//header file for the BST node and BST

#include <iostream>

#include <string>

using namespace std;

class BST\_Node{ //node in a BST--?Hold client information

public:

string lastname, firstname, address, phone;

BST\_Node \* lchild, \* rchild; /\*left and right children pointers\*/};

class Clients\_Info\_BST{ //Binary Search Tree

public:

Clients\_Info\_BST(); //stores the data in the hash table

Clients\_Info\_BST(const Clients\_Info\_BST &); //Copy Constructor

~Clients\_Info\_BST(); //Destructor

void Insert(const string &); //Insert adds a new Client's information (into a BST\_Node) into the BST

void Remove(const string &);

//Remove deletes a BST\_Node that contains the specified client info from the BST if it is there;

//otherwise a message should be printed stating so.

void Update(const string &);

//Update modifies a client information given the first and last name if it is in the BST; otherwise prints a message stating so

void Print(); //Print outputs a BST, INORDER, to the display

BST\_Node \* Search(const string &);

bool Empty(){return (root==0);}; //returns true if BST is empty; otherwise false

void Insert(BST\_Node \* &, string); //Auxicilary function used by Insert above to allow recursion

void Remove(BST\_Node \* & loc\_ptr, string); //Auxicilary function used by Removbe above to allow recursion

BST\_Node \* Search(BST\_Node \*,string); //Auxicilary function used by Search above to allow recursion

BST\_Node \* inorder\_succ(BST\_Node \*); //Return pointer to inorder successor; otherwise 0;

void Print(BST\_Node \*); //Auxicilary function used by Print above to allow recursion

private:

BST\_Node \* root; //---state information

};