/\*C++ Program To read details of a book consists of chapters,

\*chapters consist of sections and sections consist of

\*subsections. Construct a tree and print the nodes.

\*Find the time and space requirements of your method.

\*\*/

l # include <iostream>

# inc ude <cstdlib>

# include <string.h>

using namespace std;

/\*

\* Node Declaration

\*/

struct node

{

char label[10];

int ch\_count;

struct node \*child[10];

}\*root;

/\*

\* Class Declaration

\*/

class GT

{

public:

void create\_tree();

void display(node \* r1);

GT()

{

root = NULL;

}

};

void GT::create\_tree()

{

int tbooks,tchapters,i,j,k;

root = new node;

cout<<"Enter name of book";

cin>>root->label;

cout<<"Enter no. of chapters in book";

cin>>tchapters;

root->ch\_count = tchapters;

for(i=0;i<tchapters;i++)

{

root->child[i] = new node;

cout<<"Enter Chapter name\n";

cin>>root->child[i]->label;

cout<<"Enter no. of sections in Chapter: "<<root->child[i]->label;

cin>>root->child[i]->ch\_count;

for(j=0;j<root->child[i]->ch\_count;j++)

{

root->child[i]->child[j] = new node;

cout<<"Enter Section "<<j+1<<"name\n";

cin>>root->child[i]->child[j]->label;

//cout<<"Enter no. of subsections in "<<r1->child[i]->child[j]->label;

//cin>>r1->child[i]->ch\_count;

}

}

}

void GT::display(node \* r1)

{

int i,j,k,tchapters;

if(r1 != NULL)

{

cout<<"\n-----Book Hierarchy---";

cout<<"\n Book title : "<<r1->label;

tchapters = r1->ch\_count;

for(i=0;i<tchapters;i++)

{

cout<<"\n Chapter "<<i+1;

cout<<" "<<r1->child[i]->label;

cout<<"\n Sections";

for(j=0;j<r1->child[i]->ch\_count;j++)

{

//cin>>r1->child[i]->child[j]->label;

cout<<"\n "<<r1->child[i]->child[j]->label;

}

}

}

}

/\*

\* Main Contains Menu

\*/

int main()

{

int choice;

GT gt;

while (1)

{

cout<<"-----------------"<<endl;

cout<<"Book Tree Creation"<<endl;

cout<<"-----------------"<<endl;

cout<<"1.Create"<<endl;

cout<<"2.Display"<<endl;

cout<<"3.Quit"<<endl;

cout<<"Enter your choice : ";

cin>>choice;

switch(choice)

{

case 1:

gt.create\_tree();

case 2:

gt.display(root);

break;

case 3:

exit(1);

default:

cout<<"Wrong choice"<<endl;

}

}

}