

```
1: #include <stdio.h>
2:
3: void printArray(int *A, int n)
4: {
5:     for (int i = 0; i < n; i++)
6:     {
7:         printf("%d ", A[i]);
8:     }
9:     printf("\n");
10: }
11:
12: int partition(int A[], int low, int high)
13: {
14:     int pivot = A[low];
15:     int i=low+1;
16:     int j=high;
17:     int temp;
18:
19:     do
20:     {
21:
22:         for(;i<=high && A[i]<pivot;i++){
23:
24:         }
25:
26:
27:
28:
29:         for(;j>=low && A[j]>pivot;j--){
30:
31:         }
32:
33:         if (i < j)
34:         {
35:             temp = A[i];
36:             A[i] = A[j];
37:             A[j] = temp;
38:         }
39:     } while (i < j);
```

```

40:
41:     // Swap A[low] and A[j]
42:     temp = A[low];
43:     A[low] = A[j];
44:     A[j] = temp;
45:     return j;
46: }
47:
48: void quickSort(int A[], int low, int high)
49: {
50:     int partitionIndex; // Index of pivot after partition
51:
52:     if (low < high)
53:     {
54:         partitionIndex = partition(A, low, high);
55:         quickSort(A, low, partitionIndex - 1); // sort left subarray
56:         quickSort(A, partitionIndex + 1, high); // sort right subarray
57:     }
58: }
59:
60: int main()
61: {
62:     //int A[] = {3, 5, 2, 13, 12, 3, 2, 13, 45};
63:     int A[] = {9, 4, 4, 8, 7, 5, 6};
64:     // 3, 5, 2, 13, 12, 3, 2, 13, 45
65:     // 3, 2, 2, 13i, 12, 3j, 5, 13, 45
66:     // 3, 2, 2, 3j, 12i, 13, 5, 13, 45 --> first call to partition
67:     int n = 9;
68:     n = 7;
69:     printArray(A, n);
70:     quickSort(A, 0, n - 1);
71:     printArray(A, n);
72:     return 0;
73: }

```