```
/* 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.*/


#include<iostream>

#include<stdlib.h>

#include<string.h>


using namespace std;

struct node

{

    char name[20];

    node *next;

   node *down;

    int flag;

};

class Gll

{

char ch[20];    int n,i;

    node *head=NULL,*temp=NULL,*t1=NULL,*t2=NULL;

    public:

        node *create();    //to create node

        void insertb();    //to insert Book node

        void insertc();    //to insert chapter

       void inserts();     //to insert section

       void insertss();    //to insert sub-section

       void displayb();    //display tree(book)

};

node *Gll::create()

{

   node *p=new(struct node);       //to create new node i.e. p=new node

    p->next=NULL;

    p->down=NULL;

    p->flag=0;

   cout<<"\n enter the name";
```

```
            cin>>p->name;
             return p;
}

void Gll::insertb()     //to insert book name
{
       if(head==NULL)     //if no node created
        {
            t1=create();      //create node t1 & make it head node
            head=t1;
        }
      else
        {
          cout<<"\n book exist";    //else book node already created
        }
}

void Gll::insertc()       //insert chapter
{
                if(head==NULL)    //if there is no book node created
                 {
                    cout<<"\n there is no book";
                 }
        else
          {
                     cout<<"\n how many chapters you want to insert";
                   cin>>n;
                   for(i=0;i<n;i++)
                   {
                              t1=create();    //create t1 node
                           if(head->flag==0)  //head node not created chapter
                           {
                          head->down=t1;   head->flag=1
}
```

```
                    else                //already chapter is created
                    {
                            temp=head;
                        temp=temp->down;
                        while(temp->next!=NULL) //find down position to insert chapter until next!=NULL
                        temp=temp->next;
                        temp->next=t1;    //once we get next==NULL assign t1 to temp
                    }
                }
        }
}
void Gll::inserts()                 //insert section
{

        if(head==NULL)      // no book included
        {
                cout<<"\n there is no book";
          }
        else                        //book is already inserted
         {
        cout<<"\n Enter the name of chapter on which  you want to enter the section";
                cin>>ch;
                 temp=head;
                 if(temp->flag==0)    //no chapter created under head
                {
                    cout<<"\n their are no chapters on in book";
                }
                else                // chapters are available
                {
                        temp=temp->down;
                            while(temp!=NULL)
                              {
                          if(!strcmp(ch,temp->name))    //compare name of the chapter and name given by user
                          {
```

```
                                cout<<"\n how many sections you want to enter";
                                cin>>n;   //enter no. of sections
                                for(i=0;i<n;i++)
                                        {
                                  t1=create();  //create node for each section
                                        if(temp->flag==0)   //no sections are created
                                        {
                                                temp->down=t1;
                                        temp->flag=1;  cout<<"\n****";
                                        t2=temp->down;
                                        }
                                        else      //if already sections available
                                        {
                                        cout<<"\n#####";
                                         while(t2->next!=NULL)
                                        {
                                            t2=t2->next;      //search for next null node
                                        }
                                         t2->next=t1;      // insert newly created node i.e.t1  after t2
                                        }
                                }
                                break;     //after breaking no of sections included
                        }
                        temp=temp->next;    //search for next chapter name
                }   //close while
            }   //close else
        }        //close else
}             //function
void Gll::insertss()
{
        if(head==NULL)
        {
                cout<<"\n there is no book";
         }
```

```
        else
        {
 cout<<"\n Enter the name of chapter on which  you want to enter the section";  //ask for chapter
                    cin>>ch;
temp=head;
                    if(temp->flag==0)
                    {
                          cout<<"\n there are no chapters  in book";
                     }
                         else                          //if flag=1 i.e. chapter is available
                    {
                           temp=temp->down;      //search to down i.e. chapters
                             while(temp!=NULL)
                              {
                                        if(!strcmp(ch,temp->name))  //compare chapter name with ch(user entered)
                                       {
                                             cout<<"\n enter name of section in which you want to enter the sub
section";
                                             cin>>ch;     //ask for section name
                                           if(temp->flag==0)
                                             {
                                                       cout<<"\n there are no sections ";
                                                      }

                                              else
                                              {
                                                temp=temp->down;  //if chapter having sections then search down
                                             while(temp!=NULL)
                                             {
                                                       if(!strcmp(ch,temp->name))// compare section name is matched
                                                       {
                                                         cout<<"\n how many subsections you want to enter";
                                                       cin>>n;
                                                        for(i=0;i<n;i++)
```

```
                                        {
                                                t1=create();   //create node for ss
                                                if(temp->flag==0)
                                                {
                                                        temp->down=t1;
                                                temp->flag=1;  cout<<"\n****";
                                                t2=temp->down;
                                                }
                                                else   //already subsections is available
                                                {
                                                cout<<"\n#####";
                                                while(t2->next!=NULL)
                                                {
                                                        t2=t2->next;
                                                    }
                                                t2->next=t1;
                                                }
                                         }
                                        break;
                                    }
                                     temp=temp->next;   //search for next section name
                                   }
                               }
                          }
                  temp=temp->next;   //search for next chapter name
                }
          }
      }
}
void Gll::displayb()
{
if(head==NULL)
          {
cout<<"\n book not exist";
```

```
        }
        else
        {
                temp=head;
cout<<"\n NAME OF BOOK:  "<<temp->name;
                 if(temp->flag==1)
                {
                        temp=temp->down;

                         while(temp!=NULL)
                        {
cout<<"\n\t\tNAME OF CHAPTER:  "<<temp->name;
                   t1=temp;
                                 if(t1->flag==1)
                                {
                                  t1=t1->down;
                                while(t1!=NULL)
                                 {
cout<<"\n\t\t\t\tNAME OF SECTION:  "<<t1->name;
                                        t2=t1;
                                        if(t2->flag==1)
                                        {
                                                t2=t2->down;
                                           while(t2!=NULL)
                                                {
cout<<"\n\t\t\t\t\t\tNAME OF SUBSECTION:  "<<t2->name;
                                                        t2=t2->next;
                                                 }
                                        }
                                           t1=t1->next;
                                }
                                 }
                                 temp=temp->next;
                           }
```

```
                    }
            }
}
int main()
{
Gll g;
 int x;
          while(1)
           {
cout<<"\n\n enter your choice";
                     cout<<"\n 1.insert book";
                     cout<<"\n 2.insert chapter";
                    cout<<"\n 3.insert section";
                    cout<<"\n 4.insert subsection";
                    cout<<"\n 5.display book";
                    cout<<"\n 6.exit";
                    cin>>x;
                     switch(x)
                    {
case 1:       g.insertb();                          //to insert book name
                                 break;
                    case 2:      g.insertc();              //to insert chapter name
                                 break;
                    case 3:      g.inserts();              //to insert section name
                                   break;
                    case 4:      g.insertss();            //to insert sub section
                                 break;
                    case 5:      g.displayb();             //display book
                                  break;
                     case 6:     exit(0);
                    }
           }        return 0;
}
```

Output

enter your choice

1.insert book

2.insert chapter

3.insert section

4.insert subsection

5.display book

6.exit

1

enter the name DSA

enter your choice

1.insert book

2.insert chapter

3.insert section

4.insert subsection

5.display book

6.exit

2

how many chapters you want to insert 2

enter the name hashing

enter the name trees

enter your choice

1.insert book

2.insert chapter

3.insert section

4.insert subsection

5.display book

6.exit

3

Enter the name of chapter on which you want to enter the section hashing

how many sections you want to enter 3

enter the name collisionresolution

****

enter the name hashingfunc

#####

enter the name hashingtypes

#####

enter your choice

1.insert book

2.insert chapter

3.insert section

4.insert subsection

5.display book

6.exit

4

Enter the name of chapter on which you want to enter the section hashing

enter name of section in which you want to enter the sub section collisionresolution

how many subsections you want to enter 2

enter the name OpenHashing

****

enter the name ClosedHashing

#####

enter your choice

1.insert book

2.insert chapter

3.insert section

4.insert subsection

5.display book

6.exit

5

NAME OF BOOK: DSA

NAME OF CHAPTER: hashing

NAME OF SECTION: collisionresolution

NAME OF SUBSECTION: OpenHashing

NAME OF SUBSECTION: ClosedHashing

NAME OF SECTION: hashingfunc

NAME OF SECTION: hashingtypes

NAME OF CHAPTER: trees