













Problems



Quiz



Next >>

## Quick sort using Lomuto Partition

## **Lomuto's Partition Scheme:**



This algorithm works by assuming the pivot element as the last element. If any other element is given as a pivot element then swap it first with the last element. Now initialize two variables i as low and j also low, iterate over the array and increment i when arr[j] <= pivot and swap arr[i] with arr[j] otherwise increment only j. After coming out from the loop swap arr[i] with arr[hi]. This i stores the pivot element.

Refer <u>QuickSort</u> for details of this partitioning scheme. Below are implementations of this approach:-

```
C++
```

```
밂
  Dash
  000
   Αll
  \Box
Articles
  Videos
  </>>
Problems
  (?)
  Quiz
<<
    >>
```

```
#include<bits/stdc++.h>
using namespace std;
/* This function takes last element as pivot, places
   the pivot element at its correct position in sorted
    array, and places all smaller (smaller than pivot)
   to left of pivot and all greater elements to right
   of pivot */
int partition(int arr[], int low, int high)
    int pivot = arr[high]; // pivot
    int i = (low - 1); // Index of smaller element
    for (int j = low; j <= high-1; j++)
        // If current element is smaller than or
        // equal to pivot
        if (arr[j] <= pivot)</pre>
                  // increment index of smaller element
            i++;
            int temp = arr[i];
            arr[i] = arr[j];
            arr[j] = temp;
    int temp = arr[i + 1];
    arr[i + 1] = arr[high];
    arr[high] = temp;
```

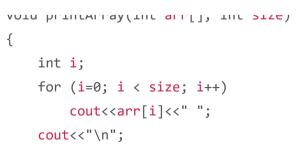


```
Dash
  000
  Αll
  \Box
Articles
  Videos
  </>
Problems
  (?)
  Quiz
```

Courses

```
return (i + 1);
}
/* The main function that implements QuickSort
 arr[] --> Array to be sorted,
 low --> Starting index,
 high --> Ending index */
void quickSort(int arr[], int low, int high)
    if (low < high)</pre>
        /* pi is partitioning index, arr[p] is now
           at right place */
        int pi = partition(arr, low, high);
        // Separately sort elements before
        // partition and after partition
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
```

**Tutorials** 



Practice

Contests





Jobs



```
}

// Driver program to test above functions
int main()
{
   int arr[] = {10, 7, 8, 9, 1, 5};
   int n = sizeof(arr)/sizeof(arr[0]);
   quickSort(arr, 0, n-1);
   cout<<"Sorted array: <<"\n";
   printArray(arr, n);
   return 0;
}</pre>
```

## Output

```
Sorted array:
1 5 7 8 9 10
```

Time Complexity:  $O(N^2)$ Auxiliary Space: O(1)

Marked as Read

Report An Issue

If you are facing any issue on this page. Please let us know.