Checkpoint 1

Download these files:


Implement and test the decrement operator for `tree_iterator`. Determine the appropriate sequence to insert the numbers 1-15 such that the resulting tree is exactly balanced. After using the `print_sideways` function to confirm the construction of this tree, test your iterators on the structure. Similarly, create a couple unbalanced trees to demonstrate that both the increment and decrement operators for iterators are debugged. Your decrement operator should correctly decrement the `end()` iterator. You can use the same “trick” we used in Lab 7 to make this work for `ds_list` iterators. A sample solution for Lab 7 is posted on the course webpage. Ask a TA if you have any questions.

**To complete this checkpoint:** Show one of the TAs your iterator decrement code and your tests cases.

Checkpoint 2

Add a member function called `accumulate` to the public interface of the `ds_set<T>` class, and provide its implementation. The function should take only one argument (of type T) and it should return the results of accumulating all the data values stored in the tree. The argument is the initial value for the accumulation. The function should only use `operator+=` on type T.

Test your code by showing that this works for both a set of ints, where the accumulate function should sum the values in the set (initial value parameter is 0), and a set of strings, where the accumulate function should concatenate the strings in the set (initial value parameter is ""). Does it matter if the `operator+=` for type T is commutative? How can you control the result of accumulate if it is not commutative?

**To complete this checkpoint:** Show a TA your completed and tested program.