The Learning Point

Search this site

Main Page: The Complete Index at a Glance | Mathematics | Computer Science | Physics
Electrical Science and Engineering | An Introduction to Graphics and Solid Modelling
Test Preparations | Ace the Programming Interviews
CoursePlex: Online Open Classes from Coursera, Udacity, etc | About

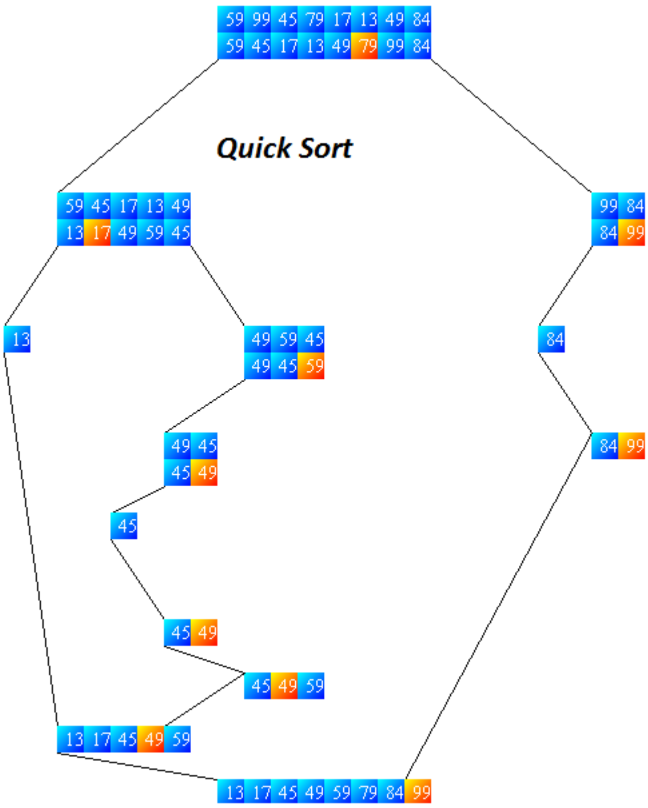
Quick Links to Various Sections on the Main Page

- The Learning Point
- Computer Science
- Algorithms: An introduction to Dynamic Programming
- CS - Data Structures and Algorithms
- CS - Programming Interviews
- CS - Miscellaneous C Programs
- CS - Intro to DB Queries
- Electrical Science and Engineering
- Electrical Sci - DC Circuits
- Electrical Sci - Digital Electronics
- Mathematics
- Mathematics - Linear Algebra
- Mathematics - Geometry
- Mathematics - Single Variable Calculus
- Mathematics - Game Theory
- Mathematics - Operations Research
- Physics
- Physics - Basic Mechanics
- Physics - Electrostatics
- Sitemap
- Recent site activity

These are Quick Links to sections of the Main Page

Computer Science >

Arrays and Sorting: Quick Sort (with C Program source code)



To go through the C program / source-code, scroll down to the end of this page

Quick Sort

Quick Sort is divide and conquer algorithm like Merge Sort. Unlike Merge Sort this does not require extra space. So it sorts in place. Here dividing step is to chose a pivot and partition the array such that all elements less than or equal to pivot are to the left of it and all the elements which are greater than or equal to the pivot are to the right of it. Recursively sort the left and right parts.

Algorithm (described in detail in the document for this tutorial)

The key to the algorithm is the partition procedure.
A 'partition' element is chosen. All elements less than the partition are put in the left half of the array, all elements greater than the partition are placed in the right half of the array.
The two halves are sorted independently and recursively.

Property:

1. Best case performance – When the partitioning produces two regions of size n/2 (where,
n is the total number of elements in the list) O(nlgn).
2. Worst case performance - When the partitioning produces one region of size n-1 (where,
n is the total number of elements in the list) and other of size 1 O(n2).
3. Average case – O(n(lgn))
4. It is not stable and uses O(lg(n)) extra space in the worst case.

Complete tutorial document with examples :

1 / 4

Quick Sort

Quick Sort is divide and conquer algorithm like Merge Sort. Unlike Merge Sort this does not require extra space. So it sorts in place. Here dividing step is to chose a pivot and partition the array such that all elements less than or equal to pivot are to the left of it and all the elements which are greater than or equal to the pivot are to the right of it. Recursively sort the left and right parts.

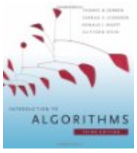
Algorithm:

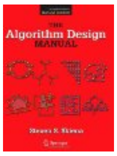
The key to the algorithm is the partition procedure.

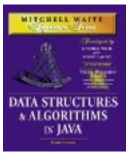
From index 0 to index n-1of the array a (where, n is the total number of elements in the list) do the following

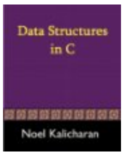
1. from = 0, to = n-1, choose the starting element as pivot.
2. i and j are such that in the array all elements a[from+1...i] are less than pivot, all elements a[i+1...j] are greater than pivot and the elements a[j+1...to] are which we have not seen which is like shown below.
   ________________i____________j____________ |
   pivot|....<=pivot......|....>=pivot......|.........|
3. If the new element we encounter is >=pivot, the above variant is still satisfied.
4. If it is less than pivot we swap a[j] with a[i+1].
5. Initially from = 0, to = number_of_elements-1 , pivot = a[0] and i = 0
6. For j = 1(from+1) to j = 5(to)
   If value at index j < pivot, increment i and swap a[i] and a[j].
7. Finally The picture is like shown below
   _________________________i______________________

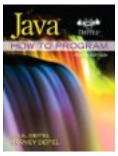
Books from Amazon which might interest you !











Quick Sort - C Program Source Code

```
#include<stdio.h>
/* Logic: This is divide and conquer algorithm like Merge Sort. Unlike Merge Sort this does not require
          extra space. So it sorts in place. Here dividing step is chose a pivot and parition the array
          such that all elements less than or equal to pivot are to the left of it andd all the elements
          which  are greater than or equal to the pivot are to the right of it. Recursivley sort the left
          and right parts.
*/

void QuickSort(int *array, int from, int to)
{
        if(from>=to)return;
        int pivot = array[from]; /*Pivot I am chosing is the starting element */
        /*Here i and j are such that in the array all elemnts a[from+1...i] are less than pivot,
          all elements a[i+1...j] are greater than pivot and the elements a[j+1...to] are which
          we have not seen which is like shown below.
          _______________________i_________________j___________
          |pivot|....<=pivot......|....>=pivot......|.........|
          If the new element we encounter than >=pivot the above variant is still satisfied.
          If it is less than pivot we swap a[j] with a[i+1].
         */
        int i = from, j, temp;
        for(j = from + 1;j <= to;j++)
        {
                if(array[j] < pivot)
                {
                        i = i + 1;
                        temp = array[i];
                        array[i] = array[j];
                        array[j] = temp;
                }
        }
        /* Finally The picture is like shown below
          _______________________i____________________
          |pivot|....<=pivot......|....>=pivot......|
        */
        temp = array[i];
        array[i] = array[from];
        array[from] = temp;
        /* So we the array is now
          ____________________i______________________
          |...<=pivot......|pivot|....>=pivot......|
        */
        /*Recursively sort the two sub arrays */
        QuickSort(array,from,i-1);
        QuickSort(array,i+1,to);
}
int main()
{
        int number_of_elements;
        scanf("%d",&number_of_elements);
        int array[number_of_elements];
        int iter;
        for(iter = 0;iter < number_of_elements;iter++)
        {
                scanf("%d",&array[iter]);
```

```
        }
        /* Calling this functions sorts the array */
        QuickSort(array,0,number_of_elements-1);
        for(iter = 0;iter < number_of_elements;iter++)
        {
                printf("%d ",array[iter]);
        }
        printf("\n");
        return 0;
}
```

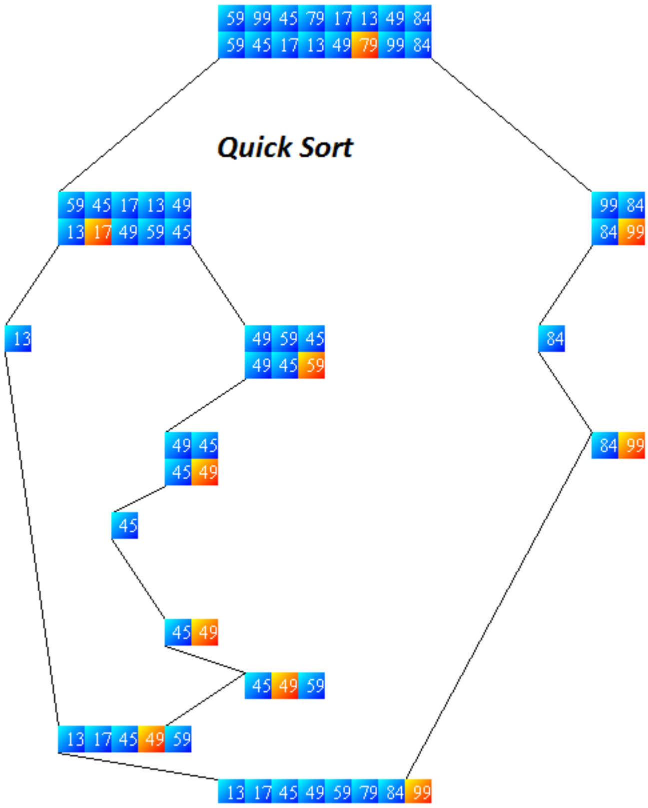
Related Tutorials :

Bubble Sort	One of the most elementary sorting algorithms to implement - and also very inefficient. Runs in quadratic time. A good starting point to understand sorting in general, before moving on to more advanced techniques and algorithms. A general idea of how the algorithm works and a the code for a C program.	**Insertion Sort**	Another quadratic time sorting algorithm - an example of dynamic programming. An explanation and step through of how the algorithm works, as well as the source code for a C program which performs insertion sort.	**Selection Sort**	Another quadratic time sorting algorithm - an example of a greedy algorithm. An explanation and step through of how the algorithm works, as well as the source code for a C program which performs selection sort.	**Shell Sort**	An inefficient but interesting algorithm, the complexity of which is not exactly known.	**Merge Sort**	An example of a Divide and Conquer algorithm. Works in O(n log n) time. The memory complexity for this is a bit of a disadvantage.	**Quick Sort**	In the average case, this works in O(n log n) time. No additional memory overhead - so this is better than merge sort in this regard. A partition element is selected, the array is restructured such that all elements greater or less than the partition are on opposite sides of the partition. These two parts of the array are then sorted recursively.	**Heap Sort**	Efficient sorting algorithm which runs in O(n log n) time. Uses the Heap data structure.	**Binary Search Algorithm**	C us al us th of el a s ar in tir

Visualizing Quick Sort (A Java Applet Visualization) :

Here's a Java Applet Visualization which might give you a more 'colorful' idea of what happens in Quick Sort.

Click here, or on the image beloq to check out a Quick Sort Algorithm - Java Applet Visualization



http://www.thelearningpoint.net/computer-science/sorting-algorithms/quick-sort

Some Important Data Structures and Algorithms, at a glance:

Arrays : Popular Sorting and Searching Algorithms			
Bubble Sort	Insertion Sort	Selection Sort	Shell Sort
Merge Sort	Quick Sort	Heap Sort	Binary Search Algorithm
Basic Data Structures and Operations on them			
Stacks	Queues	Single Linked List	Double Linked List
Circular Linked List	1.		

Tree Data Structures			
Binary Search Trees	Heaps	Height Balanced Trees	
Graphs and Graph Algorithms			
Depth First Search	Breadth First Search	Minimum Spanning Trees: Kruskal	Minumum Spanning Trees: Prim's

		Algorithm	Algorithm
Dijkstra Algorithm for Shortest Paths	Floyd Warshall Algorithm for Shortest Paths	Bellman Ford Algorithm	
Popular Algorithms in Dynamic Programming			
Dynamic Programming	Integer Knapsack problem	Matrix Chain Multiplication	Longest Common Subsequence
Greedy Algorithms			
Elementary cases : Fractional Knapsack Problem, Task Scheduling	Data Compression using Huffman Trees		

Rolly Ludan Casipit · Tabuk Institute
I was trying to count the number of comparison while sorting the
wrong. Hope you can help me on this.
Reply · Like · October 17 at 1:17pm

Jenelyn Amarila · EVSU
its not working!
Reply · Like · October 15 at 6:30am

Prashant Bhattacharji · IIT Kharagpur
What isn't working ? The C Program or the visualizatior
Reply · Like · October 16 at 1:06am

Rolly Ludan Casipit · Tabuk Institute
the implementation is perfectly working with me. try to
int main()

{

int number_of_elements = {7};

//scanf("How many elements? %d", number_of_element

...See More
Reply · Like · October 17 at 11:58am

Cherry Caduyac · Works at NoT yeT woRkiNg !! iM stiLl stUdYinG !
I don't know.
Reply · Like · October 2 at 6:05am

Shamim Reza Cse · Chittagong University
nice and clear.
Reply · Like · September 26 at 10:09am