Q. Sort an array in ascending order by the no. of factors. If the no. of factors are same then sort by value.

A : { 9, 3, 10, 6, 4 }

# of factors => 3 2 4 4 3

A : { 3, 4, 9, 6, 10 }

Comparator function :-

* In any sorting algo, at a time we only need to compare (2) elements.

* No. of arguments = 2

* Based on the parameters & rules it should tell us which arg. should come first.

```
bool comp(int a, int b){
    y1 = countfactors(a);
    y2 = countfactors(b);
    if(y1 < y2) return true;
    if(y1 > y2) return false;
    if(a <= b) return true;
    return false;
}
```

a — b (T)
b — a (F)

C++

```
sort(v.begin(), v.end(), comp);
sort(a, a+n, comp);
```

a → array.
n → size

Java

```
Arrays.sort(Arr, new Comparator<Integer>(){
    public int comp(int a, int b){
        // logic
    }
});
```

Q: Largest Number - Amazon | MakeMyTrip | PayTm | Zoho | MS!

⇒ Given an Array of size N of non negative integers, Arrange them s.t they form the largest no.

* Result may be very large so return string instead of an integers.

A : [3, 30, 34, 5, 9]

9 5 34 3 30 ✓

$1 \le N \le 10^5$

$0 \le A[i] \le 2 \times 10^9$

Logic :-

Ex

"a" "b"

{"30", "5"}

"305" < "530"

"a" + "b" = ab

"b" + "a" = ba

if (ab > ba) : (a) comes before (b)

else : (b) comes before (a).

```
bool comp ( string a , string b) {
    if( a+b > b+a ){
        return true;
    }
    return false;
}

string largestNumber ( int A[] , int N) {
    string B[N];
    for( i = 0; i< N; i++ ) {
        B[i] = to_string( A[i]);
    }
    sort ( B , B+N , comp );
    string ans = ""
    for( i = 0; i< N; i++ )
        ans += B[i];
    return ans;
}
```

Comparison b|w sorting Algo.s

	Best case	Worst case	Avg case
1) Selection Sort	$O(N^2)$	$O(N^2)$	$O(N^2)$
2) Bubble Sort	$O(N)$	$O(N^2)$	$O(N^2)$
3) Merge Sort	$O(N\log N)$	$O(N\log N)$	$O(N\log N)$
4) Insertion Sort	$O(N)$	$O(N^2)$	$O(N^2)$
5) Quick Sort.	$O(N\log N)$	$O(N^2)$	$O(N\log N)$

↓

When the Array is getting divided into half every time.