Lab 4: Implementing a Binary Search Tree

By Dr. Yingwu Zhu, Seattle University

Requirements

In this lab, you need to implement a binary search tree (BST) using a linked list. The BST is a dynamic set that maintains a list of unique integers. It must support the following operations as listed in the table below.

`Constructor`	Construct an empty list.
`Destructor`	Invoked when a BST object is destroyed.
`bool Empty() const`	Return true if the tree is empty and false otherwise.
`bool Search(int val)`	Return true if the BST contains the integer val. It is an iterative function.
`Bool RecurSearch(int val)`	Return true if the BST contains the integer val. It is a recursive function.
`void Insert(int val)`	Insert an integer val into the tree if it doesn't exist on the tree.
`int MinElement()const`	Return the minimum integer on the tree.
`Int MaxElement() const`	Return the maximum integer on the tree
`void Erase(int val)`	Remove an integer val from the tree if it exists on the tree.

You need to create four files:

- bst.h: BST header file
- bst.cpp: BST implementation file
- Makefile
- client.cpp: driver program to test your BST

BST header file

The header file is provided below. Note that you cannot change the data members and public member functions. However, you may need to add your own auxiliary private member functions for some reason. For example, when you implement your RecurSearch(), you need add a helper function to do recursive search. You can add this helper function as a private member function.

```
//bst.h
#ifndef _BST_H_
#define _BST_H_
```

```
#include <iostream>
using namespace std;

class BST {
   private:
      class Node {
        public:
          int  data_;
          Node* left_;
          Node* right_;
      };
      Node* root_;  //root node pointer
      //you may add your auxiliary functions here!
   public:
      BST();  //constructor
      ~BST(); //destructor
      bool Empty() const;
      void Insert(int val);
      int MinElement() const;
      int MaxElement() const;
      bool Search(int val) const;
      bool RecurSearch(int val) const;
      void Erase(int val);
};
#endif
```

Suggested Approach

1. Implement and test constructor, destructor, and Empty().
2. Implement and test Insert(), MinElement() and MaxElement().
3. Implement and test Search().
4. Implement and test RecurSearch().
5. Implement and test Erase().
6. Conduct an extensive test on your BST.