Sorting Practice

For all sorting algorithms: **Time and Space complexity. Stable? Adaptive? (Data moves)**

Algorithm	Stable	Adaptive	Time complexity			Space complexity
			Worst	Avg	Best	
Merge sort	Y	N	NlgN	NlgN	NlgN	N
Quick sort	N	N	N^2	NlgN	NlgN	Best: lgN Worst: N

Quick sort

QS1. Is quick sort stable?

If yes, prove it. If no, give an example array, A, (of size 5 or less), sort it with Quick_Sort, and indicate why it is not stable. Use the original array and the final, sorted array to base your proof (do not base your proof on a partially sorted array).

Hint: Focus on the pivot jump.

No. It is not stable. Example for last item used as pivot: [1,6a,6b] after partition, because of the pivot swap we get: [1,6b,6a] and the algorithm ends ([1]and [6a] are base cases).

QS2. a) We make the call: `int res = partition(a, 0, 6);`

for each of the 2 example arrays **a** given in the table below. Give the arrays after the call, and the value returned in `res`. Use the partition method from Cormen with **LAST** item used as pivot .

	0	1	2	3	4	5	6	res
Original array **a** example 1	**13**	**7**	**12**	**8**	**6**	**15**	**10**	
Array after partition	**7**	**8**	**6**	**10**	**12**	**15**	**13**	**3**
Original array **a** example 2	**17**	**11**	**12**	**16**	**3**	**8**	**9**	
Array example 2 after partition	**3**	**8**	**9**	**16**	**17**	**11**	**12**	**2**

b) We make the call: `int res = partition(a, 0, 6);`

for each of the 2 example arrays **a** given in the table below. Give the arrays after the call, and the value returned in `res`. Use the partition method from web with **FIRST** item used as pivot .

	0	1	2	3	4	5	6	res
Original array **a** example 1	**13**	**7**	**12**	**8**	**6**	**15**	**10**	
Array after partition	**10**	**7**	**12**	**8**	**6**	**13**	**15**	**5**
Original array **a** example 2	**17**	**11**	**12**	**16**	**3**	**8**	**9**	
Array example 2 after partition	**9**	**11**	**12**	**16**	**3**	**8**	**17**	**6**

Merge sort

MS1. Show the array below after each call to the Merge (not Mergesort). Highlight the elements that are modified or "touched" by merge.

Index	0	1	2	3	4	5
Orig array	15	11	12	13	17	10
	11	**15**	12	13	17	10
	11	**12**	**15**	13	17	10
	11	12	15	**13**	**17**	10
	11	12	15	**10**	**13**	**17**
	10	**11**	**12**	**13**	**15**	**17**