**Program 1:** Write a program to implement Quick sort algorithm for sorting a list of integers in ascending order

**Quick Sort Algorithm**

**Input**: A list of integers, arr[], of size n.

**Base Case**:

If the list contains 0 or 1 element, it is already sorted. Return the list.

**Choose Pivot**:

Select a pivot element from the list. There are different ways to select a pivot (first element, last element, middle element, or a random element). Let's assume we are selecting the last element as the pivot.

**Partitioning**:

Rearrange the list such that elements less than the pivot come before it and elements greater than the pivot come after it. The pivot element should be placed in its correct position.

This can be done using two pointers, i and j:

i starts at the beginning of the list.

j traverses through the list.

When an element smaller than the pivot is found, swap it with the element at index i, then increment i.

After the loop ends, swap the pivot with the element at index i to place it in its correct position.

**Recursive Sort**:

Recursively apply the Quick Sort algorithm to the sublists:

Sort the sublist to the left of the pivot.

Sort the sublist to the right of the pivot.

**Repeat** steps 2–5 until the base case is met for all sublists.

**Output**: Return the sorted list.

**Code**

#include <iostream>

using namespace std;

// Function to swap two elements

void swap(int\* a, int\* b) {

    int t = \*a;

    \*a = \*b;

    \*b = t;

}

// Function to partition the array

int partition(int arr[], int low, int high) {

    int pivot = arr[high];

    int i = (low - 1);

    for (int j = low; j <= high - 1; j++) {

        if (arr[j] < pivot) {

            i++;

            swap(&arr[i], &arr[j]);

        }

    }

    swap(&arr[i + 1], &arr[high]);

    return (i + 1);

}

// Function to perform Quick Sort

void quickSort(int arr[], int low, int high) {

    if (low < high) {

        int pi = partition(arr, low, high);

        quickSort(arr, low, pi - 1);

        quickSort(arr, pi + 1, high);

    }

}

// Function to print the array

void printArray(int arr[], int size) {

    for (int i = 0; i < size; i++) {

        cout << arr[i] << " ";

    }

    cout << endl;

}

// Driver code

int main() {

int arr[] = {10,40,20,30,70,50,60,90,80};

int n = sizeof(arr) / sizeof(arr[0]);

    cout << "Original array: ";

    printArray(arr, n);

    quickSort(arr, 0, n - 1);

    cout << "Sorted array: ";

    printArray(arr, n);

    return 0;

}

**OUTPUT**

