Page No.:
Date:

Experiment - 6

Aim - A book consists of chapters, chapters consist of sections and sections consist of subsections. Construct a tree & print the nodes. Find the time and space requirements of your method.

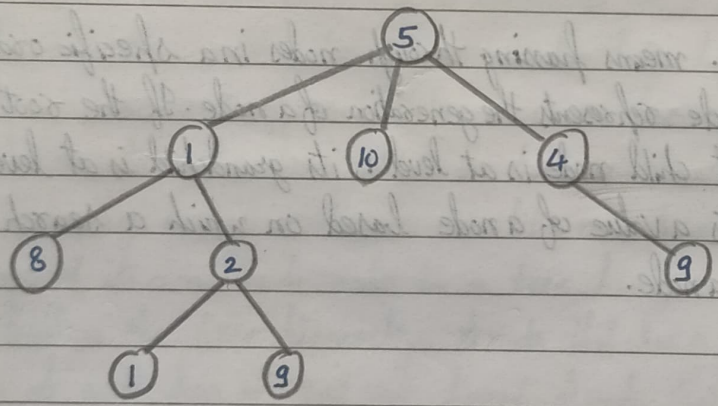
Theory:

Introduction to Tree:

Definition:

A tree T is a set of nodes storing elements such that the nodes have a parent-child relationship that satisfies the following.

- If T is not empty, T has a special tree called the root that has no parent
- Each node v of T different than the root has a unique parent node w; each node with parent w is a child of w



An internal node or inner node is any node of a tree that has child nodes and is thus not a leaf node.

There are two basic types of trees. In an unordered tree, a tree is a tree in a purely structural sense - that is to say, given a node, there is no order for the children of that node. A tree on which an order is imposed - for example, by assigning different natural numbers to each child of each node - is called an ordered tree, and data structures built on them are called ordered tree data structures. Ordered tree are by far the most common form of tree data structure. Binary search trees are one kind of ordered tree.

Page No.:
Date:

Important Terms

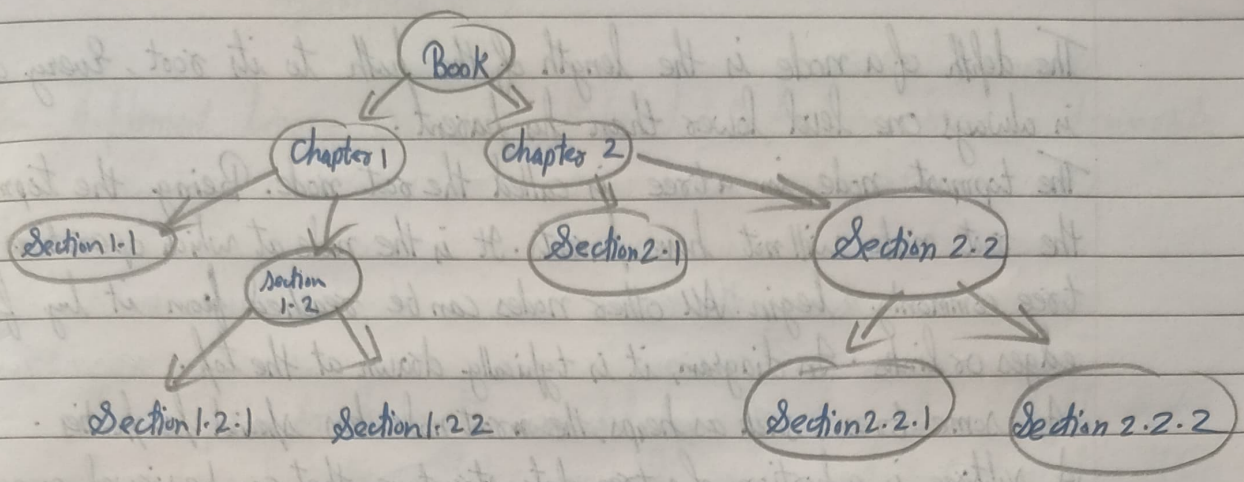
Following are the important terms with respect to tree.

- Path - Path refers to the sequence of nodes along the edges of a tree.
- Root - The node at the top of the tree is called root. There is only one root per tree and one path from the root node to any node.
- Parent - Any node except the root node has one edge upward to a node called parent.
- Child - The node below a given node connected by its edge downward is called its child node
- Leaf - The node which does not have any child node is called the leaf node.
- Subtree - Subtree represents the descendants of a node.
- Visiting - Visiting refers to checking the value of a node when control is on the node.
- Traversing - Traversing means passing through nodes in a specific order.
- Levels - Level of a node represents the generation of a node. If the root node is at level 0, then its next child node is at level 1, its grandchild is at level 2, and so on.
- Keys - Key represents a value of a node based on which a search operation is to be carried out for a node.

Advantages of trees

Trees are so useful & frequently used, because they have some very serious advantages:

- Trees reflect structural relationships in the data
- Trees are used to represent hierarchies
- Trees provide an efficient insertion & searching
- Trees are very flexible data, allowing to move subtrees around with minimum effort for this assignment we are considering the tree as follows.



Recursive definition

- T is either empty
- Or consists of a node r (the root) & a possibly empty set of trees whose roots are the children of r

Tree is a widely used data structure that emulates a tree structure with a set of linked nodes. The tree graphically is represented most commonly as on Picture 1. The circles are the nodes & the edges are the links between the tree are ~~usually~~ usually used to store and represent data in some hierarchical order. The data are stored in the nodes, from which the tree is consisted of.
A node may contain a value or a condition or represent a separate data structure or a tree of its own. Each nodes in a tree has zero or more child nodes, which are one level in the tree hierarchy. A node that has a child is called the child's parent node. A node has at most one parent. A node that has no child's in is called a leaf. and that node is of course at the bottommost level of the tree. The height of a node is the length of the longest path to a leaf from that node. The height of the root is the height of the tree. In other word, the "height" of tree is the number of levels" in the tree. or more formesly, the height of a tree is defined as follow:

1. The height of a tree with no element is 0
2. The height of a tree with 1 element is 1
3. The height of a tree with >1 element is equl to 1 + the height of its tallest subtree.

The depth of a node is the length of the path to its root. Every child node is always one level lower than his parent.
The topmost node in a tree is called the root node. Being the topmost node, the root node will not have parents. It is the node at which operations on the tree commonly begin. All other nodes can be reached from it by following edges or links. In diagram, it is typically drawn at the top.
In some trees, such as heaps, the root node has special properties.
A subtree is a portion of a tree data structure that can be viewed as a complete tree in itself. Any node in a tree T, together with all the nodes below his height. that are reachable from the node, comprise a subtree of T.
The subtree corresponding to the root nodes is the entire tree;
the subtree corresponding to any other node is called a proper subtree.
Every nodes in a tree can be seen as the root nodes of the subtree rooted at that node.

Input - Book name & its number of sections & subsections along with name.
Output - Formation of tree structure for book & its sections.
Conclusion - This program gives us the knowledge tree data structure.

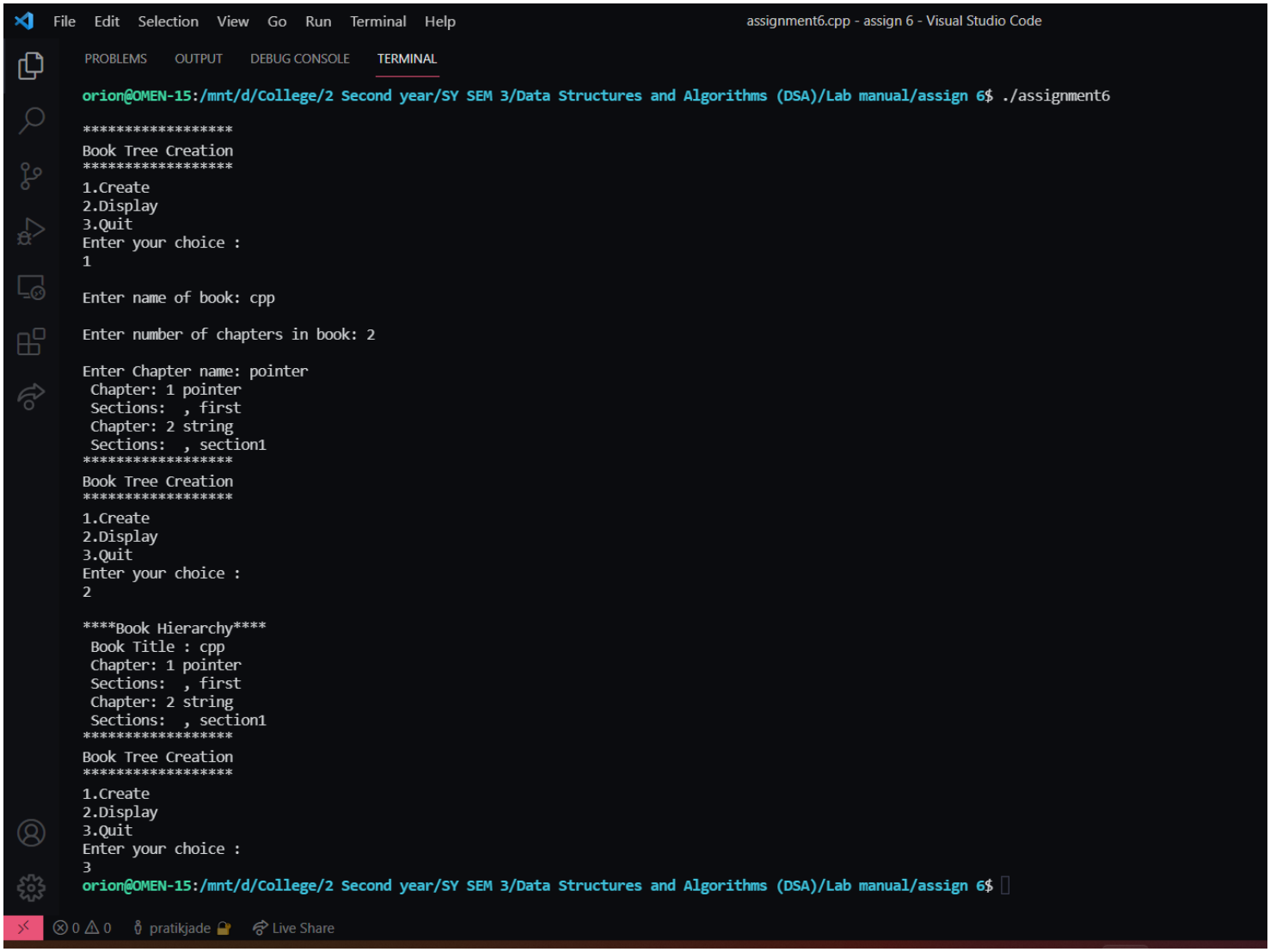
Program –

```
#include <iostream>
#include <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 BookT
{
public:
    void create_tree();
    void display(node *r1);
    BookT()
    {
        root = NULL;
    }
};
void BookT::create_tree()
{
    int tbooks, tchapters, i, j, k;
    root = new node();
    cout << "\nEnter name of book: ";
    cin >> root->label;
    cout << "\nEnter number of chapters in book: ";
    cin >> tchapters;
    root->ch_count = tchapters;
    for (i = 0; i < tchapters; i++)
    {
        root->child[i] = new node;
        cout << "\nEnter Chapter name: ";
        cin >> root->child[i]->label;
        cout << "\nEnter 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 << "\nEnter Section: " << j + 1 << " name: ";
            cin >> root->child[i]->child[j]->label;
        }
    }
}
void BookT::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++)
            {
                cout << " \n " << r1->child[i]->child[j]->label;
            }
        }
    }
}
//Main Contains Menu
int main()
{
    int choice;
    BookT BookT;
    while (1)
    {
        cout << "\n******************\n";
        cout << "Book Tree Creation";
        cout << "\n******************\n";
        cout << "1.Create" << endl;
        cout << "2.Display" << endl;
        cout << "3.Quit" << endl;
        cout << "Enter your choice : \n";
        cin >> choice;
        switch (choice)
        {
        case 1:
            BookT.create_tree();
        case 2:
            BookT.display(root);
            break;
        case 3:
            exit(1);
        default:
            cout << "Wrong choice" << endl;
        }
    }
}
```

Output-


File Edit Selection View Go Run Terminal Help
assignment6.cpp - assign 6 - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
orion@OMEN-15:/mnt/d/College/2 Second year/SY SEM 3/Data Structures and Algorithms (DSA)/Lab manual/assign 6$ ./assignment6
****
Book Tree Creation
****
1.Create
2.Display
3.Quit
Enter your choice :
1
Enter name of book: cpp
Enter number of chapters in book: 2
Enter Chapter name: pointer
Chapter: 1 pointer
Sections: , first
Chapter: 2 string
Sections: , section1
****
Book Tree Creation
****
1.Create
2.Display
3.Quit
Enter your choice :
2
****Book Hierarchy****
Book Title : cpp
Chapter: 1 pointer
Sections: , first
Chapter: 2 string
Sections: , section1
****
Book Tree Creation
****
1.Create
2.Display
3.Quit
Enter your choice :
3
orion@OMEN-15:/mnt/d/College/2 Second year/SY SEM 3/Data Structures and Algorithms (DSA)/Lab manual/assign 6$
0 0 pratikjade Live Share