Binary Trees

A binary tree is made of nodes, where each node contains a "left" pointer, a "right" pointer, and a data element. The "root" pointer points to the top-most node in the tree. The left and right pointers recursively point to smaller "subtrees" on either side. A null root pointer represents a binary tree with no elements - the empty tree.

In C++, you a binary tree is built with nodes of the type:

```c
struct node {
    int data;
    struct node* left;
    struct node* right;
};
```

A "binary search tree" (BST) or "ordered binary tree" is a type of binary tree
where the nodes are arranged in order: for each node, all data elements in its left subtree are less than or equal to the node (\( \leq \)), and all the elements in its right subtree are greater than the node (\( > \)). Note that comparison is made between the data elements in the node.

The tree shown above is a **binary search tree** -- the "root" node is a 10, and its left subtree nodes (5, 4, 7, 8) are \( \leq 10 \), and its right subtree nodes (12, 13, 16) are \( > 10 \). Recursively, each of the subtrees must also obey the binary search tree constraint: in the (5, 4, 7, 8) subtree, the 7 is the root, and its left subtree has values (4, 5) which are less than 7. Watch out for the exact wording in the problems -- a "binary search tree" is different from a "binary tree".
Consider a Binary Search Tree that has \( n \) nodes. In one extreme case, the tree could look like

![Tree Diagram](image)

This "degenerate" tree has \( n-1 \) levels (counting root at level 0).

On the other hand consider a Binary tree that is "tightly packed", with every possible node filled in:

![Tightly Packed Tree Diagram](image)

In this case Level 0 has \( 2^0 \) nodes, Level 1 has \( 2^1 \) nodes, ... , Level \( h \) has \( 2^h \) nodes, so that
\[
2^0 + 2^1 + \ldots + 2^h = n \quad \Rightarrow \quad 2^{h+1} - 1 = n \quad \Rightarrow \quad h = \log_2(n+1) - 1
\]

For \( n=1000 \), we see that in the "tightly packed" case, the number of levels, \( h \) is: \( h=\log_2(1001)-1=8.9672 \). So it is sufficient to have 9 levels to pack 1000 nodes if we use "tight packing".