Algorithms Midterm Exam 111

Name: Id:

1. Assume that you are given a sorting program/library/module. With the sorting module, you can call the sorting module by passing an array, and the sorting module will return a sorted array. However, you do not have/know the source code of sorting program. Under such circumstance, please state how do you estimate the performance of the sorting program.(5 pts.)

2. Please give the sorting trace of the 3-way quick sort algorithm for sorting the following array [1, 8, 3, 3, 2, 2, 7].(5 pts.)

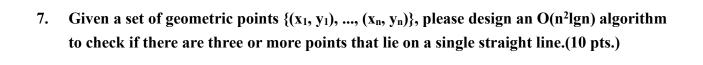
- 3. You are given the code of the weighted quick union algorithm as shown in Figure 1.
 - A. Please prove that the worst-case order of growth of the cost of find() operation for the weighted quick union algorithm is ~lgN, where N is the number of sites. (5 pts.)
 - B. If we use the same basic strategy as weighted quick-union but keep track of tree height (rather than tree size) and always link the shorter tree to the taller one, what will happen to the worst-case order of growth of the cost of find() operation?

```
(5 pts.)
                   public class WeightedQuickUnionUF
{
                         private int[] id;  // parent link (site indexed)
private int[] sz;  // size of component for roots (site indexed)
private int count;  // number of components
                         public WeightedQuickUnionUF(int N)
                               count = N:
                               id = new int[N];
for (int i = 0; i < N; i++) id[i] = i;
                               sz = new int[N];
for (int i = 0; i < N; i++) sz[i] = 1;</pre>
                         }
                         public int count()
                             return count;
                         public boolean connected(int p, int q)
                             return find(p) == find(q); }
                         private int find(int p)
{    // Follow links to find a root.
                               while (p != id[p]) p = id[p];
                               return p;
                         public void union(int p, int q)
                               int i = find(p);
                              int j = find(q);
if (i == j) return;
                              // Make smaller root point to larger one.
if (sz[i] < sz[j]) { id[i] = j; sz[j] += sz[i]; }
else { id[j] = i; sz[i] += sz[j]; }
                              count--;
                        }
```

}

Figure 1

4.	True or False (10 pts, each 2 pts.)		
	A.	The selection sort is an in-place sorting algorithm.	
	B.	A quick sort implementation with a cutoff to insertion sort for subarrays with less	
		than 10 elements is not a stable sorting algorithm.	
	C.	The Tim sort is an unstable sorting algorithm.	
	D.	The Tim sort is an in-place sorting algorithm.	
	E.	Quick sort is an O(NlgN) algorithm for sorting arrays	
5.	Please prove that the merge sort algorithm is optimal compare-based sorting algorithm.		
	(10 pts.)		
6.	Giv	ven an array A = [b, b, a, c, c, a, d, e, e], please use key-indexed counting sorting	
alg	orith	m to sort the array. Please list your trace step-by-step.(5 pts.)	



8. Please use Shell Sort Algorithm (without any performance improvement techniques, e.g. Augmented by insertion sort) to sort the following array.

$$A = [5, 6, 7, 8, 9, 10, 4, 3, 2, 1]$$
 (6 pts.)

9. Please state the known best lower bound for the 3 Sum problem (5 pts.).

10.	Please implement Bubble Sort Algorithm (using pseudocode) and indicate whether the Bubble sort is stable or not.(5 pts.)
11.	Please design an algorithm that, given an array $a[]$ of N distinct integers, find a local minimum: an index i such that $a[i-1]>a[i]$, $a[i]< a[i+1]$. Your algorithm should guarantee a performance of $O(lgN)$ (15 pts.)

12. Given the following array A = [9, 3, 5, 1, 2, 4, 6, 7, 11, 10, 8, 0]. Please give a O(logN) algorithm (on average) for finding the median in A. (9 pts.)

13. What is the divide-and-conquer strategy for algorithm design? Please state an example algorithm that is based on the divide-and-conquer strategy for solving problems. (5 pts.)

