Advanced Programming

Homework Assignment 10

Templates and Trees

General guidelines:

a. Maximize readability and ensure indentation.
b. Do exactly as required in the questions.
c. Every class in every question must be submitted in two seperate files – a header file (.h) and an implementation file (.cpp).
d. Add functions and helper methods as necessary to ensure readability. published in the course website – read them carefully!
e. Submission must be done according to the submission guidelines – a document
f. Use relevant names.
g. Use comments to document your functions and complex code parts. **Add an output example at the end of your file!**
h. Individual work and submission – paired submission is not allowed.

Important note: Unless otherwise specified, ever homework has no more than one week for submission.
Open submission boxes past the deadline are not permission for late submission.

Question 1:

In the lecture we implemented a class for representing binary trees. The class Tree is a variant of that class and is provided in the Moodle (In a folder under the tab General).

Add to the class Tree the following methods:

- **int leaves();** - returns the number of leaves in the tree. (A leaf is a node with no children.)
- **int height();** - returns the height of the tree. (An empty tree is height 0, A tree with only a root is height 1, etc.)
- **void reflect();** - swaps the children of every node, yielding a mirror image of the original tree.
- **int onlyLeftSon();** - returns the number of nodes that are left children with no siblings.

Question 2:

In the lab we implemented a class for representing binary search trees that inherits from the binary tree class.
Modify and complete your implementation and add the following methods:

- **void remove(T val);** - removes from the tree the node that contains the value val. **Note:** The removal must be done as learned in the Data Structures course using a helper function that finds the successor of the node.
- **int level(T val)** – returns the level in the tree of a node containing the value val. (The root is at level 0, its children at level 1 etc.; If val is not in the tree returns -1).

Question 3:

Combine your full implementations from the previous questions with the main program below to check their correctness:

```cpp
#include <iostream>
using namespace std;
#include "SearchTree.h"

int main()
{
       SearchTree<int> T1;
       cout<<"enter 10 numbers\n";
       int x,y;
       for (int i=0;i<10; i++)
       {
              cin>>x;
              T1.add(x);
       }
       cout<<"inorder: ";
       T1.inOrder();
       cout<<"\nenter 0-6:\n";
       cin>>x;
       while(x!=0)
       {
              switch (x)
              {
              case 1: cout<<"# of leaves: "<<T1.leaves()<<endl;
                     break;
              case 2: cout<<"height of tree: "<<T1.height()<<endl;
                     break;
              case 3:T1.reflect();
                            cout<<"reflected tree: ";
                            T1.inOrder();
                            T1.reflect();
                            cout<<endl;
                     break;
              case 4: cout<<"# left sons only: "<<T1.onlyLeftSon()<<endl;
                     break;
              case 5: cout<<"enter a number ";
                     cin>>y;
                     cout<<"level of "<<y<<" on tree: "<<T1.level(y)<<endl;
                     break;
              case 6: cout<<"enter a number ";
                     cin>>y;
                     T1.remove(y);
                     cout<<"after removing "<<y<<": ";
                     T1.inOrder();
                     cout<<endl;
              }
       cout<<"enter 0-6:\n";
       cin>>x;
       }
       return 0;
}
```

Question 4:

Declare and implement a class Student that represents a student.

The class must have the following fields:

- Id number
- Last name
- First name

The class must have (at least) the following methods:

- Assignment constructor
- Operators <=, =, <<, >>
- Other operators as necessary

Combine the class with the binary tree and binary search tree classes from the previous questions to create a program that manages registration of students to a college. The students are to be stored in a binary search tree.

Use the following main program to test your code:

```
#include <iostream>
using namespace std;
#include "SearchTree.h"
#include "Student.h"
int main(){
    SearchTree<Student> sList;
    Student tmp;
    char choice = 'i';
    while (choice != 'e'){
        cout<<"enter a-e\n";
        cin>>choice;
        switch (choice){
        case 'a':
            cout<<"enter a student\n";
            cin>>tmp;
            sList.add(tmp);
            break;
        case 'b':
            cout<<"enter a student\n";
            cin>>tmp;
            sList.remove(tmp);
            break;
        case 'c':
            cout<<"enter a student\n";
            cin>>tmp;
            if(sList.search(tmp))
                cout<<"exist\n";
            else
                cout<<"not exist\n";
            break;
        case 'd':
            sList.inOrder();
            break;
        case 'e':
            break;

        default:
            cout<<"error\n";
            break;
        }
    }
    return 0;
}
```

Good Luck!