EXERCISE 18

Write a C program to arrange a series of numbers using Quick Sort

Aim:

To write a C program to sort a series of numbers using the Quick Sort algorithm.

Algorithm:

1. Select a pivot element.
2. Rearrange the elements such that all elements less than the pivot go to the left, and greater to the right (partitioning).
3. Recursively apply the same logic to the left and right subarrays.
4. Continue until the array is sorted.

Program:

```
#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]; // pivot
   int i = (low - 1);      // index of smaller element
   for (int j = low; j < high; 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); // partition index
        quickSort(arr, low, pi - 1);  // before pivot
        quickSort(arr, pi + 1, high); // after pivot
    }
}
void display(int arr[], int size) {
    printf("Sorted array:\n");
    for (int i = 0; i < size; i++)
        printf("%d ", arr[i]);
    printf("\n");
}
int main() {
    int arr[50], n;

    printf("Enter number of elements: ");
    scanf("%d", &n);
    printf("Enter %d elements:\n", n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);
    quickSort(arr, 0, n - 1);
    display(arr, n);
```

Input and Output :

```
Enter number of elements: 5
Enter 5 elements:
10
2 3 5 6

Sorted array:
2 3 5 6 10


=== Code Execution Successful ===
```

Result :

The series of numbers has been successfully sorted using the Quick Sort method.