COMPUTER SCIENCE 112 - SPRING 2013
SECOND MIDTERM EXAM

Name: ______________________________

CIRCLE your recitation: T 5:15 T 6:55 W 3:35 W 5:15 W 6:55 Th 6:55

- Be sure your test has 4 questions.
- **DO NOT TEAR OFF THE SCRATCH PAGES OR REMOVE THE STAPLE.**
- Be sure to fill your name and circle your recitation time above, and your name in all subsequent pages where indicated.
- This is a CLOSED TEXT and CLOSED NOTES exam. You MAY NOT use calculators, cellphones, or any other electronic device during the exam.

Do not write below this line

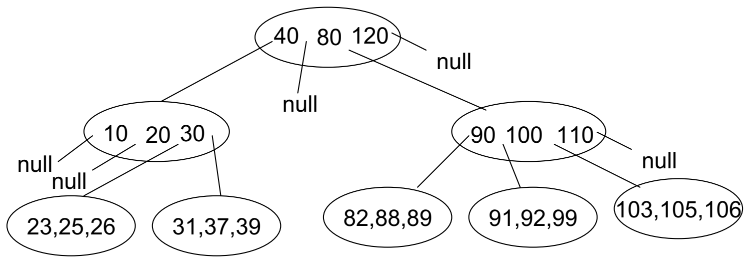
Problem	Max	Score
1 M-ary Tree	20	______
2 Expression Tree	20	______
3 Binary Search Tree	20	______
4 Character Frequencies	20	______
TOTAL	80	______

M-ary Tree (20 pts, 5+6+4+5)

a) An m-ary tree is a tree in which every node has at most m children. Consider a special m-ary tree in which a node either has exactly m children or no children. What is the minimum number of nodes in a special m-ary tree of height h? (A single node tree has height 0). Show your work, and show how your answer works out correctly by drawing a special tree with $h = 3$ and $m = 3$ (and any set of values at the nodes).

b) Suppose each node of the m-ary tree stores $(m-1)$ keys, $k_1, k_2, \ldots k_{m-1}$ in **ascending order**. All keys k such that $k < k_1$ are stored in the first subtree of this node, all keys k such that $k_1 < k < k_2$ are stored in the second subtree, and so on. All keys k such that $k > k_{m-1}$ are stored in the m-th subtree.

Here's an example 4-ary tree, with 3 keys in each node (although not shown, assume that all pointers in the leaf nodes are null):



A search for a target key, k_t, proceeds as follows (recursive):
If root is null, return false. Otherwise, do a sequential search of k_t against the keys $k_i, 1 \leq i \leq m-1$, at the root. If a match is found, return true. Otherwise, if $k_t < k_i$ for some i, then recursively search the i-th subtree and return the result. Otherwise, it must be that $k_t > k_{m-1}$, in which case recursively search the m-th subtree, and return the result.

If the height of an m-ary tree is h, how many key-to-key comparisons (exact number, **not** big O) would it take in the worst case to successfully search for a target key? Give your answer in terms of h. Show your work.

c) Draw a tree structure with m=5 and h=3, and show a worst case search on it. (You only need to show all keys in the nodes that are hit by the search - the other nodes need not show keys.)

d) Assume a full m-ary tree, i.e. an m-ary tree in which every level has the maximum possible number of nodes. If n is the total number of nodes in a full m-ary tree, what is the formula for its height, h, in terms of n? Show your reasoning by writing down a mathematical pattern from which you can generalize to the result.

2. Expression Tree (20 pts;12+8)

Assume that you are given an expression tree that contains only integer constants and the binary operators '-', '+', '*' and '/'. Here's an example:

```
    +
   / \
  2   *
     / \
    4   55
```

The following is the expression tree node definition:

```
public class ETNode {
     String data;   // either integer constant or binary operator
     ETNode left, right;
     ...
}
```

(a) Complete the following method to evaluate an expression given a reference to the root of its expression tree. You may implement helper methods if you wish.

```
// Evaluates an expression tree given its root, returns 0 if tree is empty
public static double evaluate(ETNode root) {
```

(Things to know: The `Integer.parseInt(String)` method returns the `int` value represented by the string parameter. The `Character.isDigit(char)` method returns true if the char parameter is a digit, false otherwise.)

(b) What is the worst case big O running time of your implementation on an expression that has n integer constants? State the basic unit time operations you are counting, show how many times each of these operations are done during the process, add up the operations to get a number that is a function in n, then convert to big O. Credit will depend on your showing all of the above. If you simply write a big O answer, you will NOT get any credit. (Note: Operations here mean anything that counts towards running time, not just the mathematical operations.)

3. Binary Search Tree (20 pts;10+10)

Suppose each binary search tree node is modified by adding a field that stores the number of nodes in its LEFT subtree. This modification is useful to answer the question: what is the k-th SMALLEST element in the binary search tree? ($k = 1$ means smallest element, $k = 2$ means second smallest element, etc.)

You are given the following enhanced BSTNode class definition:

```
public class BSTNode<T extends Comparable<T>> {
    T data; BSTNode<T> left, right;
    int leftSize;  // number of nodes in left subtree
    ...
}
```

(a) Implement the following **recursive** method to return the k-th smallest element in a given enhanced BST. The method should throw a `NoSuchElementException` if k is out of range. You may implement helper methods if necessary.

```
public static <T extends Comparable<T>>
T kthSmallest(BSTNode<T> root, int k) throws NoSuchElementException {
```

(b) Compute the most and least number of units of running time (not big O) for your implementation, by showing all scenarios of tree shape and value of k for which these occur, if there are n nodes in the tree. State the basic unit time operations you are counting (ignore time to compare a pointer against null), show how many times each of these operations are done for each of the scenarios, then add up the operations to get a number that is a function in n and/or k.

4. Character Frequencies (20 pts, 11+9)

You are given a text file in which you are to find how many times each character occurs (frequency). Assume the file consists of T characters in all, with D distinct characters. You may assume that D is much less than T. (For instance, if the file has 1000 characters, and consists ONLY of lowercase letters, then $T = 1000$ and $D = 26$). Note that characters are case sensitive since, when the encoded message is decoded, it should be identical to the original. (So 'A' and 'a' are two distinct characters.) Answer the following questions with regard to the running times of the various steps in the encoding process.

(a) Suppose a hash table is used to count the number of occurrences of each character (which can then be used to determine probabilities). Derive the worst case AND the expected (assuming uniform distribution of entries over the hash table locations) big O running times to compute the character frequencies. The time needed to read the characters from file into data structures should be counted in your result: assume unit time to read each character from the input file. Your work should be as detailed as possible.

(b) Describe the MOST efficient (in the worst case big O) algorithm, and required data structures to determine the frequencies of characters in an English text document that can have any character on the keyboard, with upper or lower case for letters, and print the (character,frequencies) pairs at the end. If you think the hash table based algorithm above is the most efficient in the worst case, describe why. If not, write your algorithm clearly as a sequence of steps - use pseudocode if necessary.

Analyze your algorithm for the worst case: first decide on the operations you will count, and then determine the actual number of these operations, and then give the resulting big O time. (You can use T and D again as parameters for the input data.)

You will not get ANY credit if your algorithm is not the fastest possible (big O wise).

SCRATCH PAGE

SCRATCH PAGE

SCRATCH PAGE