18. Quick Sort

Aim:

To arrange a series of numbers using Quick Sort.

Algorithm:

1. Choose a pivot element.

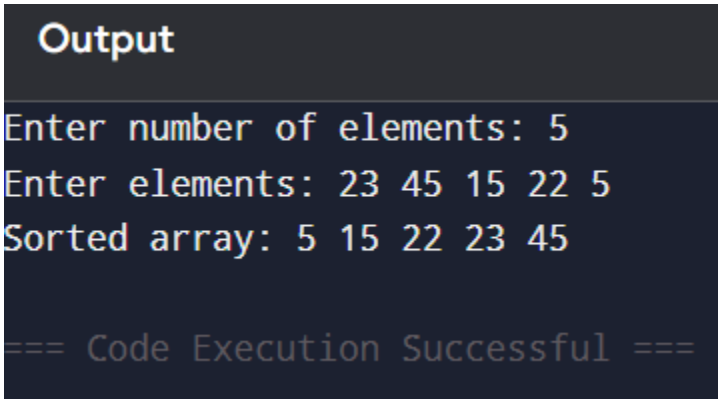
2. Partition the array such that elements smaller than pivot are on the left, larger on the right.

3. Recursively apply the same logic to left and right subarrays.

CODE:

```
#include <stdio.h>

int partition(int arr[], int low, int high) {
   int pivot = arr[high], i = low - 1;
   for (int j = low; j < high; j++) {
      if (arr[j] < pivot) {
         i++;
         int temp = arr[i]; arr[i] = arr[j]; arr[j] = temp;
      }
   }
   int temp = arr[i + 1]; arr[i + 1] = arr[high]; arr[high] = temp;
   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);
   }
}
```

```
int main() {
    int n, arr[20];
    printf("Enter number of elements: ");
    scanf("%d", &n);
    printf("Enter elements: ");
    for (int i = 0; i < n; i++) scanf("%d", &arr[i]);

    quickSort(arr, 0, n - 1);

    printf("Sorted array: ");
    for (int i = 0; i < n; i++) printf("%d ", arr[i]);
    return 0;
}
```



RESULT:

The program successfully executed and displayed the quick sort method.