CS303 Data Structures Assignment #5

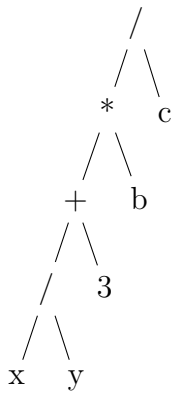
attachments and source available at https://github.com/alexskc/cs303

Aleksander Charatonik

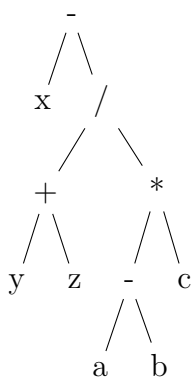
October 30, 2018

1

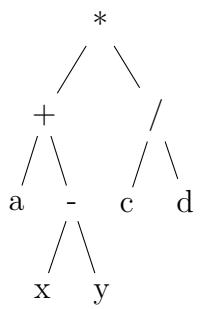
a



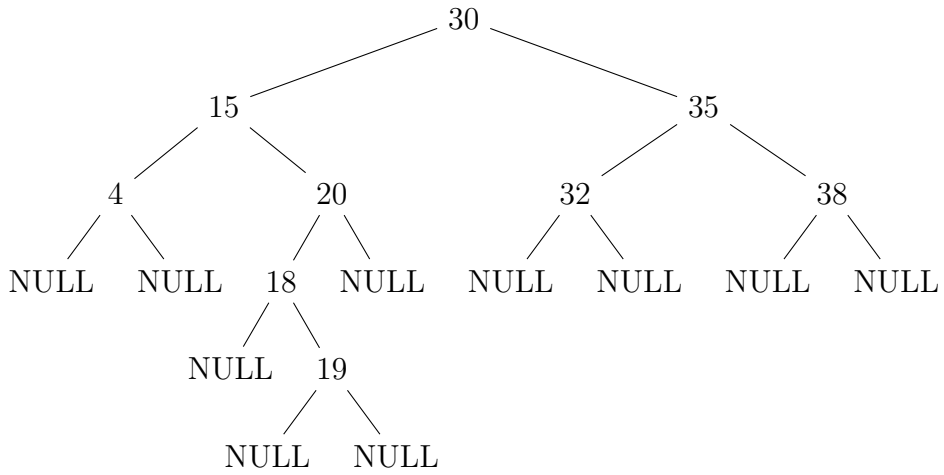
b



c



2



This is a binary search tree. All the nodes on the left of the root are smaller than it. All the nodes on the right are greater, and this is true for every sub-tree.
This isn't a full tree. Node 20 has exactly one child.
This isn't a complete tree. The first levels are, but all the descendents of 20 prevent it from being one.

3

This is simply tree traversal, with the addition that we need to keep track of the sum as we're traverssing the tree.

```
int sum_tree(Binary_Tree<int> BTree, int current_sum) {
  if (BTree == NULL) {
    return 0;
  }
  current_sum += sum_tree(BTree->left, current_sum);
  current_sum += sum_tree(BTree->right, current_sum);
  current_sum += get_data();
  return current_sum;
}
```

4

The simplest way to do this would be to turn this tree into a sorted array as if it were a binary search tree. Then, we simply check if the array is actually sorted. If it's not, then the tree must not be a binary search tree.

```
bool is_bst(BTNode<T>* root) {
  vector<T> vector;
  to_vector(root, vector) //As seen in Lecture 11 slides
  for(int i = 1; i < vector.size(); i++) {
    if (vector[i] < vector[i-1])
      return false;
  }
  return true;
}
```