Aim: write a program in c to perform quick sort

Algorithm for Quick Sort:

1. Choose a pivot element from the array.
2. Partition the other elements into two sub-arrays, according to whether they are less than or greater than the pivot.
3. Recursively sort the sub-arrays.
4. Combine the elements back into a single sorted array.

Source Code:

#include <stdio.h>

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

{

*int* temp = \*a;

    \*a = \*b;

    \*b = temp;

}

*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);

}

*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);

    }

}

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

{

*int* i;

    for (i = 0; i < size; i++)

        printf("%d ", arr[i]);

    printf("\n");

}

*int* main()

{

*int* arr[] = {10, 7, 8, 9, 1, 5};

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

    quickSort(arr, 0, n - 1);

    printf("Sorted array: \n");

    printArray(arr, n);

    return 0;

}

Output:

Sorted array:

1 5 7 8 9 10