GeeksforGeeks

A computer science portal for geeks

GeeksQuiz

Login

Home Algorithms DS GATE Interview Corner Q&A C C++ Java Books Contribute Ask a Q About

Array Bit Magic C/C++ Articles GFacts Linked List MCQ Misc Output String Tree Graph

Maximum Length Bitonic Subarray

Given an array A[0 … n-1] containing n positive integers, a subarray A[i … j] is bitonic if there is a k with i <= k <= j such that A[i] <= A[i + 1] ... <= A[k] >= A[k + 1] >= .. A[j - 1] > = A[j]. Write a function that takes an array as argument and returns the length of the maximum length bitonic subarray.

Expected time complexity of the solution is O(n)

Simple Examples

1) A[] = {12, 4, 78, 90, 45, 23}, the maximum length bitonic subarray is {4, 78, 90, 45, 23} which is of length 5.

2) A[] = {20, 4, 1, 2, 3, 4, 2, 10}, the maximum length bitonic subarray is {1, 2, 3, 4, 2} which is of length 5.

Extreme Examples

1) A[] = {10}, the single element is bitnoic, so output is 1.

2) A[] = {10, 20, 30, 40}, the complete array itself is bitonic, so output is 4.

3) A[] = {40, 30, 20, 10}, the complete array itself is bitonic, so output is 4.

Solution

Let us consider the array {12, 4, 78, 90, 45, 23} to understand the soultion.

1) Construct an auxiliary array inc[] from left to right such that inc[i] contains length of the nondecreaing subarray ending at arr[i].

For for A[] = {12, 4, 78, 90, 45, 23}, inc[] is {1, 1, 2, 3, 1, 1}

2) Construct another array dec[] from right to left such that dec[i] contains length of

Interview Experiences

Advanced Data Structures

Dynamic Programming

Greedy Algorithms

Backtracking

Pattern Searching

Divide & Conquer

Mathematical Algorithms

Recursion

nonincreasing subarray starting at arr[i].
For A[] = {12, 4, 78, 90, 45, 23}, dec[] is {2, 1, 1, 3, 2, 1}.

3) Once we have the inc[] and dec[] arrays, all we need to do is find the maximum value of (inc[i] + dec[i] – 1).
For {12, 4, 78, 90, 45, 23}, the max value of (inc[i] + dec[i] – 1) is 5 for i = 3.

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

int bitonic(int arr[], int n)
{
    int i;
    int *inc = new int[n];
    int *dec = new int[n];
    int max;
    inc[0] = 1; // The length of increasing sequence ending at first i
    dec[n-1] = 1; // The length of increasing sequence starting at fir

    // Step 1) Construct increasing sequence array
    for(i = 1; i < n; i++)
    {
        if (arr[i] > arr[i-1])
            inc[i] = inc[i-1] + 1;
        else
            inc[i] = 1;
    }

    // Step 2) Construct decreasing sequence array
    for (i = n-2; i >= 0; i--)
    {
        if (arr[i] > arr[i+1])
            dec[i] = dec[i+1] + 1;
        else
            dec[i] = 1;
    }

    // Step 3) Find the length of maximum length bitonic sequence
    max = inc[0] + dec[0] - 1;
    for (i = 1; i < n; i++)
    {
        if (inc[i] + dec[i] - 1 > max)
        {
            max = inc[i] + dec[i] - 1;
        }
    }
```

Geometric Algorithms

Popular Posts

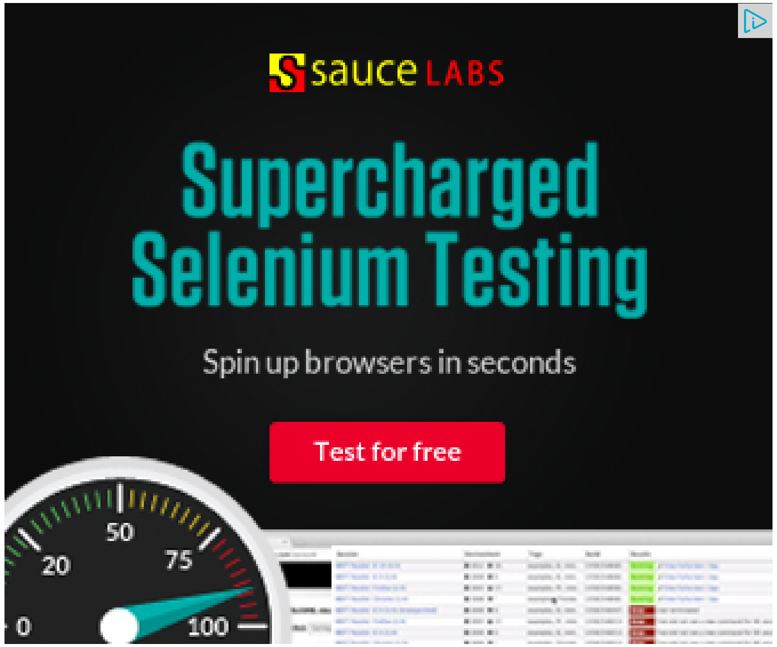
All permutations of a given string
Memory Layout of C Programs
Understanding “extern” keyword in C
Median of two sorted arrays
Tree traversal without recursion and without stack!
Structure Member Alignment, Padding and Data Packing
Intersection point of two Linked Lists
Lowest Common Ancestor in a BST.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

```
    // free dynamically allocated memory
    delete [] inc;
    delete [] dec;

    return max;
}

/* Driver program to test above function */
int main()
{
    int arr[] = {12, 4, 78, 90, 45, 23};
    int n = sizeof(arr)/sizeof(arr[0]);
    printf("\n Length of max length Bitnoic Subarray is %d", bitonic(a
    getchar();
    return 0;
}
```

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

As an exercise, extend the above implementation to print the longest bitonic subarray also. The above implementation only returns the length of such subarray.

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



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

8 Tweet 0 2

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

73 Comments GeeksforGeeks

Sort by Newest ▾

Join the discussion

Elastic SQL DB

Certified Ethical Hacker

Free Downloadable

705 Subscribe

Recent Comments

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

Write a C program to Delete a Tree. · 15 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 19 minutes ago

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

Root to leaf path sum equal to a given number · 44 minutes ago

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

Count trailing zeroes in factorial of a number · 45 minutes ago

Join the discussion…

OP Coder · 13 days ago

My solution with Time Complexity O(n) and Space Complexity O(1)

```
#include <iostream>
using namespace std;
int maxBiotonicSubArray(int a[], int n)
{
bool maxim_found = false;
int max_start, max_end, max_len;
int tmp_start, tmp_end;

int prev = a[0];
max_start = max_end = 0;
tmp_start = tmp_end = 0;

for (int i = 1; i < n; i++)
{
if (a[i] > prev)
{
if (maxim_found)// we have found the end of the current biotonic sub array
```

see more

Reply · Share ›

Gaurav Gupta · a month ago

Here is simple solution which takes O(n) times and constant extra space.
Correct me If wrong...

```
#include<stdio.h>
int max(int a, int b){
return a>b ? a :b ;
```

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

```
}
int main(){
int max_length_so_far = 2, currnt_length = 2, flag = 0, i;
int a[] = {1,2,45,21,4,7,8,1,2,3,9,0,12,23,34,45};
int size = sizeof(a)/sizeof(a[0]);

// a single element array and size 2 array is always bitonic
// so there is no checking for those arrays so min length of
// bitonic array is set to 2.
if(size < 3) return size;

flag = a[1] < a[0] ? 1 : 0 ;
for(i = 2 ; i< size ; i++){
```

see more

Reply • Share ›

Guest • 4 months ago

The method for finding the inc and dec above is flawed. Either change the defi the logic behind it.

Reply • Share ›

Himanshu • 5 months ago

Hi This is not working for the following test case: {1, 11, 2, 10, 4, 5, 2,1} .Your algo is printing the o/p as 4 whereas the max. bitonic sequence in the ar be 6.

Reply • Share ›

chaks ➦ Himanshu • 4 months ago

The question is to print the maximum length of "bitonic subarray" i.e. (4

Reply • Share ›

therock ➦ chaks • a month ago

Please explain. Why it is {4,5,2,1} ? and Why (1,11,10,4,2,1.) is

⌃ | ⌄ • Reply • Share ›

chaks ➦ therock • a month ago

Subarray is sonething which is contiguous subsequenc
continuous in the given sequence.

⌃ | ⌄ • Reply • Share ›

sandeep • 6 months ago

why to find keep aux array of in and dec.
Not just keeping track of max bitonic seq seen so far enuf and updating it wen
O(1) space complexity

⌃ | ⌄ • Reply • Share ›

CP • 6 months ago

Here's a solution with Time Complexity: O(n)
&
Space Complexity : O(1).

http://codepad.org/C8AjYnaQ

1 ⌃ | ⌄ • Reply • Share ›

Sahil • 6 months ago

Here's a solution with Time Complexity: O(n)
&
Space Complexity : O(1).

```
#include<iostream>

using namespace std;

int bitonic(int a[], int n);
int main()

{

int a[] = {20, 4, 1, 2, 3, 4, 2,100,1,2,3,4,5,6,5,4,3,2,1};

int n=sizeof(a)/sizeof(int);

cout<<bitonic(a,n); return="" 0;="" }="" int="" bitonic(int="" a[],="" int="" n)="" {=
while(i<n)="" {="" while(i<n)="" {="" if(a[i+1]<a[i])="" {="" i++;="" }="" else="" br
```

see more

• Reply • Share ›

Rahul Verma • 6 months ago

I think the logic for finding the increasing and decreasing sub sequence is flaw
the increasing sub sequence for array {12, 4, 78, 90, 45, 23} should be {1, 1, 2
finding this correct increasing and decreasing sub sequence will take O(n^2) t

• Reply • Share ›

Innocent • 7 months ago

Why is this algorithm working, pls some one explain logic

• Reply • Share ›

Marsha Donna • 7 months ago

@GeeksforGeeks pls help,...i m not able to copy paste c code from vc++ as c
comment it is not displayed correctly

• Reply • Share ›

Guest • 7 months ago

can sum1 pls validate this

```
void bitonic(int arr[],int n)
{
```

```
{
int i,diff[20],str_ind,end_ind,max_diff=0;
if(n==1)
printf("arr is bitonic\n");

for(i=0;i<n-1;i++) check="" if="" entire="" arr="" asc="" then="" it="" is="" bitoni
break;
}
if(i==n-1)
printf("arr is bitonic\n");

for(i=0;i<n-1;i++) check="" if="" entire="" arr="" is="" in="" desc="" order="" {='
if(i="=n-1)" printf("arr="" is="" bitonic\n");="" for(i="0;i&lt;n-1;i++)" diff[i]="arr[i+1
if(diff[i]="">=0)
{
str_ind=i;
```

see more

Reply • Share ›

Guest • 7 months ago

can sum1 pls validate the following code

```
void bitonic(int arr[],int n)
{
int i,diff[20],str_ind,end_ind,max_diff=0;
if(n==1)
printf("arr is bitonic\n");

for(i=0;i<n-1;i++) check="" if="" entire="" arr="" asc="" then="" it="" is="" bitoni
break;
}
if(i==n-1)
printf("arr is bitonic\n");
```

```
for(i=0;i<n-1;i++) check="" if="" entire="" arr="" is="" in="" desc="" order="" {='
if(i="=n-1)" printf("arr="" is="" bitonic\n");="" for(i="0;i&lt;n-1;i++)" diff[i]="arr[i+1
if(diff[i]="">=0)
{
str_ind=i;
do{i++;}while(diff[i]>=0);
if(diff[i]<0)
do{i++;}
while(diff[i]<0&&i<n-1); end_ind="i;" max_diff="max(max_diff,end_ind-str_ind);
str_ind="i;" do{i++;}while(diff[i]<0);="" end_ind="i;" max_diff="max(max_diff,en
printf("\n\n%d\n",max_diff+1);="" }="">
```

• Reply • Share ›

Pawan Parekh • 8 months ago

Sudhanshu , try for {1,11,2,10,4,5,2,1} it will give wrong output 4 , The right out

• Reply • Share ›

sunil ➦ Pawan Parekh • 8 months ago

{1,2,10,5,2,1} is not a subarray.

• Reply • Share ›

SudhanshuAnand • 9 months ago

This is a simple solution in linear time and constant space.
We can keep the count till the condition is satisfied and check it with max Leng

```
int bitonic(int a[], int size){
        int maxLength = 0;
        int flag = 0, count = 1;
        for(int i= 0 ;i<size-1;i++){
                if(flag == 0){
                        if(a[i]<=a[i+1]){
                                count++;
                        }else if(a[i]>a[i+1]){
                                flag = 1;
                                count++;
                        }
                }
                else if(flag == 1){
                        if(a[i]>=a[i+1]){
                                count++;
```

see more

3 ^ | ⌄ • Reply • Share ›

Om Prakash Suthar • 9 months ago

Nitin Mittal: This is sequence ..what you are suggesting is sub sequence..hope

^ | ⌄ • Reply • Share ›

trying • 10 months ago

[sourcecode language="JAVA"]
public static int maxBitonicSubArray(int[] a)
{
if(a.length == 0)
{
throw new IllegalArgumentException();
}
boolean isIncreasing = true;
int count = 1, max = Integer.MIN_VALUE;
for(int i=1; i<a.length; i++)
{
if(a[i]>a[i-1])
{
if(!isIncreasing)

{
count = 2;
isIncreasing = true;
}

see more

^ | ⌄ • Reply • Share ›

Nitin Mittal • 10 months ago

The correct bitonic sequence should be {80, 60, 30, 20, 10} which makes the

^ | ⌄ • Reply • Share ›

vishal • 10 months ago

Another code with space O(1) and Time O(n)

```
 #include<stdio.h>
struct parameters
{
    int start;
    int end;
    int count;
};
typedef struct parameters parameters;

parameters bitonic(int a[] , int n)
{
        int count, start = -1 , end = -1;
        parameters p;
        p.start = p.end = p.count = -1;
        int i = 0;
        if( n == 1)
```

see more

⌃ | ⌄ • Reply • Share ›

skulldude • 10 months ago

The following code runs in O(n) time and used O(1) space.
It is based on the following observation:

If [i,j] is a bitonic subarray that starts at i and ends at j, then it is enough to star
subarray. You can observe it by drawing some examples.

The following code uses that fact to run in O(n) time and O(1) space complex

```
void maxBitonicSubarray(int a[],int len){

        int i=0,maxLen=1,start=0;

        while(i<len){
                int j=i+1;

                // finding an increasing subarray from i
                while(j<len && a[j]>a[j-1])
                        ++j;
```

see more

⌃ | ⌄ • Reply • Share ›

Sandeep Jain • 10 months ago

4 seems to be the correct answer as the longest Bitonic Subsequeunce is {30
missed anything.

⌃ | ⌄ • Reply • Share ›

Arjita Mahajan • 10 months ago

the output for {80 60 30 40 20 10 } should be 5 but on executing the above cod

^ | ⌄ • Reply • Share ›

sush • 10 months ago

```
int bitonic(int a[], int n)
{
  int i=1,ml=-1,d=0,cl=1,k;
    for(k=1;k<n;++k)
    {
        if(i)
        {
            if(a[k]>a[k-1])
                cl++;
            else
            {
                d=1;i=0;cl++;
            }
        }
        if(d)
        {
            if(a[k]<a[k-1])
```

see more

^ | ⌄ • Reply • Share ›

Sandeep Jain • 11 months ago

Please take a closer look at the definition of Bitonic Subarray. It should be first

^ | ⌄ • Reply • Share ›

Pankaj Goyal • 11 months ago

@sandeep jain

shouldn&#039t the answer for array A[] = {20, 4, 1, 2, 3, 4, 2, 10} be equal to 6 question. the reqd. subarray is {20,4,1,2,3,4}.

Reply • Share ›

Aditya Ambashtha • 11 months ago

O(n) implementation for constant space

```
 #include<stdio.h>

int returnMaxBitonicLength(int *,int);
int higg(int *,int);
int loww(int *,int);
int max(int,int);

int main()
{
    int maxl;
    int arr[]={12, 4, 78, 90, 45, 23};

    maxl=returnMaxBitonicLength(arr,6);

    printf("Max bitonic size = %d\n",maxl);
```

see more

Reply • Share ›

Aditya Ambashtha ➦ Aditya Ambashtha • 11 months ago

change this

if (arr[0]>arr[1]) /*Decide whether arr[0] is high or low*/
hig=arr[0];
else

```
low=arr[0];
```

to this

```
if (arr[0]>arr[1]) /*Decide whether arr[0] is high or low*/
hig=0;
else
low=0;
```

⌃ | ⌄ • Reply • Share ›

Aditya Ambashtha ➦ Aditya Ambashtha • 11 months ago

change this

```
if (arr[0]>arr[1]) /*Decide whether arr[0] is high or low*/
hig=arr[0];
else
low=arr[0];
```

to this

```
if (arr[0]>arr[1]) /*Decide whether arr[0] is high or low*/
hig=0;
else
low=0;
```

⌃ | ⌄ • Reply • Share ›

varr • a year ago

There might not be a need for additional arrays.
Is this solution correct.If yes, please tell me.(I shall give the explanation.)

```
//max length bitonic subarray
#include<stdio.h>
int maxlenbiseq(int arr[],int n)
{
```

```
int maxlen=1,curlen=1;
int i,prev=arr[0],flag=0,beg=0,maxbeg=0;
for(i=1;i<n;i++) {="" if(flag="=0)" {="" if(arr[i]="">=prev)
curlen++;
else
{
curlen++;
flag=1;
}
}
else //flag==1
{
```

see more

^ | ˅ • Reply • Share ›

varr ➦ varr • a year ago

some part is missing..

```
int maxlenbiseq(int arr[],int n)
{
int maxlen=1,curlen=1;
int i,prev=arr[0],flag=0,beg=0,maxbeg=0;
for(i=1;i<n;i++) {="" if(flag="=0)" {="" if(arr[i]="">=prev)
curlen++;
else
{
curlen++;
flag=1;
}
}
else //flag==1
{
if(arr[i]<=prev)
if(arr[i]<=prev)
curlen++;
else
```

see more

˄ | ˅ • Reply • Share ›

varr ➦ varr • a year ago

for loop has not come again:

```
for(i=1;i<n;i++) {="" ....="" }="">
```

˄ | ˅ • Reply • Share ›

varr • a year ago

There might not be a need for additional arrays.

Is this solution correct.If yes, please tell me.(I shall give the explanation.)

```
//max length bitonic subarray
#include<stdio.h>
int maxlenbiseq(int arr[],int n)
{
int maxlen=1,curlen=1;
int i,prev=arr[0],flag=0,beg=0,maxbeg=0;
for(i=1;i<n;i++) {="" if(flag="=0)" {="" if(arr[i]="">=prev)
curlen++;
else
{
curlen++;
flag=1;
}
}
else //flag==1
{
```

see more

^ | ⌄ • Reply • Share ›

abhishek08aug • a year ago

What a messy solution!! Here I give you all a simple one. Cheers! ;)

```
 #include<stdio.h>

int find_max_bitonic_length(int array[], int n) {
  int i;
  int max_bitonic_length, current_bitonic_length;

  for(i=0, max_bitonic_length=0, current_bitonic_length=0;i<n; i++) {
      current_bitonic_length++;
      if(current_bitonic_length>max_bitonic_length)
        max_bitonic_length=current_bitonic_length;

      if(i-1>=0 && array[i-1]>array[i] && i+1<=n-1 && array[i]<array[
        current_bitonic_length=1;
  }
  return max_bitonic_length;
}
```

see more

^ | ⌄ • Reply • Share ›

rakitic ➦ abhishek08aug • 10 months ago

we can use only decrease variable , if no is smaller then previous , dec
previous and decrease =1...that means one length is complete , now n
in this way.....did you even got what i said ?? reply plz

^ | ⌄ • Reply • Share ›

abhishek08aug ➦ abhishek08aug • a year ago

Another optimization to ensure that maximum_bitonic_length is not up
ripple I mean a sequence of increase+decrease :)

```
 #include<stdio.h>

int find_max_bitonic_length(int array[], int n) {
  int i;
  int max_bitonic_length, current_bitonic_length;

  for(i=0, max_bitonic_length=0, current_bitonic_length=0;i<n;
      current_bitonic_length++;

      if(i-1>=0 && array[i-1]>array[i] && i+1<=n-1 && array[i]
        if(current_bitonic_length>max_bitonic_length)
          max_bitonic_length=current_bitonic_length;
        current_bitonic_length=1;
      }
  }
```

see more

^ | ⌄ • Reply • Share ›

bhavneet • a year ago

time complexity o(n)
space o(1)
pls help to find if any problem with this algo

```
 #include<stdio.h>
int func(int arr[], int n)
{
        int max=0;
        int i, inc=1, dec=0;
        for( i=2; i<n; i++)
        {
                if ( arr[i]< arr[i-1])
                        dec++;
                else if ( dec>0)
                {
                        if( inc +dec >max)
                                max = inc+ dec;
                        dec=0;
```

see more

^ | ˅ • Reply • Share ›

hunter ➦ bhavneet • 11 months ago

good one........

^ | ˅ • Reply • Share ›

bhavneet ➦ bhavneet • a year ago

will only work for strictly increasing and decreasing

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

^ | ˅ • Reply • Share ›

Arvind B R • a year ago

Failing test case :
input : 2,2,2,2,2,2,2,2,2,1
expected output : 9
received output : 2
check this link
http://ideone.com/uneqG5

^ | ˅ • Reply • Share ›

bhavneet ➦ Arvind B R • a year ago

make the following changes in code
if (arr[i] >= arr[i+1])
dec[i] = dec[i+1] + 1;
else if (arr[i] <= arr[i+1])
dec[i] = 1;

n similarly for inc

^ | ˅ • Reply • Share ›

stupid • a year ago

Here I have a doubt. If numbers are not unique like 12 4 4 78 90 90 45 30. The
be 4 4 78 90 90 45 30. Then the If condition should be if(a[i-1]=a[i+1]) for right.

^ | ˅ • Reply • Share ›

stupid ➦ stupid • a year ago

Condition should be if(a[i-1]=a[i+1]) for right. Please correct me If I am

^ | ˅ • Reply • Share ›

Sandeep Jain • a year ago

Please take a closer look at the problem statement. It says subarray, not subs
not a subarray.

^ | ˅ • Reply • Share ›

Sambuddha Datta • a year ago

Hi,
What would be the bitonic sequence of an array{1,2,4,0,3,8,6,5,9}.

Executing the above code I am getting the answer as 5 but I think that the seq
6 and not 5.

^ | ˅ • Reply • Share ›

sam • a year ago

Hi,
What would be the bitonic sequence of an array{1,2,4,0,3,8,6,5,9}
Executing the above code gives me the answer as 5 but I think there exist a se
length will be 6.

Kindly share your thoughts.

^ | ⌄ • Reply • Share ›

GeeksforGeeks ➦ sam • a year ago

Please see the above reply.

^ | ⌄ • Reply • Share ›

rakitic ➦ GeeksforGeeks • 10 months ago

we can use only decrease variable , if no is smaller then previo
previous and decrease =1...that means one length is complete
updating in this way.....did you even got what i said ?? reply plz

^ | ⌄ • Reply • Share ›

Load more comments

✉ Subscribe   Ⓓ Add Disqus to your site