#include "bintree.h"

#include <fstream>

#include <string>

#include <iostream>

using namespace std;

void BinarySearchTree::create\_animal\_tree( string file\_name )

{

BinarySearchTree::tree\_node \*tem\_tree\_node = nullptr;

int value;

string question;

string dummy;

ifstream input\_file;

input\_file.open( file\_name );

while( input\_file >> value )

{

getline( input\_file, dummy );

getline( input\_file, question );

tem\_tree\_node = create\_tree\_node( value, question );

root\_checker( tem\_tree\_node );

connector( root, tem\_tree\_node );

}

// tree\_display(root);

input\_file.close();

}

BinarySearchTree::tree\_node\* BinarySearchTree::create\_tree\_node(

int number, string question )

{

tree\_node \*newnode = nullptr;

newnode = new tree\_node;

newnode->number = number;

newnode->question = question;

newnode->left\_node = nullptr;

newnode->right\_node = nullptr;

return newnode;

}

void BinarySearchTree::connector( tree\_node\*&tem\_tree\_node,

tree\_node \*&new\_tree\_node )

{

const int TWO = 2;

const int ONE = 1;

if( tem\_tree\_node != nullptr )

{

if( TWO \* ( tem\_tree\_node->number ) == new\_tree\_node->number )

{

tem\_tree\_node->left\_node = new\_tree\_node;

return;

}

if( TWO \* ( tem\_tree\_node->number ) + ONE == new\_tree\_node->number )

{

tem\_tree\_node->right\_node = new\_tree\_node;

return;

}

if( TWO \* ( tem\_tree\_node->number ) + ONE != new\_tree\_node->number )

{

connector( tem\_tree\_node->right\_node, new\_tree\_node );

}

if( TWO \* ( tem\_tree\_node->number ) != new\_tree\_node->number )

{

connector( tem\_tree\_node->left\_node, new\_tree\_node );

}

}

}

void BinarySearchTree::root\_checker( tree\_node \*&new\_tree\_node )

{

if( root == nullptr )

{

root = new\_tree\_node;

}

}

void BinarySearchTree::tree\_display( tree\_node\*&tem\_tree\_node )

{

if( tem\_tree\_node )

{

cout << "The number is " << tem\_tree\_node->number <<

" and the question is " << tem\_tree\_node->question << endl;

tree\_display( tem\_tree\_node->left\_node );

tree\_display( tem\_tree\_node->right\_node );

}

}

void BinarySearchTree::start\_to\_play( string file\_name )

{

const int ONE = 1;

tree\_node \*dummy\_node = nullptr;

int loop\_guard = 1;

do

{

cout << endl;

choice( root, dummy\_node, loop\_guard );

}

while( loop\_guard == ONE );

fstream out\_put\_file;

out\_put\_file.open( file\_name );

read\_in\_file( root, out\_put\_file );

cout << "The animals binary tree has been saved in your original file!"

<< endl;

delete\_whole\_tree( root );

cout << "And the animal binary tee has been deleted, have a good day!!!!"

<< endl;

}

void BinarySearchTree::choice( tree\_node\*& next\_node,

tree\_node\*& previews\_node, int &loop\_guard )

{

const unsigned ONE = 1;

const unsigned ZERO = 0;

previews\_node = next\_node;

int user\_choice;

if( next\_node->right\_node == nullptr

&& next\_node->left\_node == nullptr )

{

cout << next\_node->question << endl;

cout << "Your answer: ";

user\_choice = get\_answer();

if( user\_choice == ONE )

{

cout << "Nice, I got right answer!!!!!" << endl;

cout << "Do you want to play again?" << endl;

loop\_guard = get\_answer();

}

else

if( user\_choice == ZERO )

{

cout << "Ok, fine, I can't figure out what your animal is." << endl;

add\_node( previews\_node );

cout << "Do you want to play again?" << endl;

loop\_guard = get\_answer();

}

}

else

{

cout << "Question: " << next\_node->question << endl;

cout << "Your answer: ";

user\_choice = get\_answer();

if( user\_choice == ZERO ) // NO -> left\_node

{

choice( next\_node->left\_node, previews\_node, loop\_guard );

}

if( user\_choice == ONE ) // YES -> right\_node

{

choice( next\_node->right\_node, previews\_node, loop\_guard );

}

}

}

int BinarySearchTree::get\_answer()

{

const int ONE = 1;

const int ZERO = 0;

int number;

string answer;

cin.clear();

getline( cin, answer );

if( answer == "yes" || answer == "YES" || answer == "Yes" )

{

return ONE;

}

else

{

if( answer == "No" || answer == "NO" || answer == "no" )

{

return ZERO;

}

else

{

cout << "Please enter the valid answer, only yes or no!!!" << endl;

return number = get\_answer();

}

}

}

void BinarySearchTree::add\_node( tree\_node \*previous\_node )

{

const int ONE = 1;

const int TWO = 2;

string is\_it = "Is it ";

string question\_mark = "?";

string new\_a;

string new\_q;

cout << "What is your animal in your mind? " << endl;

getline( cin, new\_a );

cin.clear();

cout << "OK. Help my by adding a question! Type a question to distinguish these two animals."

<< endl;

getline( cin, new\_q );

cin.clear();

tree\_node \*new\_right\_tree\_node = new tree\_node;

new\_right\_tree\_node->question = is\_it + new\_a + question\_mark;

new\_right\_tree\_node->number = ( previous\_node->number ) \* TWO + ONE;

new\_right\_tree\_node->right\_node = nullptr;

new\_right\_tree\_node->left\_node = nullptr;

tree\_node \*new\_left\_tree\_node = new tree\_node;

new\_left\_tree\_node->question = previous\_node->question;

new\_left\_tree\_node->number = ( previous\_node->number ) \* TWO;

new\_left\_tree\_node->right\_node = nullptr;

new\_left\_tree\_node->left\_node = nullptr;

previous\_node->question = new\_q;

previous\_node->right\_node = new\_right\_tree\_node;

previous\_node->left\_node = new\_left\_tree\_node;

return;

}

void BinarySearchTree::read\_in\_file( tree\_node \*tem\_tree\_node,

fstream &out\_file )

{

if( tem\_tree\_node == nullptr )

{

return;

}

else

{

out\_file << tem\_tree\_node->number << endl;

out\_file << tem\_tree\_node->question << endl;

read\_in\_file( tem\_tree\_node->left\_node, out\_file );

read\_in\_file( tem\_tree\_node->right\_node, out\_file );

return;

}

}

void BinarySearchTree::delete\_whole\_tree( tree\_node \*& tem\_tree\_node )

{

if( tem\_tree\_node->right\_node == nullptr )

{

delete\_whole\_tree( tem\_tree\_node->right\_node );

}

if( tem\_tree\_node->left\_node == nullptr )

{

delete\_whole\_tree( tem\_tree\_node->left\_node );

}

delete tem\_tree\_node;

}