

Claude

New chat

Chats

Recents

C++ QuickSort Implementation

C++ QuickSort Implementation

Write a quicksort program with cpp.

I'll write a quicksort implementation in C++.

QuickSort Implementation in C++

Code

This implementation of QuickSort in C++ includes:

1. A `partition` function that selects the last element as the pivot and places it at its correct position

2. The main `quickSort` recursive function that handles the divide-and-conquer approach

3. A helper `swap` function to exchange elements

4. Utility functions for printing the array and a main function to demonstrate the algorithm

The time complexity of QuickSort is:

Average case:  $O(n \log n)$

Worst case:  $O(n^2)$  - occurs when the array is already sorted or reverse sorted

Best case:  $O(n \log n)$

Would you like me to explain how the algorithm works or modify this implementation in any way?

Retry

Claude can make mistakes. Please double-check responses.

Reply to Claude...

Claude 3.7 Sonnet

Publish

Copy

```
#include <iostream>
#include <vector>

// Function to swap two elements
void swap(int& a, int& b) {
    int temp = a;
    a = b;
    b = temp;
}

// Partition function to place pivot at its correct position
int partition(std::vector<int>& arr, int low, int high) {
    // Using the last element as pivot
    int pivot = arr[high];

    // Index of smaller element
    int i = low - 1;

    for (int j = low; j < high; j++) {
        // If current element is smaller than or equal to pivot
        if (arr[j] <= pivot) {
            i++;
            swap(arr[i], arr[j]);
        }
    }

    // Place pivot at its correct position
    swap(arr[i + 1], arr[high]);
    return i + 1;
}

// The main QuickSort function
void quickSort(std::vector<int>& arr, int low, int high) {
    if (low < high) {
        // pi is partitioning index, arr[pi] is now at right place
        int pi = partition(arr, low, high);

        // Recursively sort elements before and after partition
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}

// Function to print an array
void printArray(const std::vector<int>& arr) {
    for (int num : arr) {
        std::cout << num << " ";
    }
}
```