

Sort n numbers in range from 0 to $n^2 - 1$ in linear time

Given an array of numbers of size n. It is also given that the array elements are in range from 0 to $n^2 - 1$. Sort the given array in linear time.

Examples:

Since there are 5 elements, the elements can be from 0 to 24.

Input: arr[] = {0, 23, 14, 12, 9}

Output: arr[] = {0, 9, 12, 14, 23}

Since there are 3 elements, the elements can be from 0 to 8.

Input: arr[] = {7, 0, 2}

Output: arr[] = {0, 2, 7}

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

Solution: If we use [Counting Sort](#), it would take $O(n^2)$ time as the given range is of size n^2 . Using any comparison based sorting like [Merge Sort](#), [Heap Sort](#), .. etc would take $O(n \log n)$ time.

Now question arises how to do this in $O(n)$? Firstly, is it possible? Can we use data given in question? n numbers in range from 0 to $n^2 - 1$?

The idea is to use [Radix Sort](#). Following is standard Radix Sort algorithm.

- 1) Do following for each digit i where i varies from least significant digit to the most significant digit.
.....a) Sort input array using counting sort (or any stable sort) according to the i'th digit

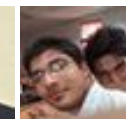
Google™ Custom Search



GeeksforGeeks



53,523 people like [GeeksforGeeks](#).



[Interview Experiences](#)

[Advanced Data Structures](#)

[Dynamic Programming](#)

[Greedy Algorithms](#)

[Backtracking](#)

[Pattern Searching](#)

[Divide & Conquer](#)

[Mathematical Algorithms](#)

[Recursion](#)

[Geometric Algorithms](#)

Let there be d digits in input integers. Radix Sort takes $O(d \cdot (n+b))$ time where b is the base for representing numbers, for example, for decimal system, b is 10. Since $n^2 - 1$ is the maximum possible value, the value of d would be $O(\log_b(n))$. So overall time complexity is $O((n+b) \cdot \log_b(n))$. Which looks more than the time complexity of comparison based sorting algorithms for a large k . The idea is to change base b . If we set b as n , the value of $O(\log_b(n))$ becomes $O(1)$ and overall time complexity becomes $O(n)$.

```
arr[] = {0, 10, 13, 12, 7}
```

Let us consider the elements in base 5. For example 13 in base 5 is 23, and 7 in base 5 is 12.

```
arr[] = {00(0), 20(10), 23(13), 22(12), 12(7)}
```

After first iteration (Sorting according to the last digit in base 5), we get.

```
arr[] = {00(0), 20(10), 12(7), 22(12), 23(13)}
```

After second iteration, we get

```
arr[] = {00(0), 12(7), 20(10), 22(12), 23(13)}
```

Following is C++ implementation to sort an array of size n where elements are in range from 0 to $n^2 - 1$.

```
#include<iostream>
using namespace std;

// A function to do counting sort of arr[] according to
// the digit represented by exp.
int countSort(int arr[], int n, int exp)
{
    int output[n]; // output array
    int i, count[n];
    for (int i=0; i < n; i++)
        count[i] = 0;

    // Store count of occurrences in count[]
    for (i = 0; i < n; i++)
        count[(arr[i]/exp)%n]++;
}
```



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](#)

```

// Change count[i] so that count[i] now contains actual
// position of this digit in output[]
for (i = 1; i < n; i++)
    count[i] += count[i - 1];

// Build the output array
for (i = n - 1; i >= 0; i--)
{
    output[count[ (arr[i]/exp)%n] - 1] = arr[i];
    count[(arr[i]/exp)%n]--;
}

// Copy the output array to arr[], so that arr[] now
// contains sorted numbers according to current digit
for (i = 0; i < n; i++)
    arr[i] = output[i];
}

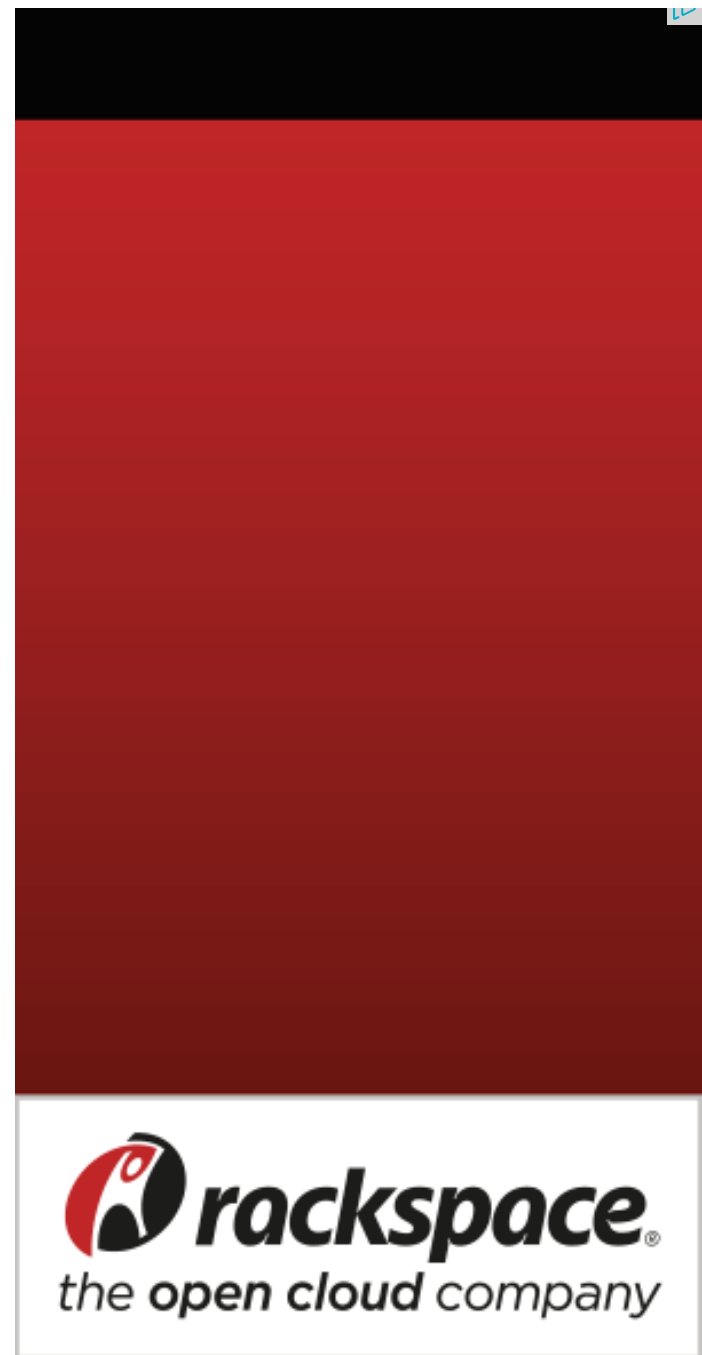
// The main function to that sorts arr[] of size n using Radix Sort
void sort(int arr[], int n)
{
    // Do counting sort for first digit in base n. Note that
    // instead of passing digit number, exp (n^0 = 1) is passed.
    countSort(arr, n, 1);

    // Do counting sort for second digit in base n. Note that
    // instead of passing digit number, exp (n^1 = n) is passed.
    countSort(arr, n, n);
}

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

// Driver program to test above functions
int main()
{
    // Since array size is 7, elements should be from 0 to 48
    int arr[] = {40, 12, 45, 32, 33, 1, 22};
    int n = sizeof(arr)/sizeof(arr[0]);
    cout << "Given array is \n";
    printArr(arr, n);
}

```



Recent Comments

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

Write a C program to Delete a Tree. · 30 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 34 minutes ago

Sanjay Agarwal bool

tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 59 minutes ago

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

Count trailing zeroes in factorial of a number · 1 hour 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

Find subarray with given sum · 1 hour ago

Find subarray with given sum · 1 hour ago

Find subarray with given sum · 1 hour ago

Find subarray with given sum · 1 hour ago

Find subarray with given sum · 1 hour ago

Find subarray with given sum · 1 hour ago

Find subarray with given sum · 1 hour ago

Find subarray with given sum · 1 hour ago

Find subarray with given sum · 1 hour ago

Find subarray with given sum · 1 hour ago

Find subarray with given sum · 1 hour ago

```
sort(arr, n);

cout << "\nSorted array is \n";
printArr(arr, n);
return 0;
}
```

Output:

Given array is
40 12 45 32 33 1 22
Sorted array is
1 12 22 32 33 40 45

How to sort if range is from 1 to n^2 ?

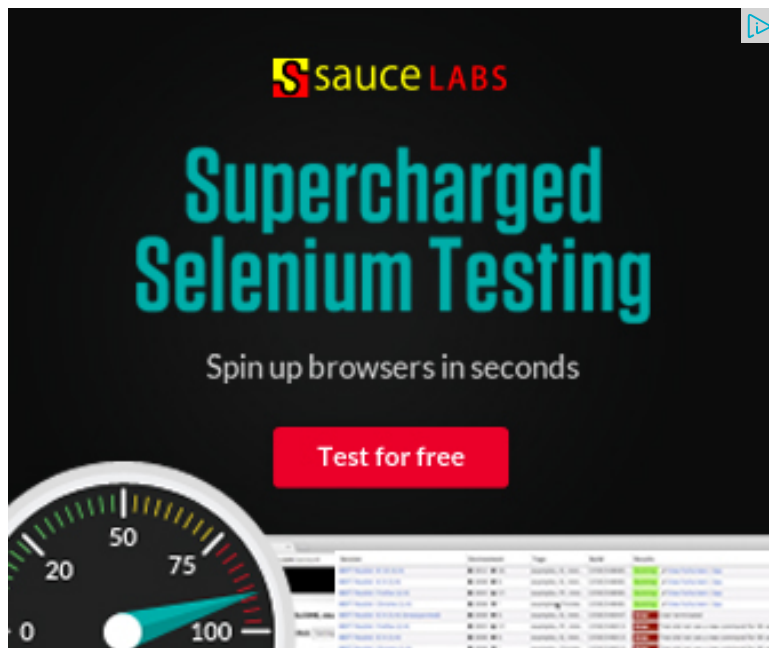
If range is from 1 to n^2 , the above process can not be directly applied, it must be changed. Consider $n = 100$ and range from 1 to 10000. Since the base is 100, a digit must be from 0 to 99 and there should be 2 digits in the numbers. But the number 10000 has more than 2 digits. So to sort numbers in a range from 1 to n^2 , we can use following process.

- 1) Subtract all numbers by 1.
- 2) Since the range is now 0 to n^2 , do counting sort twice as done in the above implementation.
- 3) After the elements are sorted, add 1 to all numbers to obtain the original numbers.

How to sort if range is from 0 to $n^3 - 1$?

Since there can be 3 digits in base n , we need to call counting sort 3 times.

This article is contributed by **Bateesh**. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above



Find subarray with given sum · 1 hour ago

AdChoices

► [Java Source Code](#)

► [4 Digit Numbers](#)

► [Counting Numbers](#)

AdChoices

► [Numbers Number](#)

► [Add the Numbers](#)

► [Numbers To](#)

AdChoices

► [Java Sort](#)

► [N Sort](#)

► [Java Array](#)

Related Topics:

- Remove minimum elements from either side such that $2 \times \text{min}$ 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



16



Tweet

6



1

Writing code in comment? Please use [ideone.com](#) and share the link here.

16 Comments

GeeksforGeeks

Sort by Newest ▼





with the recursion...



satinjal · a month ago

may u please refine the comment within sort function definition -> "instead of p passed". It would be more easier for beginner.

^ | v ·



sw · 2 months ago

The math doesnt seem right. You can replace one constant with the other whi is around 100 billion? The point is n is not constant so cant be used to replace

1 ^ | v ·



Hitesh · 2 months ago

Nice post! Building the output array cane be done in forward direction loop afte loop. That would make the life simpler!

^ | v ·



zzet → Hitesh · a month ago

actually, we have to iterate backward to keep the sort stable.

^ | v ·



Srithar · 2 months ago

This is called Radix Sort. No need for an article.

1 ^ | v ·



guest → Srithar · 2 months ago

It actually uses radix sort as a subroutine..Lot of questions are there w algorithm...Lot of people cant relate this to radix sort..We need it as a s

5 ^ | v ·



guest → guest · a month ago

... this is really sort of a trick, and the question is in essence of that.
But a good reminder.

^ | v .



sonu · 2 months ago

This question can be easily solved if Space is not important . We can use extra Array (B) . Sol for (i = 0 ; i < length ; i++) B[a[i]] = 1 ;

```
for(i = 0 ; i < n*n ; i++ ) if ( B[ i ] == 1 ) cout << i <<endl ;="">
```

^ | v .



Dev → sonu · 2 months ago

What about repeating elements? Also you are iterating a loop with n^2

1 ^ | v .



veer · 2 months ago

this problem can be solved in few loops.

b[] contain array of number to be sorted

first create a array of size =number² let a[]

then for every element in 'a' assign zero

same do for r[] where r[] is to keep track of number of repetition of number.

now assign 1 to the particular element in a[] who's index is b[i].

now reassign to input array by checking if element of a[i] is equal to 1 then assign

while take care of repetition by using value in r[i].

this algo will be O(n) if no repetition other wise $\sim O(n)$

here is sample code (

sorry it is not displaying properly)

```
// x is n^2
```

```
for(i=0;i<x;i++){ a[i]="0;" r[i]="0;" }="" for(i="0;i<number;i++){ a[b[i]]="1;" r[b[i]]="1;" repeated="" number="" }="" j="0;" for(i="0;i<x;i++){ if(a[i]="1"){ b[j]="i;" j++; } } for(i="0;i<number;i++){ for(j="0;j<r[b[i]];j++){ cout<<b[i]<<" "; }="" }
```

^ | v .



Abhishek → veer · 2 months ago

Yes you are right but you are using more space and logically you are not because you are going from $i=0$ to $i < n^2$ in one loop which logically equals

^ | v .



veer → Abhishek · 2 months ago

ya ,bro u r right.i didnt notice. sorry

^ | v .



Abhishek · 2 months ago

I think math behind the problem and the coding solution are not same because time it is sorting array according to the range of remainder (Mean if $N = 5$ than clustering all the numbers in the assending order of remainder and in second basics of divison means if we have two numbers 22 and 24 than dividend is 5 algorithm will arrange them as 22 and 24).

Correct Me If I am going on the wrong track

^ | v .



Kartik → Abhishek · 2 months ago

Abhishek, please note that the counting sort used in the program is a s theory and code by printing intermediate count[] and arr[] values.

^ | v .



Name · 2 months ago

ingenious*

^ | v .



andy qi · 2 months ago

ingenius!

2 ^ | v ·



Subscribe



Add Disqus to your site

@geeksforgeeks, **Some rights reserved**

Contact Us!

Powered by **WordPress & MooTools**, customized by geeksforgeeks team