

Name: \_\_\_\_\_

SSN: \_\_\_\_\_

## CS 140A - Final December 13, 1997

Be sure to read each problem carefully and follow the directions. Points may be marked off if you do not follow the directions. When you are asked to write code, you may write pseudo-code if you desire. Be sure to state any assumptions you make and be sure the code is clear. Please feel free to ask if you have any questions.

Problem 1	24	
Problem 2	6	
Problem 3	30	
Problem 4	12	
Problem 5	28	
Problem 6	40	
Problem 7	9	
Problem 8	5	
Problem 9	5	
Problem 10	10	
Problem 11	5	
Problem 12	8	
Problem 13	6	
Problem 14	8	
Problem 15	8	
Problem 16	8	
Problem 17	15	
Problem 18	6	
Problem 19	6	
Problem 20	6	
TOTAL	245	



1. (24 pts) True/False (please write out “True” and “False”, do not just put a T or F).

- (a) A binary tree that is full, is a complete binary tree.
- (b) If a function runs in  $o(N)$ , then it runs in  $O(N)$ .
- (c) When using hash tables, a larger table size will always increase performance.
- (d) Recurrence relations are only used to calculate the running times of recursive algorithms.
- (e) In a Red-Black Tree, if a node is red, its parent could not be red.
- (f) In a Red-Black Tree with  $N$  nodes, if  $N > 1$  there will always be at least one red node.
- (g) Heap Sort is a stable sorting algorithm.
- (h) Shell Sort typically runs faster than Insertion Sort.
- (i) Quick Sort always runs faster than Selection Sort.
- (j) Selection Sort is a stable sorting algorithm.
- (k) Insertion Sort works in  $O(N)$  if the array is already sorted.
- (l) Selection Sort works in  $O(N)$  if the array is already sorted.

2. (6 pts) List 3 factors that affect the running time of a program.

3. (30 pts) Define the following terms.

(a) Algorithm

(b) Recursion

(c) Linked List

(d) Sentinel

(e) Binary Tree

(f) Path (of a tree)

(g) Priority Queue

(h) Load Factor

(i) Open Address Hashing

(j) Primary Clustering

4. (12 pts) Place a 'Y' or 'N' (Yes/No) in row  $i$  and column  $j$  if the two conditions represented by row  $i$  and column  $j$  can occur simultaneously.

	$\text{preorder}(n) < \text{preorder}(m)$	$\text{inorder}(n) < \text{inorder}(m)$	$\text{postorder}(n) < \text{postorder}(m)$
n is in the left subtree of m			
n is in the right subtree of m			
n is a proper ancestor of m			
n is a proper descendant of m			

5. (28 pts) Next to each real-life application, write in the type of ADT (covered in class) that would be appropriate to use in the given situation. Give a very brief explanation of why that would be a good ADT to use.
- (a) A program is needed to take care of aiding the patients that enter an emergency room in a hospital. The type of injury should be a factor in who gets helped first.
  
  - (b) A library needs a way to find their books quickly and easily. The library is always adding new books to the collection, but typically does not throw any books away. This is a fairly large library.
  
  - (c) A Phone Company desires to have a program to keep track of their customers information, such as their name, account number, and balance.
  
  - (d) A university would like to write a web page that allowed a student with a unique id to check out the status of their grade.
  
  - (e) Write a program that plots the path of a person as they visit certain destinations and then enforces that they return on the same path.
  
  - (f) A large company would like to maintain a record of employees using the hierarchy of the company, including the President, the Vice Presidents, the Supervisors, ...
  
  - (g) Write a program for a Pharmacy that keeps track of the patients as they present their prescription, wait for it to be filled, and then receive their prescription.

6. (40 pts) Write the running time in Big-Oh notation for each algorithm.
- (a) Insert at head, array implementation of a list.
  - (b) Insert at head, linked list implementation of a list.
  - (c) Print, linked list implementation of a list.
  - (d) Push, array implementation of a stack.
  - (e) Push, linked list implementation of a stack.
  - (f) Dequeue, array implementation of a queue.
  - (g) Dequeue, linked list implementation of a queue.
  - (h) Search, full binary search tree.
  - (i) Remove, binary search tree (average case).
  - (j) Insert, binary search tree (worst-case).
  - (k) Insert, red-black tree.
  - (l) Insert, 2-3 tree.
  - (m) Search, hash table using linear probing (average case).
  - (n) Search, hash table using separate chaining (average case).
  - (o) Enqueue, priority queue.
  - (p) Dequeue, priority queue.
  - (q) Heap Sort.
  - (r) Bubble Sort.
  - (s) Insertion Sort.
  - (t) Quick Sort.

7. (9 pts) Write out the entire function used to calculate the hash value using:

(a) linear probing

(b) quadratic probing

(c) double hashing

8. (5 pts) Describe what makes a good hash function. Give an example of a good hash function and a bad hash function.

9. (5 pts) Describe what are typically good hash table sizes.

10. (10 pts) Show the resulting binary search tree if the following numbers were inserted. {20, 9, 50, 4, 14, 12, 46, 16, 31}

Now answer the following questions.

- (a) The parent of node 46 is?
  - (b) The depth of node 4 is?
  - (c) The height of node 50 is?
  - (d) The height of the tree is?
  - (e) The ancestors of node 14 are?
11. (5 pts) Convert the following expression into postfix notation.  
 $((a + b) * (c + d)) - (e * f)$

12. (8 pts) Show the resulting max heap if the following numbers were enqueued in the given order. {5, 12, 11, 17, 3, 18, 23, 4, 21}

Show the result if a Dequeue were performed.

13. (6 pts) Show the resulting 2-3 tree if the following numbers were inserted.  
{14, 30, 15, 4, 13, 22, 8}

14. (8 pts) Write a recursive function to print a linked list backwards.

15. (8 pts) Write a recursive function to Search for all occurrences of an item in a linked list. The function should return the number of occurrences.

16. (8 pts) Using an appropriate ADT, write an efficient function to determine if a string is a palindrome. A palindrome is a word that is spelled the same both backwards and forwards (mom, stats, wow). You do not need to write the ADT, you can just use its basic operations. You will not receive full credit unless you use a necessary and appropriate ADT. What is the running time of your algorithm?

17. (15 pts) Write a recursive function to print a binary search tree using inorder traversal. Write a recurrence relation to describe the function if the tree were full. Solve the recurrence relation to show the Big-Oh notation.

18. (6 pts) Sort the following numbers using Selection Sort. Show the resulting array at each pass. {24, 16, 2, 5, 13, 20, 9, 6, 1}

19. (6 pts) Sort the following numbers using Heap Sort. Be sure to use the linear time algorithm when building the heap. Show the resulting array at each pass. {24, 16, 2, 5, 13, 20, 9, 6, 1}

20. (6 pts) Sort the following numbers using Quick Sort using the median of three partition. State the pivot and show the resulting array at each pass.  
{24, 16, 2, 5, 13, 20, 9, 6, 1}