```java
/** Implementation of the Algorithm QuickSort
 * @author ameliado
 */
public class QuickSort {

    /**
     *
     * @param items
     * @param i
     * @param j
     * @return
     */
    public static <E> int partition(E[] items, int i, int j)
    {
        int middle = (i + j) / 2;

        // 1. pick a pivot element and swap it with the last item
        // in the range
        E pivot = items[middle];
        swap(items, middle, j);

        // 5. Repeat steps 2-4 until i,j meet
        int left = i;
        int right = j - 1;

        while (left <= right) {
            // 2. move forward index left until it finds an item bigger than the pivot
            while (left <= right && SortUtils.compare(items[left], pivot) < 0) {
                left++;
            }
            // 3. move backward index right until it finds an item smaller than the pivot
            while (left <= right && SortUtils.compare(items[right], pivot) >= 0) {
                right--;
            }
            // 4. exchange the values at left, right
```

```
36              if (left < right) {
37                  swap(items, left, right);
38                  left++;
39                  right--;
40              }
41          }
42          // 6. exchange the pivot and the item at index left
43          swap(items, left, j); // Put the pivot in its proper
  place
44
45          return left; // Return the position of the pivot
46      }
47
48      private static <E> void swap(E[] items, int i, int j) {
49          SortUtils.swap(items, i, j);
50      }
51
52      /**
53       *
54       * @param items
55       * @param i
56       * @param j
57       */
58      public static <E> void sort(E[] items, int i, int j)
59      {
60          if (i < j) {
61              int pivotIndex = partition(items, i, j);
62              sort(items, i, pivotIndex - 1);
63              sort(items, pivotIndex + 1, j);
64          }
65      }
66
67      /**
68       *
69       * @param items
70       */
71      public static <E> void sort(E[] items)
72      {
73          sort(items, 0, items.length - 1);
```

```
74    }
75}
76
```