#include "list.h"

#include <iostream>

#include <fstream>

using namespace std;

BST::BST()

{

root = nullptr;

}

BST::~BST()

{

}

//Develops the tree from the root to the leaves

void BST::addLeaves(int item)

{

addLeafPrivate(item,root);

}

//Leaf creation algorithm

//Note: node is in the scope of the BST class

BST::node\* BST::createLeaf(int item)

{

node \*newNode= new node;

newNode->data= item;

newNode->left=nullptr;

newNode->right=nullptr;

return newNode;

}

//Determines where leaves attach to the tree

void BST::addLeafPrivate(int item, node\* ptr)

{

if(root == nullptr)

root= createLeaf(item);

else if (item < ptr->data)

{

if(ptr->left != nullptr)

addLeafPrivate(item, ptr->left);

else

ptr->left = createLeaf(item);

}

else if (item > ptr->data)

{

if(ptr->right != nullptr)

addLeafPrivate(item, ptr->right);

else

ptr->right= createLeaf(item);

}

else

cout<<"duplicate items are not allowed!!!"<<endl;

}

//Traverse leftmost node, prints data, parent,prints data, then rightmost node, prints data

void BST::InOrder(node\* ptr)

{

if(ptr != nullptr)

{

InOrder(ptr->left);

cout<<ptr->data<< " ";

InOrder(ptr->right);

}

}

//Prints node data while traversing leftmost then rightmost

void BST::preOrder(node\* ptr)

{

if(ptr != nullptr)

{

cout<< ptr->data<< " ";

preOrder(ptr->left);

preOrder(ptr->right);

}

}

//Traverses to leftmost node, print data, then rightmost node, print data, then parent node, print data

void BST::postOrder(node\* ptr)

{

if(ptr != nullptr)

{

postOrder(ptr->left);

postOrder(ptr->right);

cout<<ptr->data<<" ";

}

}

/\*

A combination of public and private functions

create a layer of protection between the program and

the user

\*/

void BST::printPreOrder()

{

preOrder(root);

}

void BST::printPostOrder()

{

postOrder(root);

}

void BST::printInOrder()

{

InOrder(root);

}

void BST::input(int x)

{

int data;

int count = 0;

ifstream myfile;

myfile.open("ints.txt");

while(myfile>>data && count < x)

{

addLeaves(data);

count++;

}

myfile.close();

}

void BST::original(int x)

{

int data;

int count = 0;

ifstream myfile;

myfile.open("ints.txt");

while(myfile>>data && count < x)

{

cout << data <<" ";

count++;

}

myfile.close();

}