QuickSort

Is sorting things (say, in array, recursively)

Let's say we are sorting elements in array A, i.e, quicksort(A);

1. If the number of elements in A is 0 or 1, just return the array as your answer
2. Otherwise, Pick one element from the array. This element will be called "pivot"
3. Excluding the pivot, divide A into two partition
    - The first partition contains elements that are less than or equal to the pivot
    - The second partition contains elements that are greater than or equal to the pivot
4. The overall answer is quicksort(Part1) ++ pivot ++ quicksort(Part2)

Choosing Pivot

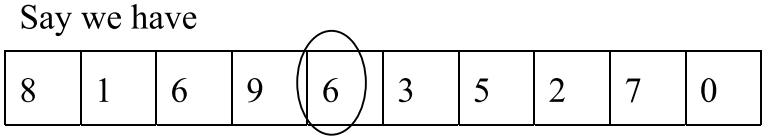
Do not choose the first element of A because if the array is originally nearly sorted or reversed sorted, All the elements will go to only one partition.

You should choose the pivot randomly.

Another way is to choose the median value from the first, the last, and the middle element of the array.

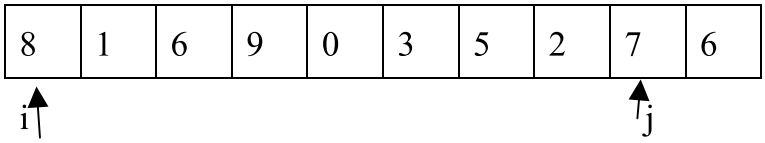
Partitioning

Say we have



And 6 is chosen as the pivot.

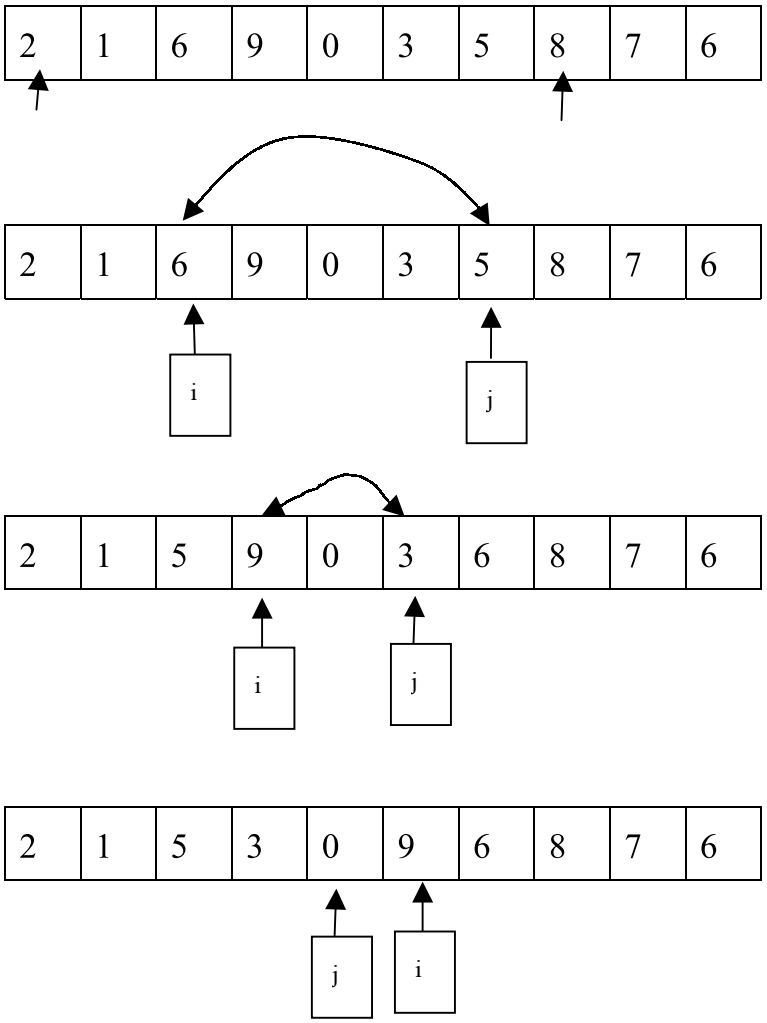
1. Swap pivot with the last element, we get



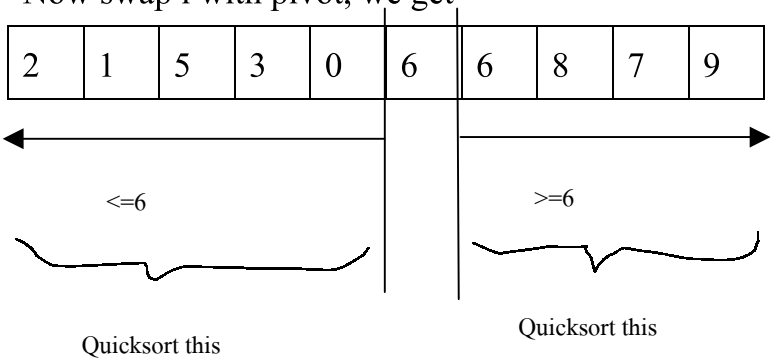
Let i and j be index shown in the picture.

2. While i is still on the left of j
   - Move i to the right, do not stop until encounter the element that is greater than or equal to the pivot
   - Move j to the left, do not stop until encounter the element that is less than or equal to the pivot
   - When i and j has stopped, we swap the element at i and j. This is an attempt to move the small number to the front and large number to the back. In this example, 8 and 2 must swap position.
   - repeat

3. When i is not on the left of j, swap i with the pivot. Now the smaller numbers are on the left of the pivot and the larger numbers are on the right of the pivot.



Now swap i with pivot, we get



Quicksort this

Quicksort this