CSCI 200

HW #6: Recap

Directions: *Complete your work on a separate sheet of paper. Submit the physical copy of your work at the beginning of class on the specified due date. Show your work. You may work in groups of up to 3 students provided that all students participate in each question. Provide a short preliminary explanation of how an algorithm works before running an algorithm or presenting a formal algorithm description, and use examples or diagrams if they are needed to make your presentation clear.*

1. Give the **pseudocode** for a $\Theta(n)$-time non-recursive algorithm that reverses a singly linked list of n elements. The algorithm should run in-place, that is, it should use no more than constant storage beyond that needed for the list itself.

2. Use the table below to convert a character key to an integer. For each of the following questions, give the contents of the hash table that results when the following keys are inserted into an initially empty 13-item hash table, which has indices 0 to 12. The keys to insert, in this order, are: $(G_1, O_1, O_2, D, J, O_3, B, T, E, A, M)$. Use $h(k) = k \mod 13$ for the hash function for the k-th letter of the alphabet (see above table for converting letter keys to integer values). Use the provided method to resolve collisions. Note that the elements to insert into the hashtable are letters. not numbers.

Letter	A	B	C	D	E	F	G	H	I	J	K	L	M
Key	0	1	2	3	4	5	6	7	8	9	10	11	12
Letter	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Key	13	14	15	16	17	18	19	20	21	22	23	24	25

   (a) Illustrate the contents of the hashtable when collisions are resolved using **chaining.**

   (b) Illustrate the contents of the hashtable when collisions are resolved using **linear probing.**

   (c) Illustrate the contents of the hashtable when collisions are resolved using **double hashing**. Let the secondary hash function be $h'(k) = 1 + (k \mod 11)$.

3. Let $A[1..n]$ be an array of n distinct numbers. If $i < j$ and $A[i] > A[j]$, then the pair (i, j) is called an **inversion** of A.

   (a) List the five inversions of the array $[2, 3, 8, 6, 1]$.

   (b) Describe an algorithm that determines the number of inversions in any permutation of n elements in $O(n \log n)$ worst-case time.

4. Specify in O-notation the worst-case run time of the following algorithms:

   (a) Mergesort n elements

   (b) Quicksort n elements, using the last element as partition

   (c) Insertion sort n elements

   (d) Heapsort n elements

   (e) Find an element in a red-black tree with n elements

   (f) Insert into a binary search tree with n elements

   (g) BFS on a graph with n vertices and m edges

   (h) Dijkstra's shortest path algorithm on a graph with n vertices and m edges

   (i) Prim's MST algorithm on a graph with n vertices and m edges

   (j) Kruskal's MST algorithm on a graph with n vertices and m edges

5. Insert into an initially empty binary search tree items with the following keys (in this order): 100, 50, 25, 75, 10, 12, 5, 4, 3, 15, 16, 132, 148, 78. Draw the single resulting tree after all insertions have been performed.

6. Rank the following functions by order of growth, from slowest-growing to fastest-growing. That is, find an arrangement $f_1, f_2, f_3, ..., f_{10}$ of the following functions satisfying $f_1 \in O(f_2)$, $f_2 \in O(f_3)$, etc.

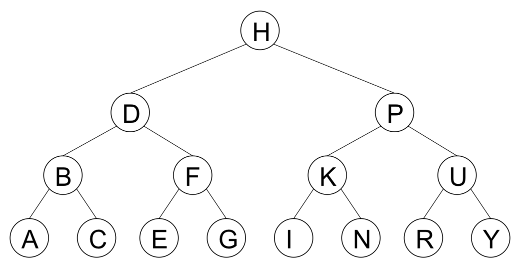
$\log_3^2(n)$	$n \log_2(n)$	2^n	n^2	$n!$
42	n^{15}	$5^{\log_5(n)}$	$n^{0.08}$	$\log_2(\log_2(n))$

7. For each of the following recurrence equations which describe the running time $T(n)$ of a recursive algorithm, express the asymptotic complexity (use the master theorem where appropriate).

   (a) $T(n) = T(2n/5) + n$
   (b) $T(n) = 27T(n/3) + \log n$
   (c) $T(n) = 2T(n/2) + n^3$
   (d) $T(n) = 4T(n/2) + n^2 \log n$
   (e) $T(n) = 2T(n) + 1$

8. As opposed to the single-source shortest path problem which seeks the shortest path from a single vertex to every vertex, the *single-destination shortest path problem* for a directed graph seeks the shortest path *from every vertex to a specified vertex* v. Describe an efficient algorithm which runs in $O(m \log n)$ time to solve the single-destination shortest paths problem on a connected digraph with positive edge weights. Analyze the running time of your algorithm to justify that it runs in $O(m \log n)$ time.

9. Consider the following binary search tree.



   (a) List the vertices following a preorder traversal.
   (b) List the vertices following a postorder traversal.
   (c) List the vertices following an inorder traversal.

10. Find the minimum spanning tree (MST) of the following graph.