# Dynamic Programming | Set 14 (Maximum Sum Increasing Subsequence)

Given an array of n positive integers. Write a program to find the sum of maximum sum subsequence of the given array such that the intgers in the subsequence are sorted in increasing order. For example, if input is {1, 101, 2, 3, 100, 4, 5}, then output should be 106 (1 + 2 + 3 + 100), if the input array is {3, 4, 5, 10}, then output should be 22 (3 + 4 + 5 + 10) and if the input array is {10, 5, 4, 3}, then output should be 10

**Solution**
This problem is a variation of standard Longest Increasing Subsequence (LIS) problem. We need a slight change in the Dynamic Programming solution of LIS problem. All we need to change is to use sum as a criteria instead of length of increasing subsequence.

Following is C implementation for Dynamic Programming solution of the problem.

```
/* Dynamic Programming implementation of Maximum Sum Increasing
   Subsequence (MSIS) problem */
#include<stdio.h>

/* maxSumIS() returns the maximum sum of increasing subsequence in arr[] of
   size n */
int maxSumIS( int arr[], int n )
{
   int *msis, i, j, max = 0;
   msis = (int*) malloc ( sizeof( int ) * n );

   /* Initialize msis values for all indexes */
   for ( i = 0; i < n; i++ )
      msis[i] = arr[i];

   /* Compute maximum sum values in bottom up manner */
   for ( i = 1; i < n; i++ )
      for ( j = 0; j < i; j++ )
         if ( arr[i] > arr[j] && msis[i] < msis[j] + arr[i])
            msis[i] = msis[j] + arr[i];

   /* Pick maximum of all msis values */
   for ( i = 0; i < n; i++ )
      if ( max < msis[i] )
         max = msis[i];

   /* Free memory to avoid memory leak */
   free( msis );

   return max;
}

/* Driver program to test above function */
int main()
{
  int arr[] = {1, 101, 2, 3, 100, 4, 5};
  int n = sizeof(arr)/sizeof(arr[0]);
  printf("Sum of maximum sum increasing subsequence is %d\n",
         maxSumIS( arr, n ) );

  getchar();
  return 0;
}
```

Time Complexity: O(n^2)

Source: [Maximum Sum Increasing Subsequence Problem]

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

## Related Topics:

- Find Union and Intersection of two unsorted arrays
- Pythagorean Triplet in an array
- Maximum profit by buying and selling a share at most twice
- Design a data structure that supports insert, delete, search and getRandom in constant time
- Print missing elements that lie in range 0 – 99
- Iterative Merge Sort
- Group multiple occurrence of array elements ordered by first occurrence
- Given a sorted and rotated array, find if there is a pair with a given sum

Tags: Dynamic Programming



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



**StrictMath** • 2 months ago

I believe it can be done in O(NlogN), if we create a BST along as we traverse the array. The key in the BST node, would contain indices to the array elements, however the BST would be created not based on the indices, but the value in the array corresponding to a particular index.

Next, when we arrive at any index 'i' in the array, we search for a particular index in the BST such that, arr[required_index] < arr[i]. This can be done in O(logN) in a BST. For every 'i' in the array, we can update the maximum sum ending at 'i', in arr[i], by : arr[i] += arr[required_index].



**Nikhil Jain** • 2 months ago

http://ideone.com/0sD8Uk

It is nlogn solution tell me if i am wrong



**Piacentero** • 6 months ago

How can I find the increasing subsequence with maximum sum?
For example: input {1, 101, 2, 3, 100, 4, 5}. output {1,2,3,100} because the sum is 106 which is maximum. Thanks

**Rahul Jain** • 8 months ago

why is there the necessity to include this condition :
msis[i]<msis[j]+a[i] ?????="">

3 ▲ | ▼ • Reply • Share ›

**lucy** ➦ Rahul Jain • 4 months ago

yaa, it is necessary since it is calculating increasing order.
1,5,3,100 without it ,add 1,5,3 in 100 and it is not right

▲ | ▼ • Reply • Share ›

**Jun** • 9 months ago

case which handles all positive,negative and every other case

http://ideone.com/YC6IyIhttp:/...

▲ | ▼ • Reply • Share ›

**prashant** ➦ Jun • 3 months ago

i dont actually see any change in your code.

▲ | ▼ • Reply • Share ›

**nk** • 10 months ago

**@GeeksforGeeks** can we solve this problem in less than n^2 .?

▲ | ▼ • Reply • Share ›

**www** ➦ nk • 8 months ago

yes we can do it in NlogN
http://www.geeksforgeeks.org/d...

▲ | ▼ • Reply • Share ›

**Random** ➦ www • 8 months ago

That's not largest sum.

1 ▲ | ▼ • Reply • Share ›

**aruna** • 10 months ago

Intializing j with i-1 would be better

▲ | ▼ • Reply • Share ›

**Vãîbhåv Joshî** • 10 months ago

Dp solution in java...

http://ideone.com/RkqnMS

▲ | ▼ • Reply • Share ›

**abrao** • 10 months ago

**shree** • 10 months ago

I think same solution should work for an array consisting of both negatvi and positive elements....am i correct?

• Reply • Share ›

**Jun** ➚ shree • 9 months ago

na...some things needs to be changed....i will post the code soon

• Reply • Share ›

**yelpz** • 10 months ago

unable to understand how this code is working ..can anybody expl-ain . what will be the elements of msis[] after processing thro-ugh this code

```
for ( i = 1; i < n; i++ )
for ( j = 0; j < i; j++ )
if ( arr[i] > arr[j] && msis[i] < msis[j] + arr[i])
msis[i] = msis[j] + arr[i];
```

• Reply • Share ›

**tushar** ➚ yelpz • 10 months ago

take a test case

1 101 1 2 3 4 5
initialize msis[] array to -1

msis[]= 1 102 1 3 6 10 15

eg i= 3

then j starts from 0 upto i-1 ie 2

so if ( arr[i] > arr[j] && msis[i] < msis[j] + arr[i])//3>1&&-1<1+3 is true

msis[i] = msis[j] + arr[i];//msis[3]=msis[0]+arr[i] ie. msis[3]=1+3

similarly traverse till j

1 • Reply • Share ›

**yelpz** ➚ tushar • 10 months ago

thanx alot... one more thing ,does msis array have to be initialized though -1 only... not through msis[i]=arr[i] ?

• Reply • Share ›

**Jun** ➚ yelpz • 9 months ago

not with -1...it must be initialised with a[i],,
see my code

• Reply • Share ›

**AlienOnEarth** • a year ago

Brute force:

void maxIncreasingSum(int arr[],int n)

{

int i,j,k,cur_sum,max_sum = INT_MIN, last;

for(i=0;i<n-1;i++) {="" for(j="i+1;j<n;j++)" {="" last="i;" cur_sum="arr[i];" for(k="j;k<n;k++)" {="" if(arr[k]="">> arr[last])

{

cur_sum = cur_sum + arr[k];

last = k;

}

}

---

see more

• Reply • Share ›

**Suvodip Bhattacharya** • a year ago

```
#include<stdio.h>
//#include<climits>
#include<iostream>
#include<climits>
using namespace std;
int max(int a,int b){ return ( a > b) ? a : b; }

int MSIS(int arr[],int start,int current,int finish)
{

if( start > finish ) return 0;

return ( current < arr[start] ) ? max( ( arr[start] + MSIS(arr,start+1,arr[start],finish) ) ,
MSIS(arr,start+1,current,finish) ) : MSIS(arr,start+1,current,finish) ;

}

int main()
{
int arr[]={1, 101, 2, 3, 100, 4, 5};
```

```
int size=sizeof(arr)/sizeof(arr[0]);
cout<<msis(arr,0,int_min,size)<<endl; }="">
```

• Reply • Share ›

**Rohit** • a year ago

IMHO this is a better and easier solution requiring traversing the aarray only once.
https://ideone.com/HmaFuZ

• Reply • Share ›

**Pawan** ➦ Rohit • a year ago

Failing for the following case :
arr = {15, 25, 1, 2, 3, 4, 5, 6, 7, 8, 9};

• Reply • Share ›

**anonymous** • a year ago

I think you are computing some particular solutions again in this solution. Like each time you are adding the previous elements involved in the maximum sum for each solution.
My code for the same,
http://ideone.com/RUgwma

• Reply • Share ›

**Mojo** • a year ago

How do I print the elements involved in the sequence?

• Reply • Share ›

**prashant jha** • a year ago

```
#include<iostream>
using namespace std;
int fun(int arr[],int h[],int low,int high)
{
int m,max=0,k;
if(h[low]!=-1)
return h[low];
if(low==high)
{
h[low]=arr[low];
return arr[low];
}
for(k=low+1;k<=high;k++)
{
if(arr[k]>arr[low])
{
m=fun(arr,h,k,high);
if(m>=max)
```

see more

• Reply • Share ›

**Krishna Sharma** • 2 years ago

```
/*
#include<iostream>

using namespace std;

int calcMaxSumIncreasingSequence(int a[], int);

int main()
{
int a[] = {1, 101, 2, 3, 100, 4, 5};
int length = sizeof(a)/sizeof(a[0]);
cout << calcMaxSumIncreasingSequence(a, length) << endl;

return 0;
}

int calcMaxSumIncreasingSequence(int a[], int length)
{
int max = a[0];
int lastAdded = 0;
for(int i=0; i
```

1 • Reply • Share ›

**Yuvaraj Velmayil** • 2 years ago

Cant we use the same technique of finding the max sum in a array with O(n) complexity ??

```
maxSoFar = a[i];
maxOverall = a[i];

for( int i =1; i<n; i++){="" if(a[i-1]="" <="" a[i]="" )="" maxsofar="a[i];" else="" maxsofar="a[i];" maxoverall="Math.max(maxOverall," maxsofar);="" }="" return="" maxoverall;="" any="" comments="" ??="">
```

• Reply • Share ›

**Asap** • 2 years ago

Can we extend O(nlogn) approach of lis http://www.geeksforgeeks.org/l...
for this questions?

2 • Reply • Share ›

**anon_user** • 2 years ago

I've done this without using an extra array, i.e., space complexity is O(1). Please correct if

I've done this without using an extra array, i.e., space complexity is O(1). Please correct if wrong.

```
#include<stdio.h>
int main()
{
    int i,j;
    int a[]={1,101,2,3,100,4,5};
    int sum=0,maxSum=0;
    for(i=1;i<7;i++)
    {
        sum=0;
        for(j=0;j<i;j++)
        {
            if(a[i]>a[j])
            {
                sum+=a[j];
            }
```

see more

• Reply • Share ›

**Sriharsha g.r.v** ➚ anon_user • 2 years ago

hi....it works but i didnt understand use of this one

//sum+=a[i];
and by ur method can we retrieve the series if asked..i am not sure?

• Reply • Share ›

**ashish** ➚ anon_user • 2 years ago

please check for given input
ary[]={1,101,2,29,3,100,4,5}

• Reply • Share ›

**abhishek08aug** • 2 years ago

Intelligent :D

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

• Reply • Share ›

**Amit** • 2 years ago

```
/* Paste your code here (You may delete these lines if not writing code) */
#include<stdio.h>
int maxSumIS(int a[],int n){
```

```
    int temp[n],i,j,sum,max = 0;
    memset(temp,-1,n);
    for(i=n-1;i>=0;i--){
         sum = 0;
         for(j=0;j<=i;j++){
             if(a[i]>=a[j]){
                 sum = sum + a[j];
             }
         }
         temp[i] = sum;
    }

    for(i =0;i<n;i++){
          if(temp[i]>max){
                        max = temp[i];
```

see more

• Reply • Share ›

**Mrityunjoy Saha** • 2 years ago

** Solution in previous post won't work for input { 1, 11, 2, 3, 15 } and corrected here **
This is much cleaner solution.
Concept:
1. Take an auxiliary array of equal size.
2. At each index compute sum till that point considering only ascending values.

Algorithm:
1. Sum at 0 index is the element value.
2. For subsequent elements compute the sum by adding current element with sum at index whose element
value is smaller and sum is maximum.

```
public class MaxSumAscendingSubArray {
       private void findMaxSum(int[] a) {
               // initialize sum. this array contains sum at each index considering
               // only ascending
               // order values
               int[] sum = new int[a.length];
```

see more

• Reply • Share ›

**Mrityunjoy Saha** • 2 years ago

This is much cleaner solution.
Concept:

Concept:
1. Take an auxiliary array of equal size.
2. At each index compute sum till that point considering only ascending values.

Algorithm:
1. Sum at 0 index is the element value.
2. For subsequent elements compute the sum by adding current element with sum at last index whose element
value is smaller.

```
public class MaxSumAscendingSubArray {
	private void findMaxSum(int[] a) {
		// initialize sum. this array contains sum at each index considering
		// only ascending
		// order values
		int[] sum = new int[a.length];
		int n = a.length - 1;
```

see more

• Reply • Share ›

**amit** • 3 years ago

Hey friends,

Why cant we use stack for finding the max contiguous sum.
- we can insert in stack as we go from left to right
- check if element is less than top of stack
- if the current element is > top of stack , pop it and push current element
- if current element is less than top, push it on stack
- if the current element is < top of stack and after popping stack becomes empty , we dont pop the stack

Does something like above work ?

[sourcecode language="C"]
/* Paste your code here (You may delete these lines if not writing code) */

• Reply • Share ›

**shanky** • 3 years ago

can this problem be solved in O(nlogn) as the lis problem???

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

• Reply • Share ›

**ajay** • 3 years ago

Is it increasing sequence or increasing subsequence?

• Reply • Share ›

**zeus** ajay • 3 years ago

increasing subsequence

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

• Reply • Share ›

**Sourabh mehrotra** • 3 years ago

the code is not working for input[1,11,2,3,15] the output should be 2+3+15=20... But its showing 27 ..... please explain why this is happening....

Thanks in advance

• Reply • Share ›

**kartik** Sourabh mehrotra • 3 years ago

Take a closer look at the problem statement and given examples. 27(1+11+15) is the correct answer for your input array.

• Reply • Share ›

**lohith** • 3 years ago

```
import java.util.HashMap;

public class MaximumSumIncreasingSubSequence {

        public static CalculatedValues cv = new CalculatedValues();

        public static void main(String[] args) {
                int array[] = { 104, 101, 2, 3, 100, 4, 5 };

                IncreasingSubSequenceObject iso = IncreasingSubSequence(array, 0,
                                array.length - 1);
                System.out.println(iso);

        }

        private static IncreasingSubSequenceObject IncreasingSubSequence(
                        int[] array, int low, int high) {
```

see more

• Reply • Share ›

**Bhavesh** • 3 years ago

First read this post
http://www.geeksforgeeks.org/a...

in which LIS is calculate in O(nlogn) and use that approach to obtain MSIS .

Take another array of length n sum[i] which stores the maximum sum that can be obtained for a particular length of subsequence

1. If A[i] is smallest among all end candidates of active lists, we will start new active list of length 1 and sum[0]=max{a[0],sum[0]}.

2. If A[i] is largest among all end candidates of active lists, we will clone the largest active list, and extend it by A[i] and update sum[len] for extended list

3. If A[i] is in between, we will find a list with largest end element that is smaller than A[i]. Clone and extend this list by A[i]. We will discard all other lists of same length as that of this modified list and also update sum[len] .

and in the end search for maximum sum in the sum[] and that is the required ans thus obtained .

1 ^ | v • Reply • Share ›

**brahma** ➚ Bhavesh • 10 months ago

can you check this one.. i implemented same algorithm but the LIS is not correct... it is giving wrong sub sequence
http://ideone.com/pVtz2r

^ | v • Reply • Share ›

**joker** • 3 years ago

just a problem based on this algo.

http://www.spoj.pl/problems/HO...

^ | v • Reply • Share ›

**joker** ➚ joker • 3 years ago

oh sry this problem is about subarrays while algo is about LIS . :-)

^ | v • Reply • Share ›

**Mukul** • 3 years ago

Instead of using the loops as given in the upper code, we can optimize it further as:

```
/*
     for ( i = 1; i < n; i++ )
     {
```

```
            for ( j = i-1; j >= 0; j-- )
            if ( arr[i] > arr[j] && msis[i] < msis[j] + arr[i])
             {
                    msis[i] = msis[j] + arr[i];
                    break;
             }
             printf("%d \n",msis[i]);
      }
*/
```

• Reply • Share ›

**hari** ➚ Mukul • 3 years ago

well this works fine !!

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

• Reply • Share ›

**atul** • 3 years ago

algorithm is correct. but i can see in maxSumIS() function is returning local variable i.e "max" . if you are freeing m/m at the end of the function then i guess returning local variable is not a good practice as it may result in returning some garbage value.

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

• Reply • Share ›

**kartik** ➚ atul • 3 years ago

I think you are getting confused here. Returning pointer to local variable is not good practice. But, returning a local variable is always fine.

• Reply • Share ›

Load more comments

Java Programming Language · 1 hour ago

- Piyush

  What is the purpose of else if (recStack[*i])...

  Detect Cycle in a Directed Graph · 1 hour ago

- Andy Toh

  My compile-time solution, which agrees with the...

  Dynamic Programming | Set 16 (Floyd Warshall Algorithm) · 1 hour ago

- lucy

  because we first fill zero in first col and...

  Dynamic Programming | Set 29 (Longest Common Substring) · 2 hours ago

- lucy

  @GeeksforGeeks i don't n know what is this long...

  Dynamic Programming | Set 28 (Minimum insertions to form a palindrome) · 3 hours ago

- manish

  Because TAN is not a subsequence of RANT. ANT...

  Given two strings, find if first string is a subsequence of second · 3 hours ago