# <2T>

# Quicksort Description

Quicksort, like merge sort, applies the divide-and-conquer paradigm introduced

in Section 2.3.1. Here is the three-step divide-and-conquer process for sorting a

typical subarray A[p..r]:

**Divide:** Partition (rearrange) the array a[p..r] into two (possibly empty) subarrays

A[p..q-1] and A[q + 1..r] such that each element of A[p..q - 1] is

less than or equal to A[q], which is, in turn, less than or equal to each element

of A[q+1 ..r]. Compute the index q as part of this partitioning procedure.

**Conquer:** Sort the two subarrays A[p..q - 1] and A[q + 1..r] by recursive calls

to quicksort.

**Combine:** Because the subarrays are already sorted, no work is needed to combine

them: the entire array A[p..r] is now sorted.

# Quicksort Pseudocode

QUICKSORT(A, p, r)  
1 if p < r  
2 q = PARTITION(A, p, r)  
3 QUICKSORT(A, p, q-`1)  
4 QUICKSORT{A, q + 1, r}

**Partitioning the array**

The key to the algorithm is the PARTITION procedure, which rearranges the subarray

A[p..r] in place.

**Partition Pseudo Code**

PARTITION(A, p, r)  
1 x = A[r]  
2 i = p – 1  
3 for j = p to r – 1  
4 if A[j] <= x  
5 I = i + 1  
6 exchange A[i] with A[j]  
7 exchange A[I + 1] with A[r]  
8 return i + 1

# Quicksort code

**def** **quicksort**(myList, start, end):

**if** start **<** end:

*# partition the list*

pivot **=** partition(myList, start, end)

*# sort both halves*

quicksort(myList, start, pivot**-**1)

quicksort(myList, pivot**+**1, end)

**return** myList

**def** **partition**(myList, start, end):

pivot **=** myList[start]

left **=** start**+**1

right **=** end

done **=** False

**while** **not** done:

**while** left **<=** right **and** myList[left] **<=** pivot:

left **=** left **+** 1

**while** myList[right] **>=** pivot **and** right **>=**left:

right **=** right **-**1

**if** right **<** left:

done**=** True

**else**:

*# swap places*

temp**=**myList[left]

myList[left]**=**myList[right]

myList[right]**=**temp

*# swap start with myList[right]*

temp**=**myList[start]

myList[start]**=**myList[right]

myList[right]**=**temp

**return** right