CPSC 223: Assignment #6

Due: Fri., Apr.12th, 2019

Implement an initial pointer-based BinarySearchTree class for storing binary search trees. **Hand in** a hard-copy of your code; and **Submit** your code to the class account on ada (where your code should compile and run).

STEP 1: **Declare and implement a** BinaryTree **class**. This class should be decleared in a file binarytree.h and implemented in a file binarytree.cpp. Your binarytree should be a friend class of a Node class and have a *protected* data field mroot of type Node pointer.

```
class  BinaryTree {
   public:
      ...
   protected:
      .....
      Node * mroot;
};
```

It should be declared in the protected section of your class so that we can extend and use it in future subclasses, like BinarySearchTree.

Implement the following for BinaryTree:

- default constructor, destructor, and copy constructor
- an assignment operator is defined, implemented, and commented. It's for your reference.
- bool IsEmpty() const;
- void preorderTraverse () const; in this assignment, it is a preorder print of the binary tree
- void inorderTraverse () const; it's an inorder print of the binary tree
- void postorderTraverse() const;

You will need the following helper functions declared in protected section:

- void copyTree (Node*& newtreep, Node* oldtreep); // for copy constructor
- void destroyTree (Node*& treep); // for destructor
- void preorder (Node* treep) const;
- void inorder (Node* treep) const;
- void postorder (Node* treep) const;

Note: To finish your program efficiently, a better way is to comment all other methods that you are not using/testing, especially those declared but not implemented.

STEP 2: **Test your BinaryTree class.** Use different constructors to build up trees and print them out using traversal methods.

STEP 3: **Declare and implement a** BinarySearchTree **class**. This class should be decleared in a file binarysearchtree.h and implemented in a file binsearchtree.cpp. Your BinarySearchTree class should be a child class of a BinaryTree class. Both BinaryTree and BinarySearchTree are friend classes of the Node class:

```
class  BinarySearchTree: public BinaryTree{
   public:
      ...
   protected:
      ...
};
```

Implement the following for BinarySearchTree:

- bool Search (const ItemType& theItem) const;
- void Insert (const ItemType& newItem);

You will need the following helper functions declared in protected field:

- bool lookup (Node * treeptr, const ItemType& theItem) const;
- void insertItem (Node *& treeptr, const ItemType& newItem);

Note: we will implement the remove, FindMax, and FindMin function in next assignment.

STEP 4: Test search and insert functions in your binary search tree class. Show the result of every operation by calling inorderTravesal().

STEP 5: **Place your files in a hw7 directory, and submit it.** Also, be sure to turn in hard-copy of your code, and any input besides your test file you used to test your code.