Arrange given numbers to form the biggest number

Given an array of numbers, arrange them in a way that yields the largest value. For example, if the given numbers are {54, 546, 548, 60}, the arrangement 6054854654 gives the largest value. And if the given numbers are {1, 34, 3, 98, 9, 76, 45, 4}, then the arrangement 998764543431 gives the largest value.

A simple solution that comes to our mind is to sort all numbers in descending order, but simply sorting doesn't work. For example, 548 is greater than 60, but in output 60 comes before 548. As a second example, 98 is greater than 9, but 9 comes before 98 in output.

So how do we go about it? The idea is to use any comparison based sorting algorithm. In the used sorting algorithm, instead of using the default comparison, write a comparison function myCompare() and use it to sort numbers. Given two numbers X and Y, how should myCompare() decide which number to put first – we compare two numbers XY (Y appended at the end of X) and YX (X appended at the end of Y). If XY is larger, then X should come before Y in output, else Y should come before. For example, let X and Y be 542 and 60. To compare X and Y, we compare 54260 and 60542. Since 60542 is greater than 54260, we put Y first.

Following is C++ implementation of the above approach. To keep the code simple, numbers are considered as strings, and vector is used instead of normal array.

```
// Given an array of numbers, program to arrange the numbers to form t
// largest number
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
using namespace std;
```


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

```
// A comparison function which is used by sort() in printLargest()
int myCompare(string X, string Y)
{
    // first append Y at the end of X
    string XY = X.append(Y);

    // then append X at the end of Y
    string YX = Y.append(X);

    // Now see which of the two formed numbers is greater
    return XY.compare(YX) > 0 ? 1: 0;
}

// The main function that prints the arrangement with the largest valu
// The function accepts a vector of strings
void printLargest(vector<string> arr)
{
    // Sort the numbers using library sort funtion. The function uses
    // our comparison function myCompare() to compare two strings.
    // See http://www.cplusplus.com/reference/algorithm/sort/ for deta
    sort(arr.begin(), arr.end(), myCompare);

    for (int i=0; i < arr.size() ; i++ )
        cout << arr[i];
}

// driverr program to test above functions
int main()
{
    vector<string> arr;

    //output should be 6054854654
    arr.push_back("54");
    arr.push_back("546");
    arr.push_back("548");
    arr.push_back("60");
    printLargest(arr);

    // output should be 777776
    /*arr.push_back("7");
    arr.push_back("776");
    arr.push_back("7");
    arr.push_back("7");*/

    //output should be 998764543431
    /*arr.push_back("1");
    arr.push_back("34");
```

Geometric Algorithms

Popular Posts

```
    arr.push_back("3");
    arr.push_back("98");
    arr.push_back("9");
    arr.push_back("76");
    arr.push_back("45");
    arr.push_back("4");
    */

    return 0;
}
```

Output:

```
6054854654
```

This article is compiled by **Ravi Chandra Enaganti**. 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

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

41 Comments GeeksforGeeks

Sort by Newest

Deepak Shrivastava • 7 months ago

my solution in java

```
import java.util.Arrays;

import java.util.Comparator;

public class Sol {

public static void main(String[] args) {

Integer a[] = { 1, 34, 3, 98, 9, 76, 45, 4 };

Arrays.sort(a, new Comparator<integer>() {
```


Recent Comments

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

Write a C program to Delete a Tree. · 11 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 14 minutes ago

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

Root to leaf path sum equal to a given number · 39 minutes ago

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

Count trailing zeroes in factorial of a number · 41 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

```
@Override

public int compare(Integer o1, Integer o2) {

return (o2 + "" + o1).compareTo(o1 + "" + o2);

}
```

see more

• Reply • Share ›

Sriharsha g.r.v ➦ Deepak Shrivastava • 5 months ago

may i know why my code is getting clumgy while copying here and oth

thanq

• Reply • Share ›

manisha • 7 months ago

another solution could be:

1.find the max number of digits a number has in an array

2.make every other number in the array to have the same number of digits by it be stored in an array brr

3.sort the new array now,here brr

4.print the numbers in the original array according to the positions of the new a

e.g:

60,548,546,54

brr={600,548,546,544}

largest number is now:6054854654

5 ^ | v • Reply • Share ›

Sriharsha g.r.v ➚ manisha • 5 months ago

thanq and here is code

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

int len[4];
int maxlen;
int temp[4];
int main()
{
int i;
int arr[]={54,546,548,60};

maxlen=getmaxdigits(arr);

appendall(arr);
printf("maxlen is %d \n",maxlen );
for(i=0;i<4;i++)
printf("%d ",temp[i] );
```

see more

^ | v • Reply • Share ›

vamshi ➚ Sriharsha g.r.v • 5 months ago

great...

^ | v • Reply • Share ›

Guest ➚ manisha • 5 months ago

thanq and here is the code

```
#include<stdio.h>
#include<stdlib.h>
int a[20][20],reach[20],n;
int len[4];
int maxlen;
int temp[4];
int main()
{
int i;
int arr[]={54,546,548,60};

maxlen=getmaxdigits(arr);

appendall(arr);
printf("maxlen is %d \n",maxlen );
for(i=0;i<4;i++)
printf("%d ",temp[i] );
```

see more

• Reply • Share ›

Aravindan B manisha • 6 months ago

It wont work for 1, 2, 10, 3, 4
your ans : 101234
actual ans may be like : 432110

• Reply • Share ›

Divanshu Aravindan B • 5 months ago

Aravindan .. Sorry his solution will work for thsi test case as we
a = {1,2,10,3,4}
ar = {11,22,10,33,44}
Sort (Ar) = {44,33,22,11,10)

so number will be 432110

2 ^ | v • Reply • Share ›

Suryabhan Singh • 8 months ago

compare function for numbers

```
bool compare(int a,int b)
{
    int count=0;
    int z=a;
    int x=b;
    while(z)
    {
        z/=10;
        count++;
    }
    z=(b*pow(10,count))+a;
    count=0;
    while(x)
    {
        x/=10;
        count++;
    }
    x=(a*pow(10,count))+b;
    return(x>=z);

}
```

^ | v • Reply • Share ›

Guest • 8 months ago

compare function for numbers-

```
bool compare(int a,int b)
{
    int count=0;
    int z=a;
    int x=b;
    while(z)
    {
        z/=10;
        count++;
    }
    z=(b*pow(10,count))+a;
    count=0;
    while(x)
    {
        x/=10;
        count++;
    }
    x=(a*pow(10,count))+b;
    //cout<<x<<" "<<z<<endl;="" return(x="">=z);

}
```

• Reply • Share ›

sesame • 8 months ago

Solution in C:

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

```
#include<math.h>

int cmpfunc(const void *a, const void *b)
{
int x= *(int*)a;
int y= *(int*)b;

int digitsX=0, digitsY=0;
int xy, yx;

while(x>0)
{
x=x/10;
++digitsX;
}
```

see more

Reply • Share ›

Akshay Trivedi • 9 months ago

this can be done using radix sort-sort numbers in msd(max significant digit) a
sorting algorithm.

1 Reply • Share ›

Tuhin Chakrabarty • 10 months ago

matrix exponentiation is same as recursive formulation for power of a number
where M is a matrix . try to check out the logn version for fibonacci numbers .

Reply • Share ›

Soumya Sengupta • a year ago

here's a simple solution.....
we scan d array and pick out the elemnts with greatest dgreatest digit on the l
so if we take the array [1, 34, 3, 98, 9, 76, 45, 4]

the elements to be picked will be,9 and 98
among 9 and 98 the elemnt to b placed in d greates position can be decided b
as given in explanation and found out.....
den the largest and secnd largst positioned elemnts can be placed on the 1st
dem wd d elemnts in dose 1st and secnd positions....

This process will again continue from start+2'th position as the first 2 positions
elemnts.....

this process continues until the array is exhausted....
:)

Reply • Share ›

vikasnitt ➦ Soumya Sengupta • 9 months ago

Soumya you are doing the same as the algo given above.

1 Reply • Share ›

Soumya Sengupta ➦ Soumya Sengupta • a year ago

sorry...
a lil typo missd....after the swapping of 9 and 98 wuth d 0'th and 1'st pc
position for the process to start by 2(start1=start+2) because we have
elemnts(9 and 98) out f d n elements in d aray....
in the next iteratn if there is only 1 elemt with the largest left didgit value
position f d array nd start the process from Start1+1(start +3).....
decided position of elemnts we increment.

Reply • Share ›

booyakasha • a year ago

In C. Compiles without any errors on GCC 4.4.3-4.
Compiler options used: -Wall.

```
#define CH_MAX 10
#include <stdio.h>
```

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

void arrangeForLargest(int[], int);
int funkyCompare(const void *, const void *);

int main(int argc, char *argv[]) {
        int r[] = {54, 546, 548, 60};   /* replace this with an array
        arrangeForLargest(r, sizeof(r) / sizeof(r[0]));
        return 0;
}

int funkyCompare(const void *x, const void *y) {
```

see more

• Reply • Share ›

Dnyaneshwar • a year ago

```
 /* //This program
#include<stdio.h>
main()
{
  int n=5,a[n],i,b[n],max=0,val,pos=0,k=0;
  printf("Enter the five Number ");

  for(i=0;i<n;i++)
    scanf(" %d" ,&a[i]);

  for(i=0;i<n;i++)
    {
      b[i]=length(a[i]);
```

```
    }
     max=maxarray(b,n);
     for(i=0;i<n;i++)
     {
```

see more

Reply • Share ›

anonymous • a year ago

```
 #include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<string.h>
void swap ( int* a, int* b )
{
    int t = *a;
    *a = *b;
    *b = t;
}
int pivote(int a[],int start,int end){
    int p = a[end];
    int i=start,j=start-1;
    char b[5];
    char pi[5];
    char str1[6];
    char str2[6];
    memset (pi,'&#092&#048',sizeof(b));
```

see more

Reply • Share ›

Niks • a year ago

```
 // Given an array of numbers, program to arrange the numbers to form
// largest number
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
using namespace std;

vector<string> store;
int len = 0;
string res = "";

void printLargest(int mask,string temp)
{
        if(mask == 0)
        {
                res = max(res,temp);
                return;
```

see more

• Reply • Share ›

Rahul • a year ago

wouldn't a variation of radix sort work? By keeping in mind the most significant numbered place.

1 • Reply • Share ›

arunaami • a year ago

haha

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

skeptic • a year ago

```
[sourcecode language="java"]
import java.util.*;
import java.io.*;

public class intut{

public static void main(String []args)
{
String A[]={"43","44","12","324","90","9","88","89"};
String B;
int i=0,j;
//find the largest number formed by the numbers in the array
Arrays.sort(A);
while(i < A.length){
System.out.println(A[i]);
++i;
}
j = A.length;
```

see more

AJAY MITTAL • a year ago

I got a simple solution jst use itoa conversion for all no.'s and store all the strin descending lexicographical order and jst print all the string from the starting of

anonymous ➦ AJAY MITTAL • 11 months ago

@AJAY sorry but i think we can't choose digits from a number we have

manoj ➚ AJAY MITTAL • a year ago

thats simple and effective

Binayak • a year ago

```
void sortWithSpecificComparision(int *arr,int n)
{
int i,j,temp,t1,t2,t3,t4,c1=0,c2=0,k;

for(i=0;i <= (n-1) ;i++)
{
for(j=i+1;j 0)
{
t1 = t1/10;
c1++;
}
while(t2 > 0)
{
t2 = t2/10;
c2++;
}
t3 = arr[i] * pow(10,c2) + arr[j];
t4 = arr[j] * pow(10,c1) + arr[i];
if( t3 < t4)
{
temp = arr[i];
arr[i]=arr[j];
arr[j] = temp;
}
}
```

}
}

  • Reply • Share ›

spiderman • a year ago

The given solution is cool and simple solution with sorting!
Keep it up!

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

  • Reply • Share ›

Ganesh P • a year ago

Hi,
You can find java code here:

[sourcecode language="JAVA"]
import java.util.Arrays;
import java.util.Comparator;

/**
* Given an array of numbers, arrange them in a way that yields the largest
* value. For example, if the given numbers are {54, 546, 548, 60}, the
* arrangement 6054854654 gives the largest value.
*
* @author GAPIITD
*
*/
public class ArrangeGivenNumbersToFormTheBiggestNumber {

/**
* @param args

• Reply • Share ›

ctrl ➚ Ganesh P • a year ago

[sourcecode language="java"]
/* Paste your code here (You may delete these lines if not writing code
return Integer.valueOf(st1) - Integer.valueOf(st2);

if condition is not required.

one more way to do is
[sourcecode language="java"]
public int compare(Object arg1, Object arg2) {
Integer i1 = (Integer) arg1;
Integer i2 = (Integer) arg2;
return (i1.intValue()+"").compareTo(i2.intValue()+"");
}

• Reply • Share ›

Sumit • a year ago

Wouldn't it be sufficient to get each digit in the array and sort them in ascendir
greatest number.

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

• Reply • Share ›

Sumit ➚ Sumit • a year ago

Sorry for the last comment, it won't work.

• Reply • Share ›

Jitendra Khushwaha • a year ago

and one more question to admin of this website.:
why don&#039t you post python code which I have written for this problem.
is there any rule that you can post only in c/c++ language.

Jitendra Khushwaha • a year ago

I want to know "matrix exponent" algorithm solving (recurrsive algorithm efficie
any weblink or it would be great if someone wrtie a n article on this.

ss • a year ago

You only need to get the greatest first digits first and when there are two or mc
which has to be place before the other

Himmat Singh Rathore • a year ago

intresting..:p

jayanth • a year ago

the compare function could be modified as

```
int myCompare(string X, string Y)
{
return (X[0]-'0') > (Y[0]-'0') ? 1 : 0;
}
```

because ultimately wt ur code does is just comparing the first digit of the two r
the greatest first digit.....

jayanth ➦ jayanth • a year ago

sorry , the above case fails when both the numbers have the same firs

sorry...the above case fails when both the numbers have the same firs

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

• Reply • Share ›

sagar2693 • a year ago

sorry the above code wasn't correctly uploaded....
plz let me know if the code below falters somewhere

```
 #include<map>
#include<iostream>
using namespace std;
int main(){
int n,i;
cin>>n;
map<string,int> mp;
string num;
for(i=0;i<n;i++){
cin>>num;
mp[num]++;
}
map<string,int> ::reverse_iterator it;
for(it=mp.rbegin();it!=mp.rend();it++){
if(it->second!=1){
while(it->second--)
cout<<it->first;
}
else cout<<it->first;
}
return 0;
}
```

• Reply • Share ›

mopurizwarriors • a year ago

Very good problem and a simple neat solution

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

• Reply • Share ›

sagar2693 • a year ago

```
#include<map>
#include<iostream>
using namespace std;
int main(){
int t,n,i;
cin>>t;
while(t--){
cin>>n;
map<string,int> mp;
string s;
for(i=0;i<n;i++){ cin=""">>s;
mp[s]++;
}
map<string,int> ::reverse_iterator it;
for(it=mp.rbegin();it!=mp.rend();it++){
if(it->second!=1){
while(it->second--)
cout<<it->first;
}
else cout<<it->first;
}
cout<<endl; }="" return="" 0;="" }="">
```