

자료구조론

Homework # 4

이름 : 윤성호

학번 : 12161756

분반 : 004

제출일 : 2019/06/08

- 개요 -

(1-2) preorder, inorder, postorder 출력

1. 구현상 특징 : p2 ~ p3
2. 사용한 Node 값 (n=11) : p4
3. 실행 화면 : p5

(1-2) preorder, inorder, postorder 출력

1. 구현상 특징

```

1 // 12161756 윤성호
2 #pragma once
3 #include <iostream>
4 using namespace std;
5 template <typename T> class BST;
6
7 template <typename T>
8 class Node { // 노드의 값과 left & right child를 지정하는 Node 클래스
9     friend class BST<T>;
10 private:
11     T val;
12     Node* left;
13     Node* right;
14 public:
15     Node(T v = 0, Node* l = NULL, Node* r = NULL) {
16         this->val = v;
17         this->left = l;
18         this->right = r;
19     }
20 };

```

```

22 template <typename T>
23 class BST {
24 private:
25     Node<T>* root;
26 public:
27     BST(T val = 0) {
28         root = new Node<T>(val);
29     }
30
31     Node<T>* getRoot() {
32         return root;
33     }
34
35     void visit(Node<T>* stNode) {
36         cout << stNode->val << " ";
37     }
38
39     void insert(Node<T>* n) {
40         if (search(root, n->val) == NULL) { // insert될 값이 현재 트리에 없을 때만 진행
41             Node<T>* stNode = root;
42             Node<T>* parent = stNode;
43
44             while (stNode != NULL) { // 기준 node가 Null값이 아닐 동안 진행
45                 parent = stNode; // parent에 stNode를 대입
46
47                 if (n->val < parent->val) // insert 대상 < parent, left child로 이동
48                     stNode = stNode->left;
49                 else // insert 대상 > parent, right child로 이동
50                     stNode = stNode->right;
51             } // while 문 완료, 현재 parent : external node
52
53             if (n->val < parent->val) // insert 대상 < parent,
54                 parent->left = n; // parent의 left child로 insert될 노드 지정
55             else // insert 대상 > parent,
56                 parent->right = n; // parent의 right child로 insert될 노드 지정
57         }
58     }
59
60

```

```

tree.h
HW4
BST<T>
61 Node<T>* search(Node<T>* stNode, T v) {
62     if (stNode == NULL) // 기준 node = NULL,
63         return NULL; // NULL 리턴
64
65     else if (v == stNode->val) // search 대상 = 기준 node,
66         return stNode; // 기준 node 리턴
67
68     else if (v < stNode->val) // search 대상 < 기준 node,
69         search(stNode->left, v); // 기준 node의 left child를 search
70
71     else // search 대상 > 기준 node,
72         search(stNode->right, v); // 기준 node의 right child를 search
73 }
74
75 void preorder(Node<T> * stNode) { // Root -> Left -> Right
76     if (stNode != NULL) {
77         visit(stNode);
78         preorder(stNode->left);
79         preorder(stNode->right);
80     }
81 }
82
83 void inorder(Node<T> * stNode) { // Left -> Root -> Right
84     if (stNode != NULL) {
85         inorder(stNode->left);
86         visit(stNode);
87         inorder(stNode->right);
88     }
89 }
90
91 void postorder(Node<T> * stNode) { // Left -> Right -> Root
92     if (stNode != NULL) {
93         postorder(stNode->left);
94         postorder(stNode->right);
95         visit(stNode);
96     }
97 }
98

```

110 % 문제 검색되지 않음

- template 사용으로 인해 header 파일 하나에 선언 및 구현을 같이했습니다.

2. 사용한 Node 값 (n=11)

55*

17

13

76

19

61

43

18

65

89

10

(55 의 경우 BST 클래스의 생성자에 의해 생성된 root 값입니다.)

3. 실행 화면 : p5

```

1 // 12161756 윤성호
2 #include "tree.h"
3
4 int main() {
5     cout << "12161756 윤성호" << endl;
6
7     BST<int> exBST(55); // 55의 root를 가진 tree 객체 생성
8     exBST.insert(new Node<int>(17)); // 17 insert
9     exBST.insert(new Node<int>(13));
10    exBST.insert(new Node<int>(76));
11    exBST.insert(new Node<int>(19));
12    exBST.insert(new Node<int>(61));
13    exBST.insert(new Node<int>(43));
14    exBST.insert(new Node<int>(18));
15    exBST.insert(new Node<int>(65));
16    exBST.insert(new Node<int>(89));
17    exBST.insert(new Node<int>(10)); // ~10 insert
18
19    cout << "<insert 순서>" << endl;
20    cout << "55(R) 17 13 76 19 61 43 18 65 89 10" << endl << endl;
21
22    cout << "<preorder>" << endl;
23    exBST.preorder(exBST.getRoot()); // preorder 출력
24    cout << endl << endl; // 55 17 13 10 19 18 43 76 61 65 89
25
26    cout << "<inorder>" << endl;
27    exBST.inorder(exBST.getRoot()); // inorder 출력
28    cout << endl << endl; // 10 13 17 18 19 43 55 61 65 76 89
29
30    cout << "<postorder>" << endl;
31    exBST.postorder(exBST.getRoot()); // postorder 출력
32    cout << endl; // 10 13 18 43 19 17 65 61 89 76 55
33
34    system("pause");
35    return 0;
36 }

```

```

C:\Users#grymp\Desktop\...
12161756 윤성호
<insert 순서>
55(R) 17 13 76 19 61 43 18 65 89 10

<preorder>
55 17 13 10 19 18 43 76 61 65 89

<inorder>
10 13 17 18 19 43 55 61 65 76 89

<postorder>
10 13 18 43 19 17 65 61 89 76 55
계속하려면 아무 키나 누르십시오 . . .

```