CS 240 Fall 2014 Exam 3 **Name: _____**

1. [4 pts] Suppose I have one very large set of integers to sort, and two different sorting algorithms. One algorithm has a tight bound runtime complexity of $O(N^2)$ and the other has a tight bound runtime complexity of $O(N \log N)$. *(As described in class, by tight bound, I mean that there is no other function g(n) that grows slower and for which the algorithm is also O(g(n))*. Which of the following best describes the runtime of two C++ functions that correctly and faithfully implement the two algorithms? (Circle the statement below that is most correct.)

(a) The $O(N^2)$ function will run faster because N^2 grows faster than N log N.

(b) The O(N log N) function will run faster because N^2 grows faster than N log N.

(c) We can't tell which function will run faster because the "hidden constants" could influence the runtime of the functions, even for "very large" arrays.

(d) The two functions will finish in almost exactly the same amount of time, because they are operating on the same array.

2. [2 pts] What is the runtime complexity of a typical hash function? (circle one)

O(N) O(logN) O(1) $O(N^2)$ O(NlogN)

3. [2 pts] What is the input parameter to a hash function?

4. [2 pts] What does a hash function return?

5. [2 pts] Devise two different hash functions that will help store integers in a hash table of size 100.

1.

2.

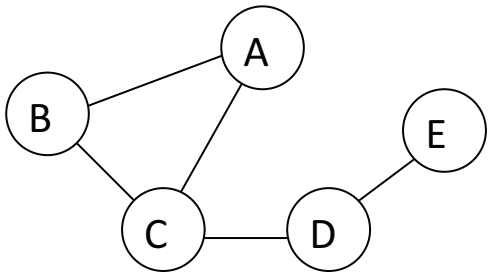
6. [3 pts] Explain the difference between a *chained hash table* ("chaining") and an *open hash table* "open addressing" (or "probing") in resolving collisions.

7. [4 pts] Suppose facebook were implemented as a graph with one vertex per user, and an edge in the graph between two people if they are facebook friends. Do you think this graph should be implemented using (circle one):

(a) An adjacency list
(b) An adjacency matrix
(c) It doesn't matter

Explain your answer:

8. [3 pts] Consider the following graph:



In what order will a *depth first search* starting from D visit the nodes? (assume that nodes are visited in alphabetical order if a node must decide between multiple nodes within the confines of the algorithm)

9. Consider the following algorithm for Breadth First Traversal of a graph, which is taken directly from the slides.

```
breadthFirstTraversal (v)
   put v in Q
   mark v as visited
   while Q not empty
      remove w from Q
      visit w
      for each successor (u) of w
          if not yet visited
              put u in Q
              mark u as visited
```

(9 a) [3 pts] What kind of container is "Q"?

(a) A Heap
(b) A First in First Out queue (for example, a doubly linked list with a tail pointer)
(c) A Sorted Array
(d) A Binary Search Tree (BST)

(9 b) [3 pts] In what order should the successors (the ***u***'s) of ***w*** be processed?

(a) Smallest to largest edge weight
(b) Largest to smallest edge weight
(c) Doesn't matter
(d) Smallest to largest distance from the initial starting point for the BF traversal (v).

(9 c). [4 pts] What if we used a Stack (i.e. last in first out) for Q, and implemented "put x in Q" as push(x), and "remove x from Q" as pop(x). What kind of traversal order would the new function represent? Explain your answer.

10. [4 pts] What is the most accurate description of the problem that Dijkstra's algorithm solves:

(a) It finds the shortest paths from every node to every other node.

(b) It finds the shortest paths from every node to a specified destination node.

(c) It finds the shortest path from a specified source node to a specified destination node.

(d) It finds the shortest path from a specified source node to every other node.

11. [4 pts] Consider arrays d[] and p[] (for distances and predecessors) in Dijkstra's Algorithm. The algorithm extracts from a Heap the closest node to the source, and then "relaxes" each outgoing edge from that node. Fill in the blanks below to complete correct pseudocode for the relax operation, assuming that x is the node extracted from the Heap, y is the node connected to x by an edge in the graph, and w(x, y) is the weight of that edge.

If (d[v] > d[u] + w(u, v))

d[_____] = _____

p[_____] = _____

12. [4 pts] We could say that a relax operation "succeeds" if the d[] and p[] values are changed; that is, the condition in the if statement (d[v] > d[u] + w(u, v)) is true. If Djikstra's algorithm runs on a connected graph with V vertices and positive weight edges, then what is the minimum number of times that the relax operation "succeeds" in updating a shortest path?

(a) V^2

(b) Log V

(c) V – 1

(d) 0

(e) 7

13. [4 pts] Draw a disconnected weighted digraph with five vertices, seven edges, and one cycle of total weight 12.

14. [4 pts] What is "minimal" about a minimum spanning tree (MST)? (Circle one letter)

(a) The shortest path between any two vertices in the tree

(b) The longest path between any two vertices in the tree

(c) The number of edges in the tree

(d) The sum total of all the weights of the edges in the tree

(e) The number of vertices in the tree

(f) The number of vertices plus edges in the tree

(g) The runtime complexity ("Big Oh") of the traversal algorithm used to visit all vertices in the tree

15. [4 pts] Kruskal's Algorithm grows a Minimum Spanning Tree (MST) by considering edges in order from smallest to largest, no matter where they are in the graph. In considering some edge *e*, describe the condition under which *e* is added to the MST. In other words, what must be true at the time that *e* is considered for it to be added to the MST?

16. [8 pts] Answer True or False for the following:

_____ A Heap is also a binary search tree (BST).

_____ A Heap is also a "complete" tree.

_____ A pre-order traversal of a Heap can be used to print the contents of a Min Heap in sorted order.

_____ A Heap can be implemented using a C++ array.

_____ Given a Heap that contains N elements, for N > 3, there is more than one organization of the elements that satisfies all the properties of a Min Heap.

_____ The *tight bound* runtime complexity for Heap Sort is $O(N^2)$. (Tight bound means there is no function g(n) that "grows slower than" N^2 and for which Heap Sort is also O(g(n)).

_____ Heapify cannot run "in place"; it requires additional memory/storage "outside" of the Heap.

_____ ExtractMin() on a Min Heap is a constant time (O(1)) operation.

17. [4 pts] Consider some internal node of a Max Heap that contains all unique values (i.e. no duplicates). If the value stored within Node A is 100, which of the following do we know for sure about the left and right children of A? Circle the letter corresponding to all true statements:

(a) A's left child stores a value that is less than 100.

(b) A's right child stores a value that is less than 100.

(c) A's right child stores a value that is larger that A's left child.

(d) If you travel "all the way" down the right subtree of A, you will find a value that is larger than the value you will find if you travel all the way down the left subtree of A.

18. [4 pts] What is the runtime complexity of ***find(x)*** on a Min Heap, where ***x*** is any value provided by the user. The function returns 1 if **x** is in the Heap, 0 if it is not. Explain your answer. (2 pts for the correct answer, 2 pts for the explanation).

19. [6 pts] If a Node is in C++ array location 37 in a Heap, where are its children, and where is its parent?

Index of left child (2 pts): _____

Index of right child (2 pts): _____

Index of parent (2 pts): _____

20. [4 pts] In the context of balanced trees, circle the letter corresponding to all of the True statements below.

(a) Given a pointer to a node whose child needs "to be rotated", the rotation operation is a constant time (i.e. O(1)) operation

(b) Rotations can only take place at the "bottom" of a tree, where the left and right pointers of a node to be rotated are leaf nodes

(c) Given a pointer to a node to whose child needs to be rotated, calculating new balancing factors for a node in an AVL tree, in order to perform a rotation at that node, is a O(log N) operation, where N is the number of nodes in the tree

(d) A tree can have a root node with a balancing factor of 0 (correctly calculated as it is supposed to be for AVL trees), and yet that tree can still be unbalanced.

21. [3 pts] Draw a small tree that stores letters (i.e. characters) in alphabetical order, and requires a single right rotation. Show the letters contained in the tree and the balancing factors.

22. [3 pts] Show what your tree from the previous question would look like after the rotation, again including letters and balancing factors.

23. [3 pts] Which of the following most accurately describes how a balancing factor is used in an AVL tree. Circle the letter for one of the following answers.

(a) One balancing factor per node, storing the difference in heights between the left and right subtrees

(b) One balancing factor per tree, storing the largest difference in subtree heights across all nodes in the tree

(c) One balancing factor per tree, storing the difference between the expected height of the tree for the number of elements it has, and the actual height of the tree

(d) One balancing factor per node, storing the height of the subtree of which it is the root.

24. [3 pts] What will be the value(s) of the balancing factor(s) of a balanced AVL tree? Explain your answer.

25. [3 pts] Which of the following best finishes the description of the Activity Selection problem that we discussed in class?

Given a resource (e.g. a room) and a list of activities (i.e. requests to use that resource for intervals of time),

(a) Find a set of activities that do not interfere with one another and that keeps the resource occupied for the *largest percentage* of time.
(b) find a set of activities that do not interfere with one another and that keeps the resource occupied for the *smallest percentage* of time.
(c) find a *smallest* set (i.e. fewest elements in the set) of activities that can use the resource without interfering with one another.
(d) find a *largest* set (i.e. most elements in the set) of activities that can use the resource without interfering with one another.
(e) find a *largest* set of activities that can use the resource at the same time.
(f) Find a *smallest* set of activities that can use the resource at the same time.

26. [3 pts] Briefly describe how to *solve* the Activity Selection problem.