Checkpoint 1

Checkpoint 1 will be available at the start of Wednesday’s lab.
*It will be a team-of-two exercise with one other person from your lab section.*

Checkpoint 2

Now let’s explore the implementation of the ds_set class, along with the use of recursive functions to manipulate binary search trees. Download and examine the files:

http://www.cs.rpi.edu/academics/courses/fall21/csci1200/labs/10_trees_I/ds_set.h
http://www.cs.rpi.edu/academics/courses/fall21/csci1200/labs/10_trees_I/test_ds_set.cpp

**PART 1:** The implementation of find provided in ds_set.h is recursive. Re-implement and test a non-recursive replacement for this function.

**PART 2:** The implementation of the copy constructor and the assignment operator is not yet complete because each depends on a private member function called copy_tree, the body of which has not yet been written. Write copy_tree and then test to see if it works by “uncommenting” the appropriate code from the main function.

**To complete this checkpoint:** Show one of the TAs your new code. Be prepared to discuss the running time for the two different versions of find for various inputs.

Checkpoint 3

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.