CS303 Homework 7

Due: 3:30 pm, Wednesday, 10/24

Eric Shih
collaborated with Mark Pfluger, Ben Kayton, and Michael Trotta

1. Initial Input: 42 57 7 40 83 78 86 89 80 91 79 84

Quick Sort	42	57	7	40	83	78	86	89	80	91	79	84	$left: 0, right: 11$
median3	42	57	7	40	83	79	86	89	80	91	78	84	$left: 0, right: 11$
Final Swap	42	57	7	40	78	79	86	89	80	91	83	84	
Quick Sort	42	57	7	40	78	79	86	89	80	91	83	84	$left: 0, right: 3$
Insertion Sort(before):	42	57	7	40	78	79	86	89	80	91	83	84	
insertion Sort(after):	7	40	42	57	78	79	86	89	80	91	83	84	
Quick Sort	7	40	42	57	78	79	86	89	80	91	83	84	$left: 5, right: 11$
median3	7	40	42	57	78	79	86	89	83	91	80	84	$left: 5, right: 11$
Final Swap	7	40	42	57	78	79	80	89	83	91	86	84	
Quick Sort	7	40	42	57	78	79	80	89	83	91	86	84	$left: 5, right: 5$
Insertion Sort (before)	7	40	42	57	78	79	80	89	83	91	86	84	
Insertion Sort (after)	7	40	42	57	78	79	80	89	83	91	86	84	
Quick Sort	7	40	42	57	78	79	80	89	83	91	86	84	$left: 7, right: 11$
Insertion Sort (before)	7	40	42	57	78	79	80	89	83	91	86	84	
Insertion Sort (after)	7	40	42	57	78	79	80	83	84	86	89	91	
Final Array:	7	40	42	57	78	79	80	83	84	86	89	91	

Number of Shifts: 10 Number of Swaps: 4 Total Shifts and Swaps: 14

2. Initial Input: 42 57 7 40 83 78 86 89 80 91 79 84

Quick Sort	42	57	7	40	83	78	86	89	80	91	79	84	$left: 0, right: 11$
median3	42	57	7	40	83	79	86	89	80	91	78	84	$left: 0, right: 11$
Final Swap	42	57	7	40	78	79	86	89	80	91	83	84	
Quick Sort left: 0 right: 3	42	57	7	40	78	79	86	89	80	91	83	84	
median3	40	7	42	57	78	79	86	89	80	91	83	84	$left: 0, right: 3$
Final Swap	40	7	42	57	78	79	86	89	80	91	83	84	
Quick Sort	40	7	42	57	78	79	86	89	80	91	83	84	$left: 0, right: 1$
Insertion Sort (before)	40	7	42	57	78	79	86	89	80	91	83	84	
Insertion Sort (after)	7	40	42	57	78	79	86	89	80	91	83	84	
Quick Sort	7	40	42	57	78	79	86	89	80	91	83	84	$left: 3, right: 3$
Insertion Sort (before)	7	40	42	57	78	79	86	89	80	91	83	84	
Insertion Sort (after)	7	40	42	57	78	79	86	89	80	91	83	84	
Quick Sort	7	40	42	57	78	79	86	89	80	91	83	84	$left: 5, right: 11$
median3	7	40	42	57	78	79	86	89	83	91	80	84	$left: 5, right: 11$
Final Swap	7	40	42	57	78	79	80	89	83	91	86	84	
Quick Sort	7	40	42	57	78	79	80	89	83	91	86	84	$left: 5, right: 5$
Insertion Sort (before)	7	40	42	57	78	79	80	89	83	91	86	84	
Insertion Sort (after)	7	40	42	57	78	79	80	89	83	91	86	84	
Quick Sort	7	40	42	57	78	79	80	89	83	91	86	84	$left: 7, right: 11$
median3	7	40	42	57	78	79	80	84	83	86	89	91	$left: 7, right: 11$
Final Swap	7	40	42	57	78	79	80	84	83	86	89	91	
Quick Sort	7	40	42	57	78	79	80	84	83	86	89	91	$left: 7, right: 9$
median3	7	40	42	57	78	79	80	83	84	86	89	91	$left: 7, right: 9$
Final Swap	7	40	42	57	78	79	80	83	84	86	89	91	
Quick Sort	7	40	42	57	78	79	80	83	84	86	89	91	$left: 7, right: 7$
Insertion Sort (before)	7	40	42	57	78	79	80	83	84	86	89	91	
Insertion Sort (after)	7	40	42	57	78	79	80	83	84	86	89	91	
Quick Sort	7	40	42	57	78	79	80	83	84	86	89	91	$left: 9, right: 9$
Insertion Sort (before)	7	40	42	57	78	79	80	83	84	86	89	91	
Insertion Sort (after)	7	40	42	57	78	79	80	83	84	86	89	91	
*hline*Quick Sort	7	40	42	57	78	79	80	83	84	86	89	91	$left: 11, right: 11$
Insertion Sort (before)	7	40	42	57	78	79	80	83	84	86	89	91	
Insertion Sort (after)	7	40	42	57	78	79	80	83	84	86	89	91	
Final Array:	7	40	42	57	78	79	80	83	84	86	89	91	

Number of Shifts: 1 Number of Swaps: 15 Total Number of Shifts and Swaps: 16

3. It is better to use a hybrid strategy over only having one sort because the best of both algorithms can be used. In our case the insertion sort is better suited to the smaller cases, while quicksort is better for larger cases. The values from the previous questions did not support my assertion, but this is only because there were several unnecessary insertion sort movements done on a partition with only one element. Without the unnecessary swaps, the hybrid strategy can be shown to be more effective.

4. The quicksort will still work because it does not matter where the pivot is placed.

Quick Sort	42	57	7	40	83	78	86	89	80	91	79	84	$left: 0, right: 11$
median3	42	57	7	40	83	78	86	89	80	91	79	84	$left: 0, right: 11$
Final Swap	42	40	7	79	83	78	86	89	80	91	57	84	

5. As with question 4, the quicksort will still work because it does not matter where the pivot is placed.

Quick Sort	42	57	7	40	83	78	86	89	80	91	79	84	$left : 0, right : 11$
median3	42	57	7	40	83	78	86	89	80	91	79	84	$left : 0, right : 11$
Final Swap	42	57	7	40	83	78	80	79	86	91	89	84	

6. If the initial if statement still reads $left+2 <= right$ then the quicksort algorithm will work, otherwise it will skip directly to insertion sort because there are not enough elements in the array. Below is the illustration of the first call when the algorithm works:

Quick Sort	3	4	1	2	5	$left : 0, right : 4$
median3	3	4	5	2	1	$left : 0, right : 4$
Final Swap	1	4	5	2	3	
Quick Sort	1	4	5	2	3	$left : 0, right : -1$