int arr[], int arr_size)
{
  int i;
  for (i = 0; i < arr_size; i++)
    printf("%d ", arr[i]);
  printf("\n");
}

/* driver program to test */
int main()
{
  int arr[] = {0, 1, 1, 0, 1, 2, 1, 2, 0, 0, 0, 1};
  int arr_size = sizeof(arr)/sizeof(arr[0]);
  int i;

  sort012(arr, arr_size);

  printf("array after segregation ");
  printArray(arr, arr_size);

  getchar();
  return 0;
}
```

Time Complexity: O(n)

The above code performs unnecessary swaps for inputs like 0 0 0 0 1 1 1 2 2 2 2 2 : lo=4 and mid=7 and hi=11. In present code: first 7 exchanged with 11 and hi become 10 and mid is still pointing to 7. again same operation is till the mid <= hi. But it is really not required. We can put following loop before switch condition to make sure that hi is pointing to location which is not 2 so that it would eliminate unnecessary swaps of 2.

```
while ((a[hi]==2) && hi >= mid)
    –hi;
if (hi < mid)
    break;
```

Thanks to rka for suggesting this change.

Source:
http://www.csse.monash.edu.au/~lloyd/tildeAlgDS/Sort/Flag/

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

Recent Comments

Aman Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 24 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 28 minutes ago

Sanjay Agarwal bool tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 53 minutes ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 54 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

▶ Java Source Code
▶ Java Array
▶ C++ Array
AdChoices
▶ Switch Array
▶ Java Sort
▶ Array Function
AdChoices
▶ Array Function
▶ Array Pointer
▶ System Array

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

10 Tweet 0 0

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

68 Comments GeeksforGeeks

Sort by Newest

kinshuk chandra • 5 days ago

That's a good method. Here is my code(from http://k2code.blogspot.in/2010...

```
low = 0;
high = arr.length - 1;

while (low < high) {
    while (arr[low] == 0) {
        low ++;
    }
    while (arr[high] == 1) {
        high --;
    }
    if (low < low ) {
        swap arr[low], arr[high]
    }
}
```

Reply • Share ›

newbie • 6 days ago

#include<stdio.h>
//#include<conio.h>

void printarr(int arr[],int size)
{
for(int i=0;i<size;i++) {="" printf("%d\t",arr[i]);="" }="" printf("\n");="" }="" void=""
int="" s="0;" int="" e="size-1;" int="" i="0;" while(i<="e)" {="" if(arr[i]="=0)" {=""
if(arr[i]="=2)" {="" if(arr[e]="=0)" {="" arr[s]="0;" arr[e]="2;" s++;="" e--;="" }=""
;="" }="" else="" {="" e--;="" i--;="" }="" }="" i++;="" }="" for(int="" i="s;i&lt;=e;i+
printarr(arr,size);="" }="" int="" main()="" {="" int="" arr[]="{0,1,2,0,2,0,1,2,0,0,1

size="(sizeof(arr))/sizeof(arr[0]);" my_sort(arr,size);="" _getch();="" }="">

Reply • Share ›

Mohaan • 14 days ago

http://ideone.com/Be3CUm

Here we need to count the 0's, 1's & 2's in the array which takes O(n) time an
array which takes O(n) time.

If it is just printing the elements in sorted format the later O(n) can be neglecte

Reply • Share ›

Rahul Maheshwari • 24 days ago

Hi all, WE can also do this question in other manner..

#include<stdio.h>

void Sortarrays(int arr[] , int size){

int count[3] = {0,0,0};

for(int i=0 ; i<size ;="" i++){="" count[arr[i]]++;="" }="" int="" j="0;" for(int="" i="0
if(count[j]="=0){" j++;="" }="" arr[i]="j;" count[j]--;="" }="" printf("\n");="" for(int="
printf("%d="" ",arr[i]);="" }="" }="" int="" main(){="" int="" arr[]="{0," 1,="" 1,="" 0
0,="" 1};="" int="" size="sizeof(arr)/sizeof(arr[0]);" sortarrays(arr,size);="" retur

Reply • Share ›

Guest • 6 months ago

Hi all,

We can also do this Qn in this way.

Set left=0, right=n-1.

if a[left]!=0 and a[right]==0 => swap a[left] and a[right]

else if(a[left]==0) left++

and

if(a[right]!=0) right--;

keep on moving in this manner in a while loop (left<=right).

Now upon exiting this while loop, set right =n-1, but dont change the value of le

After this while loop, create another while loop (left<=right)

and use the same logic to swap 1's and 2's.

Time Complexity: O(n)

Find complete code here.

http://codepad.org/3hsgUGLY

Let me know if you find any error.

• Reply • Share ›

vivek • 7 months ago

Hello,GeeksforGeeks this is my implementation to solve the problem please c
for ur correction if it is needed

can we not solve the problem using count sort.here is the simple algorithm

void count(int a[],int n)

{

int temp[3]={0}; //temp array to store the number of 0,1,2 occurs in the array a

for(int i=0;i<n;i++) {="" temp[a[i]]++;="" }="" for="" inserting="" value="" into=""
for(i="0;i&lt;temp[0];i++)" a[j++]="temp[0];" for(i="0;i&lt;temp[1];i++)" a[j++]="te
a[j++]="temp[2];" for="" printing="" the="" array="" a[]="" in="" sorted="" order=
",a[i]);="" }="">

1 ^ | v • Reply • Share ›

manish ➦ vivek • 7 months ago

nice solution

^ | v • Reply • Share ›

vivek • 7 months ago

Hello,@GeeksforGeeks this is my implementation to solve the problem please
wait for ur correction if it is needed

can we not solve the problem using count sort.here is the simple algorithm

void count(int a[],int n)
{
int temp[3]={0}; //temp array to store the number of 0,1,2 occurs in the array a

for(int i=0;i<n;i++) {="" temp[a[i]]++;="" }="" for="" inserting="" value="" into=""
for(i="0;i&lt;temp[0];i++)" a[j++]="temp[0];" for(i="0;i&lt;temp[1];i++)" a[j++]="te
a[j++]="temp[2];" for="" printing="" the="" array="" a[]="" in="" sorted="" order=
",a[i]);="" }="">

1 ^ | v • Reply • Share ›

draganwarrior • 8 months ago

solution using partioning http://ideone.com/TefXpF
O(n) time

^ | v • Reply • Share ›

draganwarrior • 8 months ago

can This problem be solved using standered partionning method

first consider pivot ==0

then pivot ==1

1 ^ | ˅ • Reply • Share ›

wakeup123 • 10 months ago

In the explanation part which suggests way to optimize the given method,--hi is

^ | ˅ • Reply • Share ›

Amit Agarwal • 10 months ago

just take three counters and count no of 0,1 and 2 and then form an array of th

^ | ˅ • Reply • Share ›

Pankaj Goyal • 11 months ago

is it necessary that the input array must contain all three 0,1 &2? I mean can tl
etc..

^ | ˅ • Reply • Share ›

Ankit Gupta • 11 months ago

solution with O(n) time and O(1) space....require 2 traversals...
simply first put all 0s in places in one traversal then all 1s in second traversal..
for a pass make 2 pointers I and j...
suppose we are traversing to adjust all 0s then.
I always points to first non 0 element and j simply traverses the array.
put the following if condition in the traversal loop.

```
if(arr[j]==0){
swap(i, j);
while(arr[i]==0)i++;
}
```

similarly do a second pass to adjust all 1s..

^ | v • Reply • Share ›

shek8034 • 11 months ago

No need to do all this Count Sort or Dutch National Flag Algo. I have a Very sin
Time complexity and O(1) Space complexity. Works for all cases.

Algorithm:
Since we have to move all 0 to left and 2 to right, we consider only these two v
automatically get adjusted to center.
1)Take two variables start=0 and end=N-1.
2)start will store the index before which all 0's are stored.
3)end will store the index after which all 2's are stored.
4)Traverse the array and check for 0 and 2.
(a) if arr[i] is 0
check if arr[start] !=0 (then arr[start] should either be 1 or 2, so swap arr[i] with
position and increment start)
(b) if it is 2
check if arr[end] !=2 (then arr[end] should either be 0 or 1, so swap arr[i] with
position and decrement end)
5)You have sorted the array inplace with minimum no of swaps :)

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

see more

2 ^ | v • Reply • Share ›

Rohit ➦ shek8034 • 10 months ago

Nice solution :)

^ | v • Reply • Share ›

Oshonic ➦ shek8034 • 11 months ago

a more compaq code with similar approach...

```
 #include<stdio.h>
int main()
{
    int a[8] = {0,0,2,1,0,2,1,2};
    int i,j,k,t;
    for( i=0, j=0, k=7; j < k ; j++ )
    {
        if( a[j] == 0 )
        {
            t = a[i];
            a[i] = a[j];
            a[j] = t;
            i++;
        }
        if( a[j] == 2 )
        {
```

see more

Reply • Share ›

shek8034 Oshonic • 11 months ago

Your code will do some unnecessary swaps. You have to add s
my code.

1 • Reply • Share ›

Hari Prasath Nallasamy • a year ago

#include<stdio.h>
#include<algorithm>
#include<iostream>
#define N 10
using namespace std;

```
{
int array[N]={1,2,2,3,1,2,3,1,2,3}, i=0, start = 0, end = N-1;
while(i<=end)
{
if(array[i] == 1)
{
if(i!=start)
swap(array[i], array[start]);
else
i++;
start++;
}
```

see more

Reply • Share ›

anonymous • a year ago

Cant we just simply count the no. of 0s,1s and 2s in one pass and then fill the
That would also take O(n) time.
In what situation does Dutch National Flag solution prove more useful ???
Please elaborate

Reply • Share ›

Ganesh • a year ago

You can find the java code here:

```
[sourcecode language="JAVA"]
/**
* Given an array A[] consisting 0s, 1s and 2s, write a function that sorts A[].
* The functions should put all 0s first, then all 1s and all 2s in last.
* Example:
* Input = {0, 1, 1, 0, 1, 2, 1, 2, 0, 0, 0, 1};
```

* Output = {0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 2, 2}
*
* @author GAPIITD
*
*/
public class SortAnArrayOf0s1sAnd2s {

public static void main(String[] args) {
int arr[] = {0, 1, 1, 0, 1, 2, 1, 2, 0, 0, 0, 1};
sortAnArrayOf0s1sAnd2s(arr);

see more

Reply • Share ›

vick • 2 years ago

i think the statements under "case 0" should be like this..

if(lo != mid)
swap(&a[lo],&a[mid]);

lo++;
mid++;

this wil avoid the unnecessary call to the swap function as in the case when th

plz crct me if i m wrong..

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

Reply • Share ›

vick • 2 years ago

i think the statements under "case 0" should be like this..

if(lo != mid)
swap(&a[lo],&a[mid]);

lo++;
mid++;

this wil avoid the unnecessary call to the swap function as in the case when th

plz crct me if i m wrong..

• Reply • Share ›

Harjit SIngh • 2 years ago

```
 /* Paste your code here (You may delete these lines if not writing co
#include<stdio.h>

#include<stdlib.h>

void swap(int a[],int first,int second)

{

    int temp;

    temp=a[first];

    a[first]=a[second];

    a[second]=temp;

}
```

see more

⌃ | ⌄ • Reply • Share ›

Mohammad Shahid ➦ Harjit SIngh • a year ago

above code will break with following input
2 0 2 2 2 0 1 0 2 1

⌃ | ⌄ • Reply • Share ›

Shashi • 2 years ago

Do it with the help of singly linked list...

⌃ | ⌄ • Reply • Share ›

red • 2 years ago

We can use two indexes. One from the left and one from the right.
Move both the indexes towards each other. Swap when the right index has 0 a
0's on the left.
Now do the same for the remaining array once again for 1's and 2's.
O(n) and In place.

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

⌃ | ⌄ • Reply • Share ›

Shyam ➦ red • 2 years ago

Dude you needn't repeat it twice for 1's and 2's... if you repeat the loop
have 2's in the correct positions

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

⌃ | ⌄ • Reply • Share ›

swetha • 3 years ago

void segregate(int a[30],int n)

```
int count[3]={0};
for(int i=0;i<n;i++)
count[a[i]]++;

int flag=0;
for(int i=0;i<3;i++)
for(int j=0;j<count[i];j++)
a[flag++]=i;
}
```

Reply • Share ›

KK123 • 3 years ago

CHECK THIS ONE:

```
 int i = 0, s = 0, last = n-1;
while(i<=last){
if(a[i] == 0 && i!=s)
{
       swap(a[i], a[s]);
       s++;
}
else if(a[i] == 2 and i!=last)
{
       swap(a[i], a[last]);
       last--;
}
else
       i++;
}
```

Reply • Share ›

Algoseekar • 3 years ago
hi GeeksforGeeks I Tried National Flag Algo for Doubly Linked lts but it Not Wc
my algo & let me know problems in this

sort doubly linked list of 0,1,2

running version of the code https://ideone.com/gx3GF
^ | ⌄ • Reply • Share ›

Algoseekar ➜ Algoseekar • 3 years ago
@sandeep,GeeksforGeeks

I Tried National Flag Algo for Doubly Linked lts but it Not Working for Dl
let me know problems in this

sort doubly linked list of 0,1,2

running version of the code https://ideone.com/gx3GF
^ | ⌄ • Reply • Share ›

qbeing • 3 years ago
Is the suggested solution doing a stable sort?
^ | ⌄ • Reply • Share ›

kartik ➜ qbeing • 3 years ago
Yes, it is stable and I think that is the reason this method should be pre
Shekhu when 0, 1 and 2 are keys.
^ | ⌄ • Reply • Share ›

Algoseekar ➜ kartik • 3 years ago
@kartik...hi I Tried National Flag Algo for Doubly Linked lts but it
a look at my algo & let me know problems in this

sort doubly linked list of 0,1,2

running version of the code https://ideone.com/gx3GF

Reply • Share ›

qbeing kartik • 3 years ago

Shekhu's solution rewrites the array, so its definitely not stable.

Reply • Share ›

qbeing qbeing • 3 years ago

I was trying to work through this solution to check if its s

As the last block of 2s is written from end, and value of
2s get reversed (2s will be in reverse order of there app

Reply • Share ›

kartik qbeing • 3 years ago

Yes, that is the case. I was wrong in my previous comm

Reply • Share ›

qbeing qbeing • 3 years ago

Also, the order of 1s would also not be retained. Only th
appearance.

Am I missing something in the solution ?

Reply • Share ›

Rajiv • 4 years ago

Simplest way is to count the number of 0s, 1s and 2s and then just fill the resu
[sourcecode language="java"]
public static int[] sort(int[] a) {
if (a != null && a.length > 0) {

```
int oneCount = 0;
int twoCount = 0;

//Find out the count for 0, 1 and 2
for (int i = 0; i < a.length; i++) {
if (a[i] == 0) {
zeroCount++;
} else if (a[i] == 1) {
oneCount++;
} else if (a[i] == 2) {
twoCount++;
} else {
throw new RuntimeException("Array cannot contain anything but 0, 1 or 2");
```

see more

Reply • Share ›

evo Rajiv • 6 months ago

{{{

```
public static int[] sort(int[] a) {

int numCounts[3]=new int[]{0,0,0};
int tail=0;

for (int i = 0; i < a.length; i++) {
numCounts[ a[i] ]++;
}

for (int i = 0; i < numCounts.length; i++) {
count=numCounts[i];
tail+=count;
```

a[j]=i;
}

return a;
}

}}}

Reply • Share ›

Sandeep Rajiv • 4 years ago
@Rajiv: Thanks for sharing code, the method is simple indeed. The m
problem of sorting an array of 0s, 1s and 2s, but this method won't wor
some records. For example, sorting all columns in MS Excel according
1s and 2s.

Reply • Share ›

ap Sandeep • 3 years ago
@Sandeep : Can u please explain in more detail what differenc
keys rather not the elements of an array.

Thank you

Reply • Share ›

Sreenivasan AC ap • 5 months ago
assume the array values are the keys to objects
class student
{
int subject_code; //KEY_VALUE 0/1/2
char name[100];
};
here subject_code is key value (0,1,2) with which we ar
replacing will not work. We cant replace subject_code t

^ | v • Reply • Share ›

ap ➦ ap • 3 years ago
do u mean that if they are keys, they can not be stored i
^ | v • Reply • Share ›

donbosio • 4 years ago
hey . to run through the loop u have used sizeof(arr)/sizeof(arr[0]) but if http://http://geeksforgeeks.org/?p=65... u will find that it is wrong . plz el am i getting something wrong?
^ | v • Reply • Share ›

rka • 4 years ago

First i think the loop term condition should be mid<=hi is correct. Becas result would not be correct if the loop condition is not mid <=hi
0 0 0 1 1 1 1 2 0 2 2 :: where lo=3; mid=8 and hi=9
if condition is mid <hi then output would be:
0 0 0 1 1 1 1 0 2 2 2
Because loop will terminate as soon as it replace hi with mid both becoi
But if the condition is mid <= hi then output would be:
0 0 0 0 1 1 1 1 2 2 2
because after swaping the 8 and 9th member it will again loop for 8th me and exchange it with 3th member.

Also, i think inside the while loop there should be one more loop before hi is pointing to member which is not 2 otherwise it is neccessary perfoi for example in below condition:
0 0 0 0 1 1 1 2 2 2 2 2 : lo=4 and mid=7 and hi=11
Now in present code: first 7 exchanged with 11 and hi become 10 and m operation is till the mid <= hi.

Reply • Share ›

GeeksforGeeks → rka • 4 years ago

@rka: Thanks for suggesting the optimization. We have included it to t

Reply • Share ›

Rohit → GeeksforGeeks • 10 months ago

@rka and @geeksforgeeks: How is lo=4 until we have a similar
before switch statement? I think we need to add the following lo
number of swaps 0 in your example(already sorted).
while ((a[lo]==0) && lo mid)
break;

Please let me know if I am missing something.

Reply • Share ›

rbk • 4 years ago

I think we can use the counting sort here, because the range of numbers in the
requirement will be very less as as there are only 3 types of numbers.

Count the number of 0s, 1s and 2s and change the array accordingly.
Time Complexity O(n)
Space complexity O(1)

Reply • Share ›

Load more comments

Subscribe   Add Disqus to your site