Xilinx Zynq FPGA, TI DSP, MCU 기반의 프로그래밍 및 회로 설계 전문가 과정

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – 장성환
redmk1025@gmail.com

* TREE

스택이나 큐 같은 구조는 자료의 갯수가 많을 경우에 탐색의 시간이 너무 많다.
따라서 트리구조를 만들게 된다.

기본트리
AVL 트리 - 검색 속도가 가장 빠르다.
RB 트리 - 구글 등 사용

```
#include <stdio.h>
#include <stdlib.h>

 typedef struct _tree{
    int data;
    struct _tree *left;
    struct _tree *right;
}tree;

tree* get_node(){
    tree * tmp=(tree*)malloc(sizeof(tree));
    tmp->left=NULL;
    tmp->right=NULL;
    return tmp;
}


void tree_ins(tree **root, int data){
    if(*root==NULL){
        *root = get_node();
        (*root)->data =data;
        return;
    }
    else if((*root)->data >data)
        tree_ins(&(*root)->left,data);
```

```
    else if((*root)->data < data)
        tree_ins(&(*root)->right , data);
} //트리 삽입

void print_tree(tree *root){
    if(root){
        printf("%d\n",root-> data);
/*
        if(root->left)
            printf(“left=%d, “,root->left->data);
        else
            printf(“left = NULL, “);

        if(root->right)
            printf(“left=%d, “,root->right->data);
        else
            printf(“left = NULL, “);
*/
        print_tree(root->left);
        print_tree(root->right);
    }
} // 트리 표현

tree *chg_node(tree *root){
    tree *tmp =root;

    if(!root->right)
        root = root->left;
    else if(!root->left)
        root = root->right;
    free(tmp);

    return root;
} //chg_node 함수
```

```
tree *find_max(tree *root, int *data){
    if(root->right)
        root->right = find_max(root->right, data);
    else{
        *data = root->data;
        root = chg_node(root);
    }
    return root;
} //find max 함수

tree *delete_tree(tree *root, int data){
    int num;
    tree *tmp;
    if(root == NULL){
        printf("Not Found\n");
        return NULL;
    }
    else if(root->data > data){
        root->left = delete_tree(root->left,data);
    }
     else if(root->data <data){
        root->right = delete_tree(root->left,data);
    }
     else if(root->left && root->right){
        root->left = find_max(root->left,&num);
        root->data = num;
    }
    else
        root = chg_node(root);
    return root;
} // 트리 삭제 함수
```

```
int main(void){
    int i;
    int data[14]={50,45,73,32,48,46,16,37,120,47,130,127,124};
    tree *root = NULL;
    for(i=0;data[i];i++){
        tree_ins(&root,data[i]);
    }

    print_tree(root);
    printf("\n");

    delete_tree(root,50);
    print_tree(root);

    return 0;
}
```

---

* tree 삭제의 요건

(*root)→data>data
(*root)→data<data //두 조건은 순회용
두 조건에 걸리지 않으면 data 를 찾았다는 것이다.

(*root)→left && (*root) →right //서브 트리면
find_max 함수를 사용 (left 를 넘기고 right 에 null 이 나올때 까지 재귀)

else
chg_node 함수를 사용

체인지 노트 함수를 분석하면,
if(!root->left)

return root→right;
if(!root->right)
return root→left; 의 알고리즘을 이용한다.