Quick

First element is taken as pivot

**10**, 16, 8, 12, 15, 6, 3, 9, 5

i j i will keep on increasing till it find the element which is > than pivot

i j Since 16>10(pivot) i stops at index 1, and 5<10 so index of j is 8

**5** **16 Swap both**

10, **5**, 8, 12, 15, 6, 3, 9, **16** result

i j Since 12>10 i stops at index 3, and 9<10 so index of j is 7

**9** **12 Swap both**

10, 5, 8, **9**, 15, 6, 3, **12**, 16 result

i j Since 15>10 i stops at index 4, and 3<10 so index of j is 6

**3** **15** **Swap both**

**10**, 5, 8, 9, 3, **6**, 15, 12, 16

j i

6, 5, 8, 9, 3, **10**, 15, 12, 16 Partition is done and we got 10 as partitioned element

public class QuickSort {  
 public static void main(String[] args) {  
 int[] x = {9, 2, 4, 7, 3, 7, 10};  
 System.*out*.println(Arrays.*toString*(x));  
  
 int low = 0;  
 int high = x.length - 1;  
  
 *quickSort*(x, low, high);  
 System.*out*.println(Arrays.*toString*(x));  
 }  
  
 private static void quickSort(int[] array, int low, int high) {  
 if (low < high) {  
 int j = *partition*(array, low, high);  
 *quickSort*(array, low, j);  
 *quickSort*(array, j + 1, high);  
 }  
 }  
  
 static int partition(int[] array, int low, int high) {  
  
 int pivot = array[low];  
 int i = low;  
 int j = high;  
  
 while (i < j) {  
  
 */\*\*  
 \* It iterates from lowest index till it found an element greater than pivot  
 \* i.e, iterates continuously though smaller elements  
 \** ***@param*** *i is the index from the left  
 \*/* do {  
 i++;  
 } while (array[i] < pivot);  
  
 */\*\*  
 \* It iterates for highest till it found an element less than pivot  
 \* i.e, iterates continuously though higher elements  
 \** ***@param*** *j is the index from the right  
 \*/* do {  
 j--;  
 } while (array[j] > pivot);  
  
 */\*\*  
 \* when i is less than j, swap both the element  
 \*/* if (i < j) {  
 int temp = array[i];  
 array[i] = array[j];  
 array[j] = temp;  
 }  
 }  
  
 */\*\*  
 \* When i will be equal or greater than j, loop ends  
 \* finally swaps pivot element with element with jth index  
 \*/* int temp = array[low];  
 array[low] = array[j];  
 array[j] = temp;  
  
 return j;  
 }  
}