HW8 실습: 트리 순회

(과제 아닙니다, 제출하지 않습니다.)

질문

pemds81718@gmail.com

실습 파일 만들기

1. makefile

```
run: hw8
    ./hw8 < input.in

hw8: ./src/*
    g++ -std=c++14 -Wall -Wextra -g ./src/hw8.cpp -o hw8

clean:
    rm -f hw8
```

2. input.in

```
35.3 15.7 81.5 4.5 66.7 91.2 2.3 5.2 88.2 94.5
```

3. src/hw8.cpp

```cpp
#include <iostream>

#include "BinaryTree.h"

void visit(double& value) { std::cout << value << " "; }

int main() {
  BinaryTree<double> tree;
  double value;

  std::cout << "Enter doubles:\n";
  while (std::cin >> value) {
    tree.add(value);
  }

  std::cout << "Preorder traversal: ";
  tree.preorderTraverse(visit);
  std::cout << std::endl;

  std::cout << "Inorder traversal: ";
  tree.inorderTraverse(visit);
  std::cout << std::endl;

  std::cout << "Postorder traversal: ";
  tree.postorderTraverse(visit);
  std::cout << std::endl;

  return 0;
}
```

4. src/BinaryTree.h

```
#ifndef __BINARYTREE_H__
#define __BINARYTREE_H__
#include <memory>

#include "TreeNode.h"

template <typename T>
class BinaryTree {
  using TreePtr = std::shared_ptr<TreeNode<T>>;

 private:
  TreePtr rootPtr;

protected:
  int getHeightHelper(TreePtr subTreePtr) const;

  // 재귀적으로 새로운 노드를 좌 -> 우 순서로 추가하여 트리의 균형을 유지한다.
  auto balancedAdd(TreePtr subTreePtr, TreePtr itemNodePtr);

  // oldTreeRootPtr 를 루트로 하는 트리를 복사하고 그 복사본의 포인터를 반환한다.
  TreePtr copyTree(const TreePtr oldTreeRootPtr) const;

  // 재귀 순회 헬퍼 함수
  void preorderHelper(std::function<void(T&)> visit, TreePtr treePtr) const;
  void inorderHelper(std::function<void(T&)> visit, TreePtr treePtr) const;
  void postorderHelper(std::function<void(T&)> visit, TreePtr treePtr) const;

 public:
  /*
  생성자
  */
  BinaryTree();
  BinaryTree(const T& rootItem);
  BinaryTree(const T& rootItem, const TreePtr leftTreePtr,
             const TreePtr rightTreePtr);
  BinaryTree(const BinaryTree<T>& aTree);
  /*
  Interface
  */
  int getHeight() const { return getHightHelper(rootPtr); }
  bool add(const T& newData);
  /*
  순회 함수
  */
```

```
  void preorderTraverse(void visit(T&)) const {
    preorderHelper(visit, rootPtr);
  };
  void inorderTraverse(void visit(T&)) const { inorderHelper(visit, rootPtr); };
  void postorderTraverse(void visit(T&)) const {
    postorderHelper(visit, rootPtr);
  };
  // 추가 실습(optional)
  // void iterativeInorderTraverse(std::function<void(T&)> visit) const;
};

#include "BinaryTree.cpp"

#endif  // __BINARYTREE_H__
```

5. TreeNode.h

```
#ifndef __TREENODE_H__
#define __TREENODE_H__
#include <memory>

template <typename T>
class TreeNode {
 private:
  T item;
  std::shared_ptr<TreeNode<T>> leftChildPtr;
  std::shared_ptr<TreeNode<T>> rightChildPtr;

 public:
  TreeNode();
  TreeNode(const T& anItem);
  TreeNode(const T& anItem, std::shared_ptr<TreeNode<T>> leftPtr,
           std::shared_ptr<TreeNode<T>> rightPtr);

  void setItem(const T& nodeItem);
  T getItem() const;

  void setLeftChildPtr(std::shared_ptr<TreeNode<T>> leftPtr);
  void setRightChildPtr(std::shared_ptr<TreeNode<T>> rightPtr);

  auto getLeftChildPtr() const;
  auto getRightChildPtr() const;
};  // end TreeNode

#include "TreeNode.cpp"
#endif  // __TREENODE_H__
```

6. BinaryTree.cpp 과 TreeNode.cpp

헤더에 선언한 함수 구현하기

7. 결과

컴파일 및 실행: make

실행 결과 예시

```
~/ex8 ❯ make
g++ -std=c++14 -Wall -Wextra -g ./src/hw8.cpp -o hw8
./hw8 < input.in
Enter doubles:
Preorder traversal: 35.3 15.7 4.5 2.3 91.2 94.5 81.5 66.7 88.2 5.2
Inorder traversal: 2.3 4.5 15.7 94.5 91.2 35.3 88.2 66.7 81.5 5.2
Postorder traversal: 2.3 4.5 94.5 91.2 15.7 88.2 66.7 5.2 81.5 35.3
```

구조 참고

```
.
├── .vscode
│   └── ....
├── input.in
├── makefile
└── src
    ├── BinaryTree.cpp
    ├── BinaryTree.h
    ├── TreeNode.cpp
    ├── TreeNode.h
    └── hw8.cpp
```