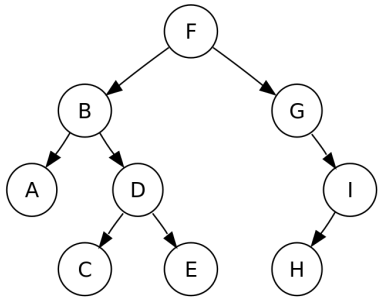
Assigned: Monday February 23, 2015, **Due: Friday, March 6, 2015 11:59pm.**



Binary Search Tree Implementation

1. Overview

This project is a straightforward binary search tree implementation. A binary search tree has the following invariant: all nodes to the left of a node are smaller in value, and all nodes to the right are larger in value.

Binary Search Trees have logarithmic asymptotic behavior for the following functions:

`find_min()`, `find_max()`, `contains()`, `insert(value)`, `remove()`, `tree_height()`, `node_count()`, and others.

The Binary Search Tree functions can be almost all be coded recursively. You will be responsible for writing all of the functions above recursively.

This project will also include writing a copy constructor and an assignment overload function. The copy constructor is called in the following cases:

```
BinarySearchTree *new_tree = old_tree;
      and
BinarySearchTree *new_tree(old_tree);
```

The assignment overload is called in the following case:

```
BinarySearchTree *new_tree;
new_tree = old_tree; // assignment overload gets called here
```

In other words, when you try to make a copy of an object, those functions get called, depending on the situation. Why? Basically, the general idea is to make a “deep copy” of your object (in this case, a binary tree full of nodes that have to be created dynamically). The C++ compiler only makes “shallow copies” automatically — pointers and fundamental types are copied to a new object.

There is nothing magical about either a copy constructor or an assignment operator overload, but it does take some time to learn how to write them. The format is always the same, however, so you should be able to follow the code on the page below to write the ones for this assignment.

You should read the following carefully in order to understand what these two functions do:

http://www.cplusplus.com/articles/y8hv0pDG/

2. Implementation Specifics

For this assignment, we will provide the BinarySearchTree implementation file (.h file), and a structured outline of the BinarySearchTree functions (.cpp file). You are responsible for writing most of the implementation functions, although we have provided you with some of the functions already. We have thoroughly commented **`BinarySearchTree.h`**, so please refer to that file for specific information.

3. Low Level Details

Getting the files

There are two ways to get the files for this assignment. The first is by copying the original files from the class folder. The second is to use "git" to pull the files from the GitHub cloud server.

Method 1: copy files from the class folder

First ssh to the homework server and, and make a directory called orderedListAssignment

```
ssh -X your_cs_username@homework.cs.tufts.edu
mkdir hw4
```

change into that directory

```
cd hw4
```

At the command prompt, enter (don't forget the period):

```
cp /comp/15/public_html/assignments/hw4/files/* .
```

Method 2: pull from GitHub:

At the command prompt, enter

"`git clone https://github.com/Tufts-COMP15/2015s_HW4.git`"

change into the directory that git created:

"`cd hw4`"

Using Eclipse (assuming you have followed the steps above to get the files):

1. See online video https://www.youtube.com/watch?v=BbnLZ2oqrD0
2. If in the lab, simply open Eclipse, and then start following from step 6 below.
3. If at home, ensure that you have installed (Mac) XQuartz, or (Windows) Cygwin (see here: https://www.youtube.com/watch?v=BbnLZ2oqrD0) or (Win) putty and XMing (http://sourceforge.net/projects/xming/).
4. ssh to the homework server using the -X argument:
   ssh -X your_cs_username@homework.cs.tufts.edu
5. start eclipse by typing "eclipse" (no quotes)

6. Use the following steps to set up your project (you'll get used to the steps quickly, steps A & B only need to be done once at the beginning of the course):
   A. Default location for your workspace is fine
   B. Click on "Workbench", then click "Window->Open Perspective->Other", and choose C++
   (note: if C++ is not listed, close Eclipse and re-open by typing Eclipse)
   C. Window->Preferences
      General->Editors->Text Editors

Displayed tab width (8) (recommended)
Show print margin: 80 col
Show line numbers
C/C++ -> Code Style
Select K&R [built-in] and then “Edit”, then rename to “K&R 8-tab”
Change Tab size to 8 (recommended)
Click “Apply” then “Ok”

D. File->New C++ Project
Fill in project name (no spaces!) IMPORTANT: the name CANNOT be exactly the same as your folder name. I suggest to simply put name_project (with the _project) for all project names (e.g., hw4_project)
Use default location should be checked.
Select Empty Project->Linux GCC
Click Next
Click "Advanced Settings"
1. On the left, under C/C++ General->Paths and Symbols (left hand side)
(you may have to click on the triangle next to C/C++ General)
Select "[Debug]" at the top for Configuration
Source Location Tab
Click "Link Folder"
Check "Link to folder in the file system"
Browse to find your folder.
(should be something like h/your_username/hw4)
Click OK
Click Apply

2. C++ Build->Settings (may have to click on the triangle)
For Configuration (at the top), select [All configurations]
GCC C++ Compiler: Command should be "clang++" (no quotes)
GCC C Compiler: Command: clang
GCC C++ Linker: Command: clang++
Click "Apply"
Click OK

Click Finish

E. Click on the white triangle next to your project name.
The folder you linked to should be listed. Click on its triangle.
The files in that folder should be listed.

F. Test the build by clicking on the hammer in the icon bar.
You should see some compiling messages in the Console window at the bottom. (Things like, "Building file...; clang++, etc.)

G. The program will compile, but will have many warnings. You need to write the functions — double-click on List_linked_list.h to see the header file and List_linked_list.cpp to see the code.

Compiling and running:

1. At the command prompt, enter “**`make`**” and press enter or return to compile.
2. At the command prompt, enter “**`./hw4`**” and press return to run the new program.

Providing:

At the command prompt, enter “**`make provide`**” and press enter or return.