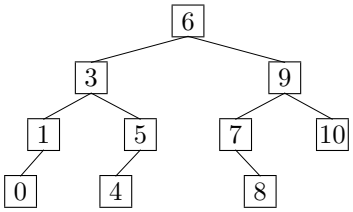
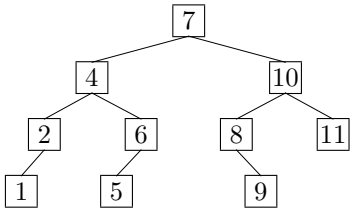
BST Test

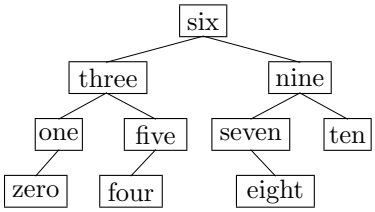
Original



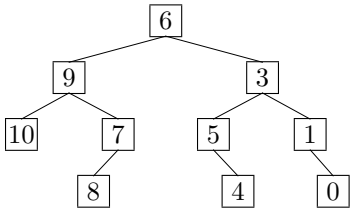
Incr



Strs



Rev




```
abstract class Comp<T>{ abstract boolean comp(T a, T b); }

class LTE extends Comp<Integer>{
    boolean comp(Integer a, Integer b){ return a<=b; }
}

class BST<T>{
    BST<T> insert(T d, Comp<T> c){ return new Node<T>(d, this, this); }
}

class Node<T> extends BST<T>{
    T data;
    BST left, right;

    Node(T d, BST<T> l, BST<T> r){ data = d; left = l; right = r; }
    BST<T> insert(T d, Comp<T> c){
       if(c.comp(d, data))
          return new Node<T>(data, left.insert(d,c), right);
       return new Node<T>(data, left, right.insert(d,c));
    }
}

class Incr extends IDf{
    int apply(Integer i){ return i+1; }
}

class Strs extends IDf{
    static String nums[] = {"zero","one","two","three","four","five",
                            "six","seven","eight","nine","ten"};
    String apply(Integer i){ return nums[i]; }
}

class Rev extends IDb{
    BST combine(Node t, Object d, BST l, BST r){ return new Node(d, r, l); }
    BST combine(BST l){ return l; }
}
```