#include <iostream>

#include <string>

using namespace std;

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

public:

string lastname, firstname, address, phone\_number;

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 & s){cout<<"Inside Client\_Info\_BST Insert\n";};

//Insert adds a new Client's information (into a BST\_Node) into the BST

void Remove(const string & s){cout<<"Inside Client\_Info\_BST Remove\n";};

//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 & s){cout<<"Inside Client\_Info\_BST Update\n";};

//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( ){cout<<"Inside Client\_Info\_BST Print\n";};

//Print outputs a BST, INORDER, to the display

BST\_Node \* Search(const string & s){cout<<"Inside Client\_Info\_BST Search\n"; return 0;};

///\* 'Other possible memberf functions (some may be public and some may be private

//You may need to implement other member functions\*/

bool Empty(); //returns true if BST is empty; otherwise false

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

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

BST\_Node \* Search(BST\_Node \*,string item); //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

};

class Client\_Address\_Book{

public:

Client\_Address\_Book(){};//default constructor will read data from input file "client\_address\_data.txt".

Client\_Address\_Book(const Client\_Address\_Book &); //Copy Constructor

~Client\_Address\_Book();//Destructor

void Insert(const string & s); // insert record

//Insert adds a new Client's information to the hash table

void Remove(const string & s); //remove record

//Remove deletes a client from the hash table if it is there; otherwise a message should be printed stating so.

void Update(const string & s); //update record; see example below

void Print\_BST(const string & s); //Print a BST (cell in hash table) inorder to the screen

void Print\_Hash\_Table(){"Inside Client\_Address\_Book Print\_Hash\_Table\n";};

//function will print hash table to the screen

void Print\_Hash\_Table\_to\_File(const string & filename);

//function will print hash table to output file

bool \* Search(const string & s){"Inside Client\_Address\_Book Search\n"; return 0;};

//return true if client found; otherwise false

unsigned int Hash\_Function(const string & s);

//return the index of the BST in the hash table

//Hint:Remember that the insert, remove and search function for Clients\_Address\_Book will use Client\_Info\_BST’s insert, remove and search respectively.

private:

int capacity; //SET THIS VALUE EQUAL TO 27 YOUR DEFAULT CONSTRUCTOR //STOP CAPS LOCKING

Clients\_Info\_BST \*hash\_table; // USING 1 THROUGH 26 or whatever you like

};

int main(){

Client\_Address\_Book My\_Book;

//My\_Book.Insert("Bullard Lofton 777 Glades Road 207-2780");

//My\_Book.Remove("Bullard Lofton");

//My\_Book.Update("1 Bullard Lofton Comb Harry 555 Palmetto Park Road 555-3444");

//My\_Book.Print\_BST("B");

//My\_Book.Print\_Hash\_Table();

//Client\_Address\_Book Your\_Book = My\_Book; //Invoke the copy constructor

//Your\_Book.Print\_Hash\_Table\_to\_File(/\* the output filename goes here\*/);

return 0;}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Notes for Update Function:

1. My\_Book\_Update(“1 James Clark Lofton Bullard 777 Glades Run 527-6623”);

If first character is a 1, this means all three fields will be changed.

2. My\_Book\_Update(“2 James Clark Lofton Bullard 777 Glades Run”);

If first character is a 2, this means the Name and Address fields will be changed.

3. My\_Book\_Update(“3 James Clark 777 Glades Run 555-6666”);

If first character is a 3, this means the Address and Phone Number fields will be changed.

4. My\_Book\_Update(“4 James Clark Lofton Bullard 555-6666”);

If first character is a 4, this means the Name and Phone Number fields will be changed.

5. My\_Book\_Update(“5 James Clark Lofton Bullard”);

If first character is a 5, this means the Name field will be changed.

6. My\_Book\_Update(“6 James Clark 777 Glades Run”);

If first character is a 6, this means the Address field will be changed.

7. My\_Book\_Update(“7 James Clark 555-6666”);

If first character is a 7, this means the Phone Number field will be changed.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/