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

Merge k sorted arrays | Set 1

Given k sorted arrays of size n each, merge them and print the sorted output.

Example:

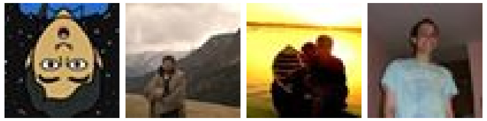
```
Input:
k = 3, n =  4
arr[][] = { {1, 3, 5, 7},
            {2, 4, 6, 8},
            {0, 9, 10, 11}} ;

Output: 0 1 2 3 4 5 6 7 8 9 10 11
```

A simple solution is to create an output array of size n*k and one by one copy all arrays to it. Finally, sort the output array using any O(nLogn) sorting algorithm. This approach takes O(nkLognk) time.

We can merge arrays in O(nk*Logk) time using Mean Heap. Following is detailed algorithm.

1. Create an output array of size n*k.
2. Create a min heap of size k and insert 1st element in all the arrays into a the heap
3. Repeat following steps n*k times.
    a) Get minimum element from heap (minimum is always at root) and store it in output array.
    b) Replace heap root with next element from the array from which the element is extracted. If the array doesn't have any more elements, then replace root with infinite. After replacing the root, heapify the tree.

Following is C++ implementation of the above algorithm.



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

```
// C++ program to merge k sorted arrays of size n each.
#include<iostream>
#include<limits.h>
using namespace std;

#define n 4

// A min heap node
struct MinHeapNode
{
    int element; // The element to be stored
    int i; // index of the array from which the element is taken
    int j; // index of the next element to be picked from array
};

// Prototype of a utility function to swap two min heap nodes
void swap(MinHeapNode *x, MinHeapNode *y);

// A class for Min Heap
class MinHeap
{
    MinHeapNode *harr; // pointer to array of elements in heap
    int heap_size; // size of min heap
public:
    // Constructor: creates a min heap of given size
    MinHeap(MinHeapNode a[], int size);

    // to heapify a subtree with root at given index
    void MinHeapify(int );

    // to get index of left child of node at index i
    int left(int i) { return (2*i + 1); }

    // to get index of right child of node at index i
    int right(int i) { return (2*i + 2); }

    // to get the root
    MinHeapNode getMin() { return harr[0]; }

    // to replace root with new node x and heapify() new root
    void replaceMin(MinHeapNode x) { harr[0] = x;  MinHeapify(0); }
};

// This function takes an array of arrays as an argument and
// All arrays are assumed to be sorted. It merges them together
// and prints the final sorted output.
```

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

```
int *mergeKArrays(int arr[][n], int k)
{
    int *output = new int[n*k];  // To store output array

    // Create a min heap with k heap nodes.  Every heap node
    // has first element of an array
    MinHeapNode *harr = new MinHeapNode[k];
    for (int i = 0; i < k; i++)
    {
        harr[i].element = arr[i][0]; // Store the first element
        harr[i].i = i;  // index of array
        harr[i].j = 1;  // Index of next element to be stored from arr
    }
    MinHeap hp(harr, k); // Create the heap

    // Now one by one get the minimum element from min
    // heap and replace it with next element of its array
    for (int count = 0; count < n*k; count++)
    {
        // Get the minimum element and store it in output
        MinHeapNode root = hp.getMin();
        output[count] = root.element;

        // Find the next elelement that will replace current
        // root of heap. The next element belongs to same
        // array as the current root.
        if (root.j < n)
        {
            root.element = arr[root.i][root.j];
            root.j += 1;
        }
        // If root was the last element of its array
        else root.element =  INT_MAX; //INT_MAX is for infinite

        // Replace root with next element of array
        hp.replaceMin(root);
    }

    return output;
}
```

```
// FOLLOWING ARE IMPLEMENTATIONS OF STANDARD MIN HEAP METHODS
// FROM CORMEN BOOK
// Constructor: Builds a heap from a given array a[] of given size
MinHeap::MinHeap(MinHeapNode a[], int size)
{
    heap_size = size;
```

```
    harr = a;  // store address of array
    int i = (heap_size - 1)/2;
    while (i >= 0)
    {
        MinHeapify(i);
        i--;
    }
}

// A recursive method to heapify a subtree with root at given index
// This method assumes that the subtrees are already heapified
void MinHeap::MinHeapify(int i)
{
    int l = left(i);
    int r = right(i);
    int smallest = i;
    if (l < heap_size && harr[l].element < harr[i].element)
        smallest = l;
    if (r < heap_size && harr[r].element < harr[smallest].element)
        smallest = r;
    if (smallest != i)
    {
        swap(&harr[i], &harr[smallest]);
        MinHeapify(smallest);
    }
}

// A utility function to swap two elements
void swap(MinHeapNode *x, MinHeapNode *y)
{
    MinHeapNode temp = *x;  *x = *y;  *y = temp;
}

// A utility function to print array elements
void printArray(int arr[], int size)
{
   for (int i=0; i < size; i++)
       cout << arr[i] << " ";
}

// Driver program to test above functions
int main()
{
    // Change n at the top to change number of elements
    // in an array
    int arr[][n] =  {{2, 6, 12, 34},
                     {1, 9, 20, 1000},
```

705 Subscribe

Recent Comments

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

Write a C program to Delete a Tree. · 4 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 8 minutes ago

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

Root to leaf path sum equal to a given number · 32 minutes ago

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

Count trailing zeroes in factorial of a number · 34 minutes ago

newCoder3006 If the array contains negative numbers also. We...

Find subarray with given sum · 59 minutes ago

newCoder3006 Code without using while loop. We can do it...

Find subarray with given sum · 1 hour ago

```
                     {23, 34, 90, 2000}};
    int k = sizeof(arr)/sizeof(arr[0]);

    int *output = mergeKArrays(arr, k);

    cout << "Merged array is " << endl;
    printArray(output, n*k);

    return 0;
}
```

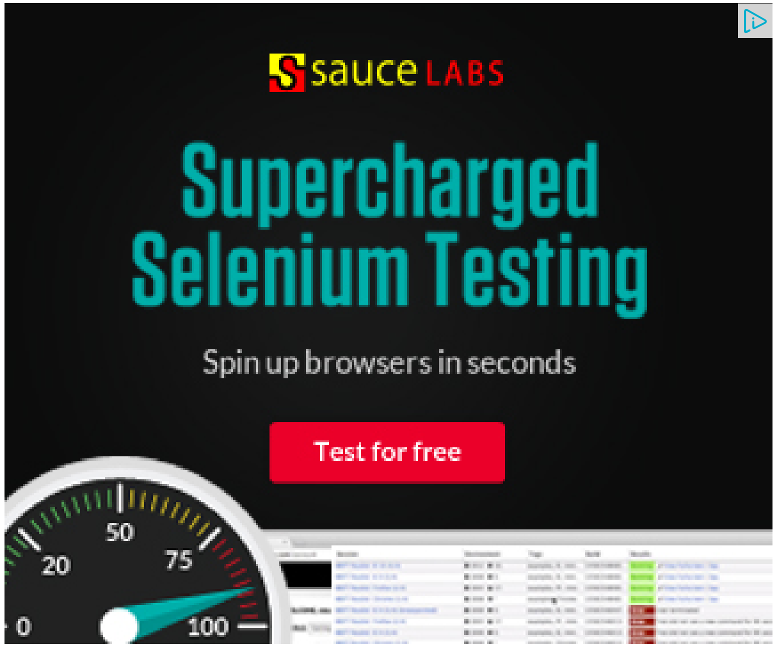
Output:

```
Merged array is
1 2 6 9 12 20 23 34 34 90 1000 2000
```

Time Complexity: The main step is 3rd step, the loop runs n*k times. In every iteration of loop, we call heapify which takes O(Logk) time. Therefore, the time complexity is O(nk Logk).

There are other interesting methods to merge k sorted arrays in O(nkLogk), we will sonn be discussing them as separate posts.

Thanks to vignesh for suggesting this problem and initial solution. 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

37 Tweet 3 1

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

20 Comments GeeksforGeeks

Sort by Newest ▾

Join the discussion…

Ritesh Mahato • 4 days ago

Admin pl correct this : "We can merge arrays in O(nk*Logk) time using Mean I

It should be Min heap.

⌃ | ⌄ • Reply • Share ›

Nikunj Banka • 2 months ago

An alternative approach can be to bottom up merge sort the array considering then 8k.... The complexity would again be O(NKlogK) as we have to merge the have to look at all the NK numbers.

You can see the working code here. http://ideone.com/fuxBSK

1 ⌃ | ⌄ • Reply • Share ›

Uma Trika • 4 months ago

We can merge arr1[] and arr2[] in to temp_arr[] And then merge temp_arr[] an The time complexity will be O(n). But it needs n*2k temporary array to store th

⌃ | ⌄ • Reply • Share ›

Ankit Chaudhary • 4 months ago

Another nk*log(k) solution: Modified mergeSort.

create two global array of size n*k each, one is our final sorted array(arr[n*k]), temporary array (tmp[n*k])

step 1 :Copy 2D array to arr[n*k]
step2 : call mergeK(2D array, 0,k-1,n)

prototype : void mergeK(int a[][MAX],int low,int high,int n)

Recursive Algo:

```
void mergeK(int a[][MAX],int low,int high,int n){
1: if(low>=high)
return;
2 : int mid=(low+high)/2;
3: mergeK(a,low,mid,n);
4 : mergeK(a,mid+1,high,n);
// now merge two sorted array a[low][n] to a[high][n]
5 : merge(arr, low*n, (mid+1)*n ,(mid+1)*n, (high+1)*n);
```

see more

⌃ | ⌄ • Reply • Share ›

Abhishek Kumar ➦ Ankit Chaudhary • 4 months ago

yar can u tell me how nk*logk complexity is for your method ??

⌃ | ⌄ • Reply • Share ›

Ankit Chaudhary ➦ Abhishek Kumar • 4 months ago

height of recursion tree is log(k).
At every level total number of elements are n*k (for merging).
time= n*k*height = nklog(k).

⌃ | ⌄ • Reply • Share ›

Vikram Ojha • 6 months ago

we have taken here 2D array having k rows....bt I think we need to take K differ
i suppose

⌃ | ⌄ • Reply • Share ›

Jash Sayani • 6 months ago

Interesting idea. But you are assuming that second element of any array is lar
if you have {3,4,6,7} and {1,2,5,8}, then you put 3 and 1 in heap and extractMin
than 3, which you already put in the result array. I think using the merge step o
would be a good bet.

⌃ | ⌄ • Reply • Share ›

Yukang ➦ Jash Sayani • 6 months ago

No, we don't have this assumption.

put 3 and 1 in heap do NOT mean we put it in final output array,
you extract 1 from heap, and put it in final array, at the same time put 2
and the next extract value is 2, this is right.

2 ⌃ | ⌄ • Reply • Share ›

James Fraser • 7 months ago

Exception in thread "main" java.lang.NullPointerException
harr[i].element = arr[i][0];

Java gets a null pointer exception for MinHeapNode. Could anyone please prov

⌃ | ⌄ • Reply • Share ›

Deepak Shrivastava • 7 months ago

i have written a c implementation of the same

```
#include<iostream>

using namespace std;

int left(int i) {

return 2*i + 1;

}

int right(int i) {

return 2*i+2;
```

```
void swap (int* a, int* b) {

int tmp = (*a);
```

see more

1 ⌃ | ⌄ • Reply • Share ›

Guest • 9 months ago

```
main()

{

    int *a,*b,*c,*d,m,n,l,i,*merge(int*,int*,int*,int,int,int);

    printf("Enter the size of 1st array\n");

    scanf("%d",&m);

    a=(int*)malloc(sizeof(int)*m);

    printf("Enter the elements of 1st array\n");

    for(i=0;i<m;i++) scanf("%d",&a[i]);="" printf("enter="" the="" si
```

see more

⌃ | ⌄ • Reply • Share ›

Guest • 9 months ago

main()

```
{

int *a,*b,*c,*d,m,n,l,i,*merge(int*,int*,int*,int,int,int);

printf("Enter the size of 1st array\n");

scanf("%d",&m);

a=(int*)malloc(sizeof(int)*m);

printf("Enter the elements of 1st array\n");

for(i=0;i<m;i++) scanf("%d",&a[i]);="" printf("enter="" the="" size="" of="" 2nd=
b="(int*)malloc(sizeof(int)*n);" printf("enter="" the="" elements="" of="" 2nd=""
scanf("%d",&b[i]);="" printf("enter="" the="" size="" of="" 3rd="" array\n");="" sc
c="(int*)malloc(sizeof(int)*l);" printf("enter="" the="" elements="" of="" 3rd="" a
scanf("%d",&c[i]);="" d="(int*)malloc(sizeof(int)*(m+n+l));" d="merge(a,b,c,m,
for(i="0;i&lt;(m+n+l);i++)" printf("%d\n",d[i]);="" return="" 0;="" }="" int*="" merg
```

see more

^ | ⌄ • Reply • Share ›

shivam • 10 months ago

Why can't we do it like the 'merge' function we make in mergesort ?

As far as I know every time merge function is called it merges two sorted array
(where m and n are sizes of the two sorted arrays) ?

Please correct me if I am wrong.

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

Guest ➜ shivam • 7 months ago

Simple merging would take O(n*k^2) time.

^ | ⌄ • Reply • Share ›

Zebadiah ➜ shivam • 10 months ago

"Merge algorithms generally run in time proportional to the sum of the l
that operate on large numbers of lists at once will multiply the sum of tl
figure out which of the pointers points to the lowest item" http://en.wikip

This would make a standard merge O(n*m) where n is the length of all
The point of using the heap is to reduce the cost of finding the min eler
the overall order O(n*log m)

^ | ⌄ • Reply • Share ›

vishal ➜ Zebadiah • 9 months ago

Instead of using heap we can do pairwise merging of arrays an
arrays and so on..
since each array has n elements and there are k arrays total nc
After final merge we will have one resultant array of size n*k.
Total no of levels in recursion tree of this operation(merging arr
ceiling of logk)
So total cost of merging k arrays will be
: "O(nklogk)"
(since there are logk levels and merging at each level will cost t
are nk elements at each level and we are merging arrays pairw

```
/* Paste your code here (You may delete these lines if r
```

^ | ⌄ • Reply • Share ›

Zebadiah · 10 months ago

Here's a c# solution for the more trivial way of doing this. It's pretty easy to see
replacing the inner loop that finds the current min with a much faster minheap.
arrays of different lengths for no apparent reason.

```
[sourcecode language="C#"]
// Generate a random number of non-decreasing lists of random size
Random r = new Random();
int numArrays = r.Next()%10;
int[][] data = new int[numArrays][];
for(int i = 0; i < numArrays; i++)
{
data[i] = new int[r.Next()%10];
int lastValue = -100;
for(int j = 0; j < data[i].Length; j++)
{
lastValue += r.Next()%100;
data[i][j] = lastValue;
}
```

see more

^ | ⌄ · Reply · Share ›

Ankita · 10 months ago

```
Having some problem with code. Not giving the correct output for the
int arr[][n] =  {{200, 60, 120, 34},
                     {10, 19, 20, 100},
                     {23, 34, 90, 20}};

output :- Merged array is
10 19 20 23 34 90 20 100 200 60 120 34
```

Reply • Share ›

GeeksforGeeks → Ankita • 10 months ago

@Ankita: The input array doesn't seem to be valid. {200, 60, 120, 34} is

1 • Reply • Share ›