CSE 332 Summer 2010
Section Worksheet #5

1. Sort 3, 1, 4, 1, 5, 9, 2, 6, 5 using insertion sort.

Input	3	1	4	1	5	9	2	6	5
i=0	*3	1	4	1	5	9	2	6	5
i=1	3	1*	4	1	5	9	2	6	5
Insert	1	3	4	1	5	9	2	6	5
i=2	1	3	*4	1	5	9	2	6	5
i=3	1	3	4	*1	5	9	2	6	5
Insert	1	1	3	4	5	9	2	6	5
i=4	1	1	3	4	*5	9	2	6	5
i=5	1	1	3	4	5	*9	2	6	5
i=6	1	1	3	4	5	9	*2	6	5
Insert	1	1	2	3	4	5	9	6	5
i=7	1	1	2	3	4	5	9	*6	5
Insert	1	1	2	3	4	5	6	9	5
i=8	1	1	2	3	4	5	6	9	*5
	1	1	2	3	4	5	5	6	9

2. Sort 3, 1, 4, 1, 5, 9, 2, 6, 5 using merge sort.

Continually cut input into half.
1. (3, 1, 4, 1, 5, 9, 2, 6, 5)
2. (3, 1, 4, 1) (5, 9, 2, 6, 5)
3. (3, 1) (4, 1) (5, 9) (2, 6, 5)
4. (3) (1) (4) (1) (5) (9) (2,6) (5)
5. (3) (1) (4) (1) (5) (9) (2) (6) (5) (on this step, only need to divide the last chunk undivided chunk

Now merge back together, sorting each pair by going through each pair that was divided, adding the lesser values of one to a new array until the other array in the pair has lesser values, add all those, etc.
6. (1,3)(1,4) (5,9) (2,6) (5)
7. (1, 1, 3, 4) (5,9) (2,5,6)
8. (1, 1, 3, 4) (2, 5, 5, 6, 9)
9. (1, 1, 2, 3, 4, 5, 5, 6, 9)

3. Sort 3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5 using quick sort with median-of-three pivot, with insertion sort cutoff at 3.

At first step, take median of items at index 0, index 10, and index 5. Use this as your pivot. So for the first example, pivot is median of 3, 9, and 5, answer is 5. Place pivot at end of array (already there at the moment).

Then go through array on the left (pointer i) until you find a value greater than or equal to the pivot. Go through array on right until you find value less than the pivot (pointer j). Stop and swap.

Input:	3	1	4	1	5	9	2	6	5	3	5
Pivot:5	3	1	4	1	5	9	2	6	5	3	5
	3	1	4	1	5					3	5
i&j stopped					i					j	
Swap 5 & 3	3	1	4	1	3	9	2	6	5	5	5
						i	j				
Swap	3	1	4	1	3	2	9	6	5	5	5
When i & j cross, stop						j	i				
Swap pivot with i	3	1	4	1	3	2	5	6	5	5	5

Repeat by picking another pivot, etc. until sorted.

4. Sort 25, 36, 85, 93, 21, 74, 22, 12 using radix sort with radix=10.

Table shows the two passes below. After first pass, go through the buckets and output in order: 21, 22, 12, 93, 74,25,85,36. Then take that input, and use it to second pass, and output results from that pass similarly (sorted order).

	0	1	2	3	4	5	6	7	8	9
Pass 1		21	22	93	74	25	36			
			12			85				
Pass 2		12	21	36				74	85	93
			22							
			25							

5. What would be the runtimes of the following algorithms if your data were all identical (only one unique item, e.g 7,7,7,7,7), sorted, or reverse sorted?

	Identical	Sorted	Reverse-sorted
Insertion Sort	O(n)	O(n)	$O(n^2)$
Selection Sort	$O(n^2)$	$O(n^2)$	$O(n^2)$
Heapsort	O(n)	O(n log n)	O(n log n)
Mergesort	O(n log n)	O(n log n)	O(n log n)
Quicksort	$O(n^2)$	O(n log n)	O(n log n)
Bucket Sort	O(n+k)	O(n+k)	O(n+k)
Radix Sort	O(P(B+n))	O(P(B+n))	O(P(B+n))