

CSci 161: Computer Science II

Final exam, Fall 2005

Name: _____

1. Suppose a rooted binary tree contains exactly 50 nodes.

How many nodes in this tree can have no parent? _____

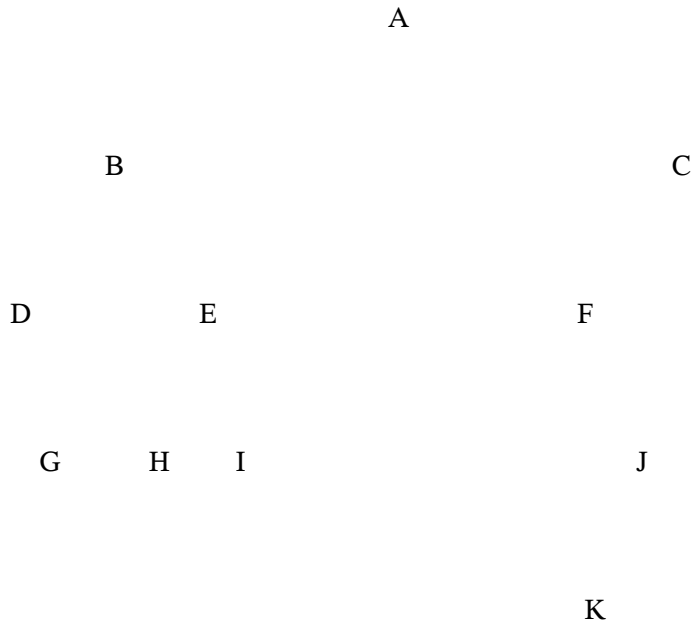
How many nodes in this tree can have more than one parent? _____

What is the smallest possible height this tree could have? _____
(height = maximum distance from root to a leaf)

What is the smallest number of leaves this tree could have? _____

What is the largest number of leaves this tree could have? _____

2. Show the order in which the nodes of the following binary tree are visited by inorder, preorder, postorder and level-order traversals.



(a) preorder traversal: _____

(b) inorder traversal: _____

(c) postorder traversal: _____

(d) level-order traversal: _____

3. (a) The numbers 47, 15, 63, 84, 31, 24, 91, 39, 16 and 76 are to be inserted into an initially-empty binary search tree in the order listed. Draw a picture of the tree after these numbers have all been inserted.
- (b) The numbers 24 and 47 are to be deleted from the tree constructed in part (a). Draw a picture of the tree after these two deletions have been done.
4. (a) Write a complete Java class definition for a binary tree node.
- (b) Write a part of a Java class definition for a binary tree, specifying the class' attributes and defining a class constructor.
- (c) Write a `public boolean isEmpty()` method for this binary tree class.

7. A Priority Queue is an abstract data type (ADT) which can be implemented in many different ways. One particularly efficient implementation uses the heap data structure, but other implementations are possible.

Describe abstractly the nature of the data that is stored in a Priority Queue and describe the fundamental operations that can be performed on a Priority Queue (what they do, what values do they return, etc.). Do not describe the details of any particular implementation ... just describe the ADT. You can write your answer in English, or as a Java interface with embedded comments.

8. (a) Write a part of a Java class definition for a heap (an implicitly-represented heap-ordered complete binary tree), specifying the class' attributes and defining a class constructor.

(b) Write a `private int left_child(int n)` method for this heap class that computes the index of the left child of the node with index `n`. If the node has no left child, the value `-1` should be returned.

(c) Write a `public int sizeOf()` method for this heap class, which returns the number of items in the heap.

9. A Dictionary is an abstract data type which can be implemented in many different ways. Some particularly efficient implementations use binary search trees and hash tables, but other implementations are possible.

Describe abstractly the nature of the data that is stored in a Dictionary and describe the fundamental operations that are performed on a Dictionary (what they do, what values do they return, etc.). Do not describe the details of any particular implementation ... just describe the ADT. You can write your answer in English, or as a Java interface with embedded comments.

10. (a) What look-up speed are we hoping for when we implement the Dictionary ADT as a hash table?

(b) What look-up speed are we hoping for when we implement the Dictionary ADT as a binary search tree?

(c) What look-up speed should we expect when we implement the Dictionary ADT as an ordered linked list?

11. (a) Fill in the hash table below, showing how to insert the 7 integers 57, 39, 16, 49, 23, 69 and 46, in this order, into an initially empty hash table with maximum capacity 11. Use division modulo 11 as the hash function, and use a linear probe to handle collisions.

0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	

- (b) Suppose that each entry in the above hash table is looked up one time, What is the average number of items that need to be peeked at when doing these lookups?
12. (a) Write a part of a Java class definition for a hash table containing integer entries, specifying the class' attributes and defining a class constructor.
- (b) Write a `private int hashvalue(int key)` method which computes a hash value for an integer key by computing `key%table_size`. The computed hash value is the value returned.
- (c) Write a `public void insert(int x)` method that inserts an integer x into the hash table, using a linear probe if necessary to handle collisions. Note: the integer x is the key of the item to be inserted and, for simplicity, we will assume there is no associated data.

13. Describe the data structures needed to represent a graph or digraph using the adjacency matrix technique.

14. Describe the data structures needed to represent a graph or digraph using the adjacency list technique.

15. (a) What is the advantage of using the adjacency matrix technique to represent a graph?

(b) What is the advantage of using the adjacency list technique to represent a graph?

16. (a) Write a part of a Java class definition for a directed graph, using the adjacency matrix strategy to store information about the graph. You should specify all the class' attributes and define a class constructor.

(b) Write a `public boolean isEmpty()` method for this class.

(c) Write a `private int indexOf(Comparable x)` method for this class, which finds and returns the index of a particular vertex in the vertex array. If the item is not found in the vertex array, the value -1 should be returned.