CSC230 Lab 11

Due: Apr. 24th, 11:59pm

Goal: This lab will teach you about tree operations.

Part I

In our lecture of tree, several different traversal were introduced. They are inorder traversal, preorder traversal, postorder traversal, and level-order traversal.

It is surprisingly simple to write recursive function to implement inorder traversal, preorder traversal, and postorder traversal. One sample implementation of inorder traversal was provided in the slides. It is listed as the follows.

```
/** Print BST root in alphabetic order */
template <class T>
void show(TreeNode<T>* root){
  if(root == nullptr) return;
  show(root->getLeft());
  cout << root->getDatum() << endl;
  show(root->getRight());
}
```

In this lab, implement a **non-recursive** version or inorder traversal for binary tree. Please implement the nonRecInorder() function in lab11.cpp file. This function should use stack STL defined by C++. Please use the STL, do NOT use the stack defined by yourself. It will make debugging easy.

The algorithm of this non-recursive inorder traversal is listed as follows.

```
Create an TreeNode* stack
while(true)
   while(root is not nullptr){
      push(root)
      root = root->left;
   }

   if(stack is empty) break;

   root = top();
   pop();

   print out the datum of root

   root = root->right;
}
```

Requirements and Hints:

- You should use empty(), top(), pop(), push() functions of stack STL
- The idea of the algorithm is to push the left subtree of a node to stack, pop the node, print out the value of the node, then push the right subtree to the stack

Part II

The second part of this lab is to implement level traversal of binary tree. In this part, please do NOT use any recursion either. The idea of Level traversal is to

- Visit the root
- When traversing level l, keep all the elements at level l+1 in **queue**
- Repeat this process until all levels are completed

Please implement levelOrder() function in lab11.cpp file. The implementation of this function uses queue STL of C++. Please make sure that you use the queue STL, do NOT use your own queue definition.

The algorithm of level order traversal is listed as follows.

```
Create a queue
If(root is not nullptr) push root to the queue

While(queue is not empty){

    temp = Dequeu()

    Print out datum of temp

    If left pointer of temp is not nullptr, push it to the queue

    If right pointer of temp is not nullptr, push it to the queue
}
```

Requirements and Hints:

- You should use front(), push(), pop() functions of queue STL.

Once you finish the coding, type the following command to compile it.
g++ lab11.cpp

The execution result of test.cpp is:

```
jikaili$ ./a.out
```

```
Inorder traversal result:
9
9
4
100
7
10
Level traversal result:
100
9
10
9
4
7
```

Your implementation must have the exactly same result.

Wrap up

Jar you C++ files and the downloaded data files into lab11.jar. Submit the completed file to Canvas.