CS 1332 Exam 3 - not all topics are represented here. See the list of major topics uploaded as a T-Square Announcement. This is only a sample!

November 19, 2014

1. (5 points) This sorting algorithm assumes the first element is sorted. It then slides the second element up if it is smaller than the first. Now the first two elements are considered sorted. The third element now slides up if it is smaller than the second, and if so possibly even past the first element (if it is smaller than that one also). Now the first three elements are considered sorted. This process continues until all elements of the array are sorted. (Assume it is sorting into increasing order.)
   (a) selection sort
   (b) insertion sort
   (c) radix
   (d) merge sort

2. (5 points) Initially, this sorting algorithm sorts by the least significant digit. Collecting together the results from that sort, the data is put through the next sorting - a sort by the next to the least significant digit. This process of sorting again and again using a more and more significant digit continues until the final sort which sorts by the most significant digit. It is important that the steps proceed as required for a stable sort. This algorithm is useful when the data has a required format like a 5 digit zip code or a 9 digit social security number.
   (a) selection sort
   (b) shell sort
   (c) radix
   (d) bubble sort

3. (5 points) This sort might sort all the individual groups of elements of the array found at an index distance of 16 apart. It then starts over, working through the array again, sorting elements found at an index distance of 8 apart. It then starts over, working through the data again, sorting data found at an index distance of 4 apart. And so on. Which sort is this? (Use the process of elimination on this question if necessary.)
   (a) pancake sort
   (b) shell sort
   (c) cocktail shaker sort
   (d) bogo sort

4. (5 points) This sorting algorithm finds the minimum and places it in the first position. It then finds the second minimum and places it in second position, etc. (Assume it is sorting into increasing order.)
   (a) selection sort
   (b) insertion sort
   (c) bucket sort
   (d) merge sort

5. (5 points) This sorting algorithm compares the first and second elements. It swaps them if needed. It then compares the second with the third, swapping if needed, etc. (Assume it is sorting into increasing order.)
   (a) selection sort
   (b) shell sort
   (c) bucket sort
   (d) bubble sort
6. (5 points) What makes a sort stable versus not stable?

7. (5 points) What makes a sort in-place versus not in-place?

8. (5 points) Quick sort. You are given the following array of values. Assume 55 is chosen as the pivot. Perform a stable version of the partitioning step. Complete only the first partitioning of the array showing where the pivot is placed and all the other elements. (Do not finish the sorting.) Be sure you show all of the numbers. Assume you are sorting into ascending (increasing) order.

```
55      33      77      22      44      66      11
```

9. (5 points) Quick sort. Show the first partitioning that would result from performing an in-place partitioning of the following array using the 55 value as the pivot. Assume you are sorting into ascending (increasing) order. This one will not be "stable". Always start your scanning from the righthand side.

   At the end of that partitioning, be sure you show all of the numbers (including the 55).

```
55      33      77      22      44      66      11
```

10. (10 points) Write a static version of the merge method of mergeSort assuming it takes two presorted arrays of ints as parameters. It returns a new array with all the ints. You may assume that the two arrays passed in will have been instantiated.

 Points earned: _____ out of a possible 0 points. Graded by: _____

11. Matching. Place the number of the answer that BEST matches. Choices can be used once, more than once, or not at all. In all cases, n refers to the number of data items.

1. O(1)	4. O(log n)
2. $O(2^n)$	5. $O(n^2)$
3. O(n)	6. O(n log n)

    (a) (3 points) _____ Big O of quick sort when the data is already sorted and the middle index's element is used as the pivot.

    (b) (3 points) _____ Big O of best case of insertion sort.

    (c) (3 points) _____ Big O of quick sort if the data is already sorted in decreasing order, but you want increasing order, and the first element is used as the pivot.

    (d) (3 points) _____ Big O of mergesort.

    (e) (3 points) _____ Big O of selection sort if the data is already sorted the way you want.

    (f) (3 points) _____ Big O of quick sort when the data is already sorted and the first element is used as the pivot.

    (g) (3 points) _____ Big O of worst case kth selection among n elements. (Step one is to know what this is, and how it is done!.)

12. (10 points) Show the complete breakdown of how mergesort would make recursive calls and subsequent merges for these numbers:

    50 70 90 30 32 77 15 99

13. (15 points) Imagine you already have a static min method that takes an int array and two int indices as parameters. The min method returns the index of the first minimum found within the array within those indices (inclusive). Using this extremely helpful helper method, code selectionSort. You method is to be static and have an array of ints as it's only parameter. Your method performs selection sort (in-place) and does not return anything.

14. (10 points) Given a string and a pattern, fill in the dynamic programming-based matrix to show the process of calculating the length of the longest common subsequence.

15. (10 points) Explain the algorithm for LCS.

16. (10 points) Performing brute-force (a.k.a. naive) pattern matching exactly as explained in this course's textbook and slides, give the exact number of character comparisons when searching for "abacab" within "abacaabaccabacabaabb".

17. (10 points) Performing KMP exactly as explained in this course's textbook and slides, give the exact number of character comparisons when searching for "abacab" within "abacaabaccabacabaabb".

18. (10 points) Performing Boyer-Moore exactly as explained in this course's textbook and slides, give the exact number of character comparisons when searching for "abacab" within "abacaabadcabacabaabb".

19. (10 points) Given the pattern "abacab" write the array representing KMP's failure function (exactly as done in the book).

20. (10 points) Given the pattern "abacab" write the array representing BoyerMoore's last occurrence function (exactly as done in the book).

21. (5 points) What data structure is typically used to control a breadth-first search of a graph?

22. (5 points) What is a typical convenient characteristic of a path found using breadth-first search as apposed to depth-first search?

23. (10 points) Compare and contrast Prim's MST algorithm with Kruskal's MST algorithm.

24. (10 points) Given a directed graph, fill in the matrix that represents the process of finding all pairs, shortest paths. (Suitable graph coming soon.)

 Points earned: _____ out of a possible 0 points. Graded by: _____

25. Be sure you are familiar with the Big O of all typical operations for all the data structures we have studied. Adding, removing, searching would be just the start for each and every data structure we study. If other operations are asked about, you should be familiar enough with Big O to surmise an answer even if it is a novel operation we have not literally talked about. This should not be a matter of memorization!

26. Be sure you can perform the typical operations by hand for all the data structures we have studied. Adding (with or without balancing as appropriate for the data structure of interest), removing, searching of any data structure we have studied.

27. Less likely, but still possible, if an operation is described involving a data structure, you might be expected to show how it would work - even if it is not a classic operation that we have studied.

More questions may be added.

 Points earned: _____ out of a possible 0 points. Graded by: _____