```cpp
#include <iostream>
#include <string>
using namespace std;

struct node // Node Declaration
{
    string label;
    int ch_count; // Number of child nodes (chapters, sections, or subsections)
    struct node *child[10]; // Child nodes (can be chapters, sections, or subsections)
} *root;

class GT // Class Declaration
{
public:
    void create_tree();
    void display(node *r1);

    GT()
    {
        root = NULL;
    }
};

void GT::create_tree()
{
    int tchapters, tsections, tsubsections, i, j, k;

    root = new node; // Create the root node
    cout << "Enter name of book: ";
    cin.get();
    getline(cin, root->label);

    cout << "Enter number of chapters in book: ";
    cin >> tchapters;
    root->ch_count = tchapters;

    for (i = 0; i < tchapters; i++)
    {
        root->child[i] = new node; // Create a new node for each chapter
        cout << "Enter the name of Chapter " << i + 1 << ": ";
        cin.get();
        getline(cin, root->child[i]->label);

        cout << "Enter number of sections in Chapter " << root->child[i]->label << ": ";
        cin >> tsections;
        root->child[i]->ch_count = tsections;

        for (j = 0; j < tsections; j++)
        {
            root->child[i]->child[j] = new node; // Create a new node for each section
            cout << "Enter Name of Section " << j + 1 << " in Chapter " << root->child[i]->label << ": ";
            cin.get();
            getline(cin, root->child[i]->child[j]->label);

            cout << "Enter number of subsections in Section " << root->child[i]->child[j]->label << ": ";
            cin >> tsubsections;
            root->child[i]->child[j]->ch_count = tsubsections;

            for (k = 0; k < tsubsections; k++)
            {
                root->child[i]->child[j]->child[k] = new node; // Create a new node for each subsection
                cout << "Enter Name of Subsection " << k + 1 << " in Section " << root->child[i]->child[j]->label << ": ";
                cin.get();
                getline(cin, root->child[i]->child[j]->child[k]->label);
            }
        }
    }
}

void GT::display(node *r1)
{
    int i, j, k, tchapters, tsections, tsubsections;

    if (r1 != NULL)
    {
        cout << "\n----- Book Hierarchy -----";
        cout << "\nBook title: " << r1->label;
        tchapters = r1->ch_count;

        for (i = 0; i < tchapters; i++)
        {
            cout << "\nChapter " << i + 1 << ": " << r1->child[i]->label;
            tsections = r1->child[i]->ch_count;

            for (j = 0; j < tsections; j++)
            {
                cout << "\n  Section " << j + 1 << ": " << r1->child[i]->child[j]->label;
                tsubsections = r1->child[i]->child[j]->ch_count;

                for (k = 0; k < tsubsections; k++)
                {
                    cout << "\n    Subsection " << k + 1 << ": " << r1->child[i]->child[j]->child[k]->label;
                }
            }
        }
    }
    cout << endl;
}

int main()
{
    int choice;
    GT gt;

    while (true)
    {
        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();
            break;
        case 2:
            gt.display(root);
            break;
        case 3:
            cout << "Thanks for using this program!" << endl;
            return 0; // Exit the program
        default:
            cout << "Wrong choice!" << endl;
        }
    }
    return 0;
}
```