**Practical No: 03**

**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.**

**Algorithm**

**Step 1: Start**

1. Define a node structure that holds:
   * A label (string) for the name of the book, chapter, or section.
   * An integer ch\_count to represent the number of children.
   * An array child[] of pointers to child nodes (maximum of 10).

**Step 2: Initialize the Tree**

* Create a class GT with:
  + A pointer root to represent the root of the tree.
  + Methods:
    - create\_tree() to build the tree structure.
    - display(node\* r1) to traverse and print the tree.

**Step 3: Implement the create\_tree Function**

* **Input**: Get the name of the book and the number of chapters.
* Allocate memory for the root node and store the book's name in its label.
* For each chapter:
  + Create a child node for the chapter and assign its name.
  + Get the number of sections in the chapter.
  + For each section:
    - Create a child node for the section and assign its name.

**Step 4: Implement the display Function**

* **Input**: A pointer to the root node.
* If the root node is not NULL:
  + Print the book title.
  + For each chapter:
    - Print the chapter name.
    - For each section of the chapter:
      * Print the section name.

**Step 5: Main Menu**

* Display a menu with three options:
  1. **Create**: Call the create\_tree() function to build the hierarchy.
  2. **Display**: Call the display() function to print the hierarchy.
  3. **Quit**: Exit the program.

**Step 3: Create the Tree**

* Begin the creation process by prompting the user for the book’s name and the number of chapters.
* **For each chapter**:
  + Create a child node for the chapter.
  + Ask the user to input the name of the chapter and the number of sections it contains.
  + **For each section**:
    - Create a child node for the section.
    - Ask the user to input the section’s name.

**Step 4: Display the Tree**

* Perform a **level-wise traversal** starting from the root:
  1. Print the name of the book (root label).
  2. **For each chapter**:
     + Print its name.
     + **For each section within the chapter**:
       - Print its name.

**Step 5: Menu for User Interaction**

* Display a menu with the following options:
  1. **Create**: Call the create\_tree() method to build the tree.
  2. **Display**: Call the display(root) method to print the hierarchy.
  3. **Quit**: Exit the program.

#include <iostream>

#include <string.h>

using namespace std;

struct node // Node Declaration

{

string label;

//char label[10];

int ch\_count;

struct node \*child[10];

} \* root;

class GT // Class Declaration

{

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.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;

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

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

{

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

cout << "Enter Name of Section " << j + 1 << " : ";

cin.get();

getline(cin, root->child[i]->child[j]->label);

}

}

}

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 << "\nChapter " << i + 1;

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

cout << "\nSections : ";

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

{

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

}

}

}

cout << endl;

}

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();

break; // Add this to avoid automatic fall-through

case 2:

gt.display(root);

break;

case 3:

cout << "Thanks for using this program!!!";

exit(1);

default:

cout << "Wrong choice!!!" << endl;

}

}

return 0;

}

**INPUT:**

1. Create

Enter name of book : Data Structures

Enter number of chapters in book : 2

Enter the name of Chapter 1 : Arrays

Enter number of sections in Chapter : Arrays : 2

Enter Name of Section 1 : Introduction to Arrays

Enter Name of Section 2 : Operations on Arrays

Enter the name of Chapter 2 : Linked Lists

Enter number of sections in Chapter : Linked Lists : 1

Enter Name of Section 1 : Types of Linked Lists

2. Display

**OUTPUT:**

-----Book Hierarchy---

Book title : Data Structures

Chapter 1 : Arrays

Sections :

Introduction to Arrays

Operations on Arrays

Chapter 2 : Linked Lists

Sections :

Types of Linked Lists