QuickSort.sort

QuickSort.java

```
import java.util.List;

public class QuickSort {

    //Function to determine the partitions
    // partitions the array and returns the middle index (subscript)

    static int partition(List<Integer>arr, int top, int bottom)
    {
         // Set x=arr[top]
         // Swap all arr entries >= x to the bottom
         // Swap all arr entries < x to the top
         // Return the first index of last entry in first section

        Integer midVal = arr.get(top);
        while (bottom > top)
        {
            if ( arr.get(top+1)< midVal)
            {
                Integer temp = arr.get(top);
                arr.set(top, arr.get(top+1));
                arr.set(top+1, temp);
                top += 1;
            }
            else
            {
                Integer temp = arr.get(bottom);
                arr.set(bottom, arr.get(top+1));
                arr.set(top+1, temp);
                bottom -= 1;
            }
        }
        return top;

    }
    static void sort(List<Integer> arr)
    {
        qsort(arr, 0, arr.size()-1);
    }

    static void qsort(List<Integer>arr, int top, int bottom)
    {
         // top = subscript of beginning of vector being considered
         // bottom = subscript of end of vector being considered
         // this process uses recursion - the process of calling itself
        int middle;
        if (top < bottom)
        {
```

```
            middle = partition(arr, top, bottom);
            qsort(arr, top, middle);   // sort top partition
            qsort(arr, middle+1, bottom);    // sort bottom partition
        }
        return;
    }

}
```

Last Updated:

November 8, 2013 10:13 PM