**Lab 7 - Binary Search Trees**

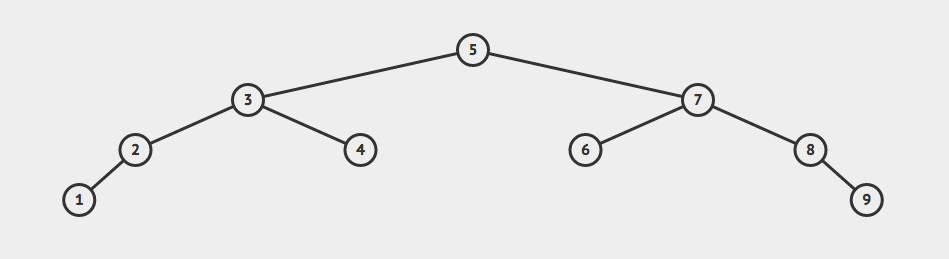
# Part 1: BST

* Create a link based Binary Search tree composed of a Node and a Tree struct. You should have a header file, BST.h, with the following:
  + Node struct containing left, right, and parent pointers, in addition to holding an Data struct value.
  + Tree struct containing a pointer to the root of the tree.
  + A function declaration for a function that allocates a tree, and initializes the root to NULL;
  + A function declaration for a function that takes a Data struct as a parameter, allocates a node, and initializes the left, right, parent fields to NULL.
* You should also have a source file, BST.c, that implements the two declared functions:
  + Tree \* createTree();
  + Node \* createNode();
* Test your functions and structure to ensure everything is initialized correctly by creating a Tree and adding a root to it.

# Part 2: BST Operations

* Alter your header file to contain the function declarations for insert and search. Implement the operations in your BST.c file.
* INSERT:
  + Create a function, void insert(Tree \* bst, Data value), that inserts into the tree – Helpful hints:
    - Make sure you check for the special case of an empty tree [if(bst->root == NULL)],
    - After checking for the root, use a separate helper function to insert a value into the tree, void insertNode(Node \* node, Data value), that you can use for the recursive call
    - If the value is already in the tree, just print a message that you cannot insert duplicate values
* SEARCH:
  + Create a function, Node \* search(Tree \* bst, Data value), that searches for a value in the tree. You only need to print out if the value is found or not – Helpful hints:
    - Make sure you check for the special case of an empty tree [if(bst->root == NULL)],
    - After checking for the root, use a separate helper function to search the tree, Node \* searchNode(Node \* node, Data value), that you can use for the recursive call

Part 3: Testing Your Tree

* In your main, do the following to test your tree:
  + Using your insert function, read in the 9 integers for a data.txt file (put these numbers in your file: 5 7 3 8 2 6 4 9 1), create a Data struct for each, and insert them into the tree.
    - Your tree should look like the image below:
  + Using your search function, prompt the user for a value to search for in the tree or 0 to stop searching.
    - Print out the node value, the parent node value, and the child node values, if not a leaf node

# Part 4: Submission

Create a tar archive with the command “tar -czvf lab7.tar.gz .”, and then email your archive to bu580u2017@gmail.com and cc your TA dmu1@binghamton.edu before the submission deadline. Make sure you do not include the executable in your archive (make clean before creating the archive). Late assignments will not be accepted under any circumstances. Plan to turn in your assignments early.

Demo your lab before the demo deadline (after the submission deadline) by downloading your submission from class Gmail and extracting your archive with the command “tar -xvf lab7.tar.gz”. Then compile (with your makefile), and run your code, show your source to your TA, and answer any questions your TA may have.

# Grading Guidelines

## Part 1:

* + Part A: 3 points
* **Part 2:**
  + Insert works correctly: 3 points
  + Search works correctly: 2 points
* **Part 3:**
  + Tests by using data.txt: 1 point

## Style Guidelines and Memory Leaks

* + Follows Style Guidelines: 1 point

Submission Deadline: 11:59pm 12/10/2017 EST

Demo Deadline: 2:00pm 12/15/2017 EST

Note: You can demo before the submission deadline for this lab.