Practical 3:

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

// Node structure representing each unit in the book hierarchy
struct Node {
    string data;
    Node* nextSibling;
    Node* firstChild;

    Node(string value) {
        data = value;
        nextSibling = nullptr;
        firstChild = nullptr;
    }
};

// Function to insert a child node
Node* insertChild(Node* parent, const string& childData) {
    Node* child = new Node(childData);
    if (!parent->firstChild) {
        parent->firstChild = child;
    } else {
        Node* temp = parent->firstChild;
        while (temp->nextSibling) {
            temp = temp->nextSibling;
        }
        temp->nextSibling = child;
    }
    return child;
}

// Recursive function to display the book structure
void displayTree(Node* node, int level = 0) {
    if (!node) return;

    for (int i = 0; i < level; i++) cout << "\t";
    cout << "- " << node->data << endl;

    displayTree(node->firstChild, level + 1);
    displayTree(node->nextSibling, level);
}
```

```
// Recursive function to delete the tree and free memory
void deleteTree(Node* node) {
    if (!node) return;
    deleteTree(node->firstChild);
    deleteTree(node->nextSibling);
    delete node;
}

int main() {
    string bookTitle;
    cout << "Enter the title of the book: ";
    getline(cin, bookTitle);

    Node* book = new Node(bookTitle);

    int numChapters;
    cout << "Enter the number of chapters in the book: ";
    cin >> numChapters;
    cin.ignore();

    for (int i = 0; i < numChapters; i++) {
        string chapterTitle;
        cout << "Enter title of chapter " << (i + 1) << ": ";
        getline(cin, chapterTitle);

        Node* chapterNode = insertChild(book, chapterTitle);

        int numSections;
        cout << "Enter number of sections in chapter \"" << chapterTitle << "\": ";
        cin >> numSections;
        cin.ignore();

        for (int j = 0; j < numSections; j++) {
            string sectionTitle;
            cout << "Enter title of section " << (j + 1) << ": ";
            getline(cin, sectionTitle);

            Node* sectionNode = insertChild(chapterNode, sectionTitle);

            int numSubsections;
            cout << "Enter number of subsections in section \"" << sectionTitle << "\": ";
            cin >> numSubsections;
            cin.ignore();
            for (int k = 0; k < numSubsections; k++) {
                string subsectionTitle;
                cout << "Enter title of subsection " << (k + 1) << ": ";
                getline(cin, subsectionTitle);
                insertChild(sectionNode, subsectionTitle);
            }
        }
    }

    // Display the entire structure
    cout << "\nBook Structure:\n";
    displayTree(book);

    // Clean up dynamically allocated memory
    deleteTree(book);

    return 0;
}
```