

CSci 161: Computer Science II

Final exam, Fall 2006

Name: _____

This test has 20 questions of equal value. If a question gives you trouble, it's a good idea to move on to the next question and come back to it later.

1. *Understanding Recursion.*

There are three important conditions that every correct recursion must satisfy. List them below:

1

2

3

2. *Recursive Programming.*

Code a recursive Java method to compute 2^n for any positive integer n , by calculating $2^{n/2} * 2^{n/2}$ (if n is even) or $2 * 2^{n/2} * 2^{n/2}$ (if n is odd, when $n/2$ represents integer division)

```
int powerOf2 (int n) {
```

```
}
```

3. *Answer these questions for the case of a binary tree with exactly 40 nodes.*

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

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

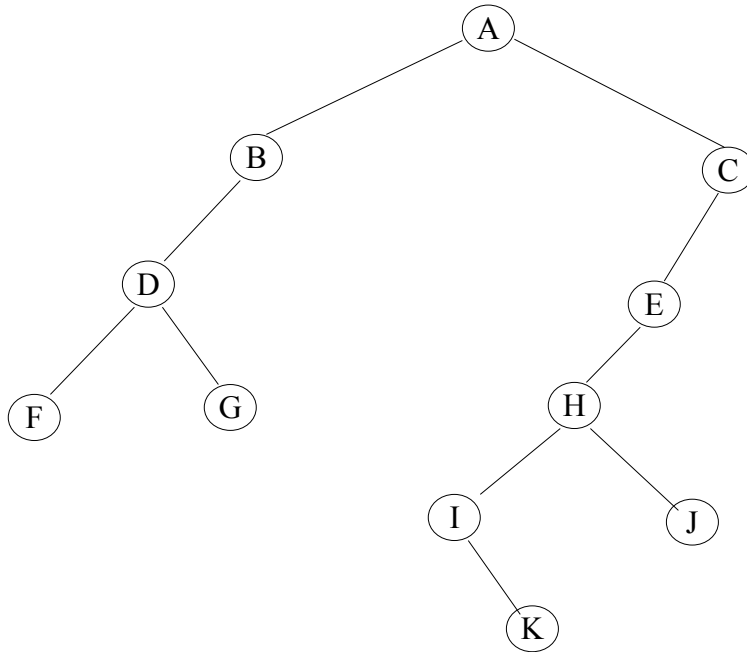
What is the smallest possible height this tree could have? _____

What is the largest possible height this tree could have? _____

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

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

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



preorder traversal: _____

inorder traversal: _____

postorder traversal: _____

level-order traversal: _____

5. Write a complete Java class definition for a binary tree node. Make the fields of this class public so that you won't need accessor or mutator methods.

```
public class BinaryTreeNode {
```

```
}
```

6. Write a portion of the Java class definition for a binary tree, specifying the class' attributes and defining a class constructor.

```
public class BinaryTree {  
  
  
  
  
  
  
  
  
  
}
```

Code an isEmpty() method for this binary tree class.

```
public boolean isEmpty() {  
  
  
  
}
```

7. 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.

Your tree should have 10 nodes when you are finished.

8. The numbers 24 and 47 are to be deleted from the tree constructed above. Draw a picture of the tree after these two deletions have been done.

11. 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	

12. 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?

13. Write a part of a Java class definition for a hash table containing integer entries. Declare the class' fields and define a class constructor. All data to be stored in the hash table will be positive integers, so you can use a negative value or zero to identify an empty location in the hash table.

```
public class HashTable {
```

```
}
```

Code an insert() method that inserts an integer x into this hash table. Use a linear probe to handle collisions. The hash value for a particular integer x will be $x \% \text{table_size}$, where table_size is the maximum capacity of the table.

```
public void insert(int x) {
```

```
}
```

14. Write a part of a Java class definition for a directed graph that uses an adjacency matrix to store information about the graph's edges. You should specify all the class' attributes and define a class constructor.

```
public class DirectedGraph {
```

```
}
```

Write an isEmpty() method for this class.

```
public boolean isEmpty() {
```

```
}
```

Write an indexOf() 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.

```
private int indexOf(Comparable x) {
```

```
}
```