**Homework 3**

Given a binary search tree and a number n, find the node in the tree that contains the largest number that is less than n. If there is no value in the tree less than n then return nullptr. The binary search tree over which you will search will be given as a parameter. The Node interface is as follows:  
  
class Node {

public:

Node(int);

~Node();

int getValue();

std::shared\_ptr<Node> getLeft();

std::shared\_ptr<Node> getRight();

};

The public interface of your JltFinder class must include the following:

//public constructor, receives the root of the binary search tree over which to search.

JltFinder(std::shared\_ptr<Node>);

//search for the node with the value closest to but less than the value provided.

std::shared\_ptr<Node> Find(int);

**What to turn in:**

* Your Student.h header.
* A header file named JltFinder.h.
* An implementation file named JltFinder.cpp.
* All files in a directory named implementation.

**How the grading will work for this assignment:**

My grading script requires a simple flat directory structure. Place your file(s) in a directory named “implementation”. Note that the directory name is not capitalized. When your submission is unzipped it should contain a single directory named implementation. All of your source files must be included in the implementation directory.

The grader will unzip your submission and copy my grading program into your implementation directory along with any additional dependencies required by the program. The program will be compiled and run. If the program doesn’t compile with your implementation then you will receive no credit for the assignment. My test suite will then be run against your implementation. For an assignment worth m points, that has n tests, each test will be worth m/n points.

There will be 5 tests. This assignment is worth a total of 20 points. Each test is worth 4 points.