Objectives

1. Create a Binary Tree, based on input
2. Check the tree to determine if the tree meets Binary Search Tree property.
3. Rinse and repeat for multiple trees till you run out of input. Have fun!

High-level problem description

For this challenge, you are giving a series of binary trees and you are to check if each given binary tree is also a binary search tree. Recall that a binary search tree is a binary tree such that:

1. The value at a node is never less than an entry in its left subtree.
2. The value at a node is less than every entry in its right subtree.

For instance, the tree to the left is in fact a binary search tree.

Inspiration: Note, this problem is inspired by Problem 4.5 from the book Cracking the Code Interview.

Input

You will be given a series of trees specified in Breadth First Search order from standard input:

```
N N_0 N_1 ... N_N-1
```

- The first number of each line is the number of nodes in the binary tree.
- The remaining numbers are the values for each node.
- Any nodes with the value -1 are considered empty nodes (ie. they don't exist).

Output

For each tree, you should output the message:

```
Tree N is a BST
```

if the binary tree is a binary search tree. Otherwise, you should output:

```
Tree N is not a BST
```

Note, N is the tree number in the order in each it was read (starting from 1).
Example

Given the following input:

```
2 20 20
3 20 -1 20
14 8 3 10 1 6 -1 14 -1 4 7 -1 -1 13
```

Your program should output the following:

Tree 1 is a BST
Tree 2 is not a BST
Tree 3 is a BST

Hints

➔ You can easily store the tree using an `array` based representation. For node at index `i`, left child is at `2i+1` and right child is at `2i+2`.

➔ You should consider a `recursive` solution that ensures the [binary search tree] invariant is maintained.

➔ You should try to hand draw the trees described in the input. And check that your tree representation is correct.

Resources

1. Create a file called `bst.cpp` which will house your solution. Grab the following files and place them into your folder with the commands provided:

   ```
   ```

Rubric

+80 Program implemented as intended
+50 Program passes all output tests
+20 Gameplay works. No segfaults.
+5 On receiving ctrl+d, the program does not expect more input.
+5 Good program structure and code style

+5 Program passes memory test
+5 Program passes timing test
+10 Standard lab report - Describe how you represented your trees and checked for bst property.