GeeksforGeeks

A computer science portal for geeks

GeeksQuiz

- 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

Find the closest pair from two sorted arrays

Given two sorted arrays and a number x, find the pair whose sum is closest to x and **the pair has an element from each array**.

We are given two arrays ar1[0…m-1] and ar2[0..n-1] and a number x, we need to find the pair ar1[i] + ar2[j] such that absolute value of (ar1[i] + ar2[j] – x) is minimum.

Example:

```
Input:  ar1[] = {1, 4, 5, 7};
        ar2[] = {10, 20, 30, 40};
        x = 32
Output:  1 and 30
```

```
Input:  ar1[] = {1, 4, 5, 7};
        ar2[] = {10, 20, 30, 40};
        x = 50
Output:  7 and 40
```

We strongly recommend to minimize your browser and try this yourself first.

A **Simple Solution** is to run two loops. The outer loop considers every element of first array and inner loop checks for the pair in second array. We keep track of minimum difference between ar1[i] + ar2[j] and x.

We can do it **in O(n) time** using following steps.
1) Merge given two arrays into an auxiliary array of size m+n using merge process of merge sort. While merging keep another boolean array of size m+n to indicate whether the current element in merged array is from ar1[] or ar2[].

2) Consider the merged array and use the linear time algorithm to find the pair with sum closest to x. One extra thing we need to consider only those pairs which have one element from ar1[] and other from ar2[], we use the boolean array for this purpose.

Can we do it in a single pass and O(1) extra space?
The idea is to start from left side of one array and right side of another array, and use the algorithm same as step 2 of above approach. Following is detailed algorithm.

```
1) Initialize a variable diff as infinite (Diff is used to store the
   difference between pair and x).  We need to find the minimum diff.
2) Initialize two index variables l and r in the given sorted array.
       (a) Initialize first to the leftmost index in ar1:  l = 0
       (b) Initialize second  the rightmost index in ar2:  r = n-1
3) Loop while l < m and r >= 0
       (a) If  abs(ar1[l] + ar2[r] - sum) < diff  then
           update diff and result
       (b) Else if(ar1[l] + ar2[r] <  sum )  then l++
       (c) Else r--
4) Print the result.
```

Following is C++ implementation of this approach.

```
// C++ program to find the pair from two sorted arays such
// that the sum of pair is closest to a given number x
#include <iostream>
#include <climits>
#include <cstdlib>
using namespace std;

// ar1[0..m-1] and ar2[0..n-1] are two given sorted arrays
// and x is given number. This function prints the pair  from
// both arrays such that the sum of the pair is closest to x.
void printClosest(int ar1[], int ar2[], int m, int n, int x)
{
    // Initialize the diff between pair sum and x.
    int diff = INT_MAX;

    // res_l and res_r are result indexes from ar1[] and ar2[]
    // respectively
    int res_l, res_r;

    // Start from left side of ar1[] and right side of ar2[]
    int l = 0, r = n-1;
    while (l<m && r>=0)
    {
       // If this pair is closer to x than the previously
       // found closest, then update res_l, res_r and diff
       if (abs(ar1[l] + ar2[r] - x) < diff)
       {
           res_l = l;
           res_r = r;
           diff = abs(ar1[l] + ar2[r] - x);
       }

       // If sum of this pair is more than x, move to smaller
       // side
       if (ar1[l] + ar2[r] > x)
           r--;
       else  // move to the greater side
           l++;
    }

    // Print the result
    cout << "The closest pair is [" << ar1[res_l] << ", "
         << ar2[res_r] << "] \n";
}

// Driver program to test above functions
int main()
{
    int ar1[] = {1, 4, 5, 7};
    int ar2[] = {10, 20, 30, 40};
    int m = sizeof(ar1)/sizeof(ar1[0]);
    int n = sizeof(ar2)/sizeof(ar2[0]);
    int x = 38;
    printClosest(ar1, ar2, m, n, x);
    return 0;
}
```

Output:

```
 The closest pair is [7, 30]
```

This article is contributed by Harsh. 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

Tweet 3 g+1 1

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

21 Comments GeeksforGeeks 1 Login

Recommend Share Sort by Newest

Join the discussion…

David • 14 days ago

I think i miss something... If you run it for [1,2,3,4,5] and [10,11,12,13,14] it outputs "1,10" while it should be "5,10". Am I wrong?

• Reply • Share ›

kanu • a month ago

why cant we use binary search for this question ?

• Reply • Share ›

Anon • 3 months ago

Shouldn't pseudocode 3b) be another "if" statement?. Change from 'else if' to 'if'

1 • Reply • Share ›

blue • 4 months ago

How can we use the above approach when it isn't mentioned that all the elements of the second array are larger than the last element of the first element ? It is necessary to merge the two arrays and then use the above approach.

• Reply • Share ›

Guest • 5 months ago

These solutions are really far too complex. Especially if the question is giving you two SORTED arrays. The solution isn't to see if you can code at all in this case. For two sorted arrays the answer is simply

System.out.println(dat1[dat1.length] + " , " + dat2[dat2.length]);

For two unsorted arrays you can solve this in O(n) by simply looping through the arrays and returning the Max of each one. You can do it in O(n) minus a little bit if you use one loop to check both arrays and a second loop to finish up the second array should it be longer check both arrays and a second loop to finish up the second array should it be longer.

Really questions like this aren't testing your ability to code but your ability to analyse requirements. My .02.

• Reply • Share ›

KKK Destroyer ➦ Guest • a month ago

What the f?? max of each one? How does that return the closest pair? No wonder you can't get a job outside wipro or infy.

• Reply • Share ›

palindname • 5 months ago

just in case the value of x turns out to be EQUAL to the sum of two elements..

```
while (l<m &&="" r="">=0)
{
//Verifying if the sum is equal to x
if (abs(ar1[l] + ar2[r]) == x)
{
res_l=l;
res_r=r;
exit(0);
}

// If this pair is closer to x than the previously
// found closest, then update res_l, res_r and diff
if (abs(ar1[l] + ar2[r] - x) < diff)
{
res_l = l;
res_r = r;
```

see more

• Reply • Share ›

Ankur Dandecha • 6 months ago

In jaVa,

```
class test {
public static void main(String[] args) {

int[] I = {1, 4, 5, 7};
int[] J = {10, 20, 30, 40};
int x = 50;

int i_ = 0;
int j_ = 0;
int j_ = 0;
int min_diff = I[0] + J[0] - x;
if (min_diff < 0) min_diff *= -1;

for (int i = 0; i < I.length; ++i) {
for (int j = 0; j < J.length; ++j) {
int diff = I[i] + J[j] - x;
if (diff < 0) diff *= -1;
if (diff < min_diff) {
```

see more

• Reply • Share ›

vicharak • 6 months ago

i am bit confused here... any one plz help me.....
if we change the loop condition to (l<r) from="" (l<m&&r="">=0) ...... will it make a difference....

• Reply • Share ›

Anurag Singh ➦ vicharak • 6 months ago

It will make a difference. Try changing it in above code and run. You will get a different result [1, 40] instead of [7, 30].
Here l and r are the pointer on two different arrays.
If we put l < r condition, then we will not process last few elements in 1st array and 1st few elements in 2nd array because we are coming out of loop as soon as l and r cross each other.

• Reply • Share ›

Anonymous • 6 months ago

Update

"if abs(ar1[l] + ar2[r] - sum) > diff"
to
"If abs(ar1[l] + ar2[r] - sum) < diff"
in pseudocode.

• Reply • Share ›

GeeksforGeeks Mod ➦ Anonymous • 6 months ago

Thanks for pointing this out. We have updated the same.

1 • Reply • Share ›

Sameer ➦ GeeksforGeeks • 5 months ago

We can reduce complexity from O(n) to O(1) in situations when x doesn't has either a lower or upper limit.

```
if ( ( a1[0] + a2[0] ) > x) //No lower limit
```

```
{
res_l = 0; res_r = 0;
}
else if ( ( a1[m-1] + a2[n-1] ) < x) //No upper limit

{
res_l=m-1; res_r=n-1;
}
```

• Reply • Share ›

vicharak ➦ Anonymous • 6 months ago

good observation

• Reply • Share ›

DQ • 6 months ago

Correct me if I'm wrong, but why can't you just do a "fake" merge (i.e. no need to create the actual merged array) like the following?

1. Two variables, n1 and n2
2. Whenever you "fake merge" a number in an array, compare that with n1 or n2 (depending on which array) and update the other one.
3. If it's smaller than the smallest difference so far, update the smallest.

Simple one pass O(1) space, and very straightforward.

• Reply • Share ›

Ekta Goel ➦ DQ • 6 months ago

For this to work, we need to keep one in ascending and the other array is descending order. Then only, this merge procedure will work!! Otherwise u'll not be able to decide which to increment i(used for first array) or j(for second array).

• Reply • Share ›

Lakshay Thareja ➦ Ekta Goel • 16 days ago

they are sorted so why can't we keep one at the left of 1st array and second at the right of the other array.

• Reply • Share ›

Ekta Goel ➦ Lakshay Thareja • 16 days ago

That is what we re doing, one pointer points to the leftmost(starting0 of first array and second to the rightmost(ending) of the other array.

• Reply • Share ›

Ashish khandelwal • 6 months ago

In the Algorithm there is a mistake in 3 (a) It should be

If abs(ar1[l] + ar2[r] - sum) < diff then if abs(ar1[l] + ar2[r] - sum) < diff then

2 ^ | ˅ • Reply • Share ›

samar • 6 months ago

why we take two different indexes
like left of arr1 and right of arr2

^ | ˅ • Reply • Share ›

pango89 • 6 months ago

Good One.. It is similar to the problem where we have only one sorted array.

^ | ˅ • Reply • Share ›

Subscribe   Add Disqus to your site   Privacy

-
-
-
-
  - Interview Experiences
  - Advanced Data Structures
  - Dynamic Programming
  - Greedy Algorithms
  - Backtracking
  - Pattern Searching
  - Divide & Conquer
  - Mathematical Algorithms
  - Recursion
  - 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
- Follow @GeeksforGeeks

- ## Recent Comments

  - lt_k

    i need help for coding this function in java...

    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) · 2 hours ago

  - manish

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

    Given two strings, find if first string is a subsequence of second · 2 hours ago