TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

10회차 (2018-03-07)

강사 - Innova Lee(이상훈)
gcccompil3r@gmail.com

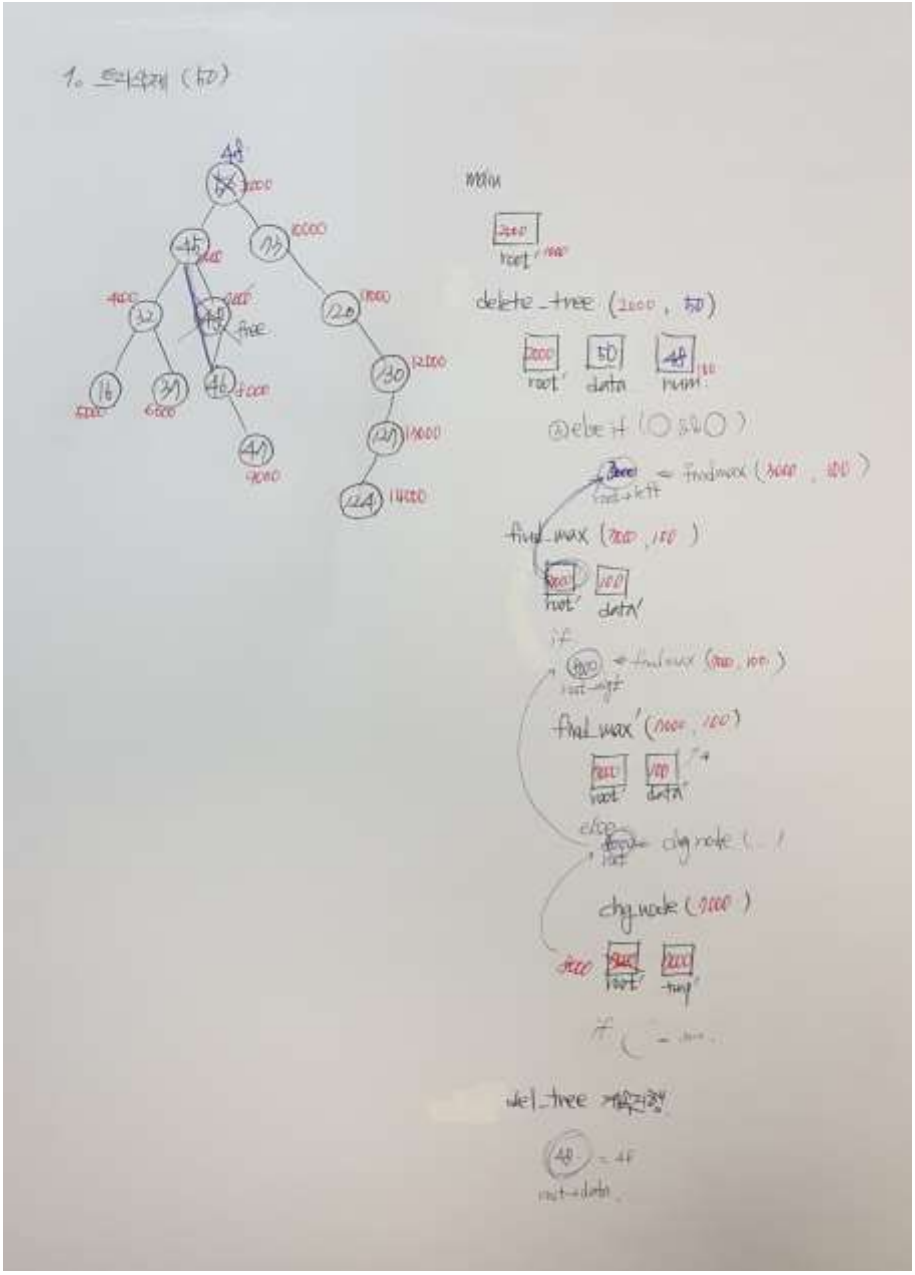
학생 - 정유경
ucong@naver.com

Binary Tree.c

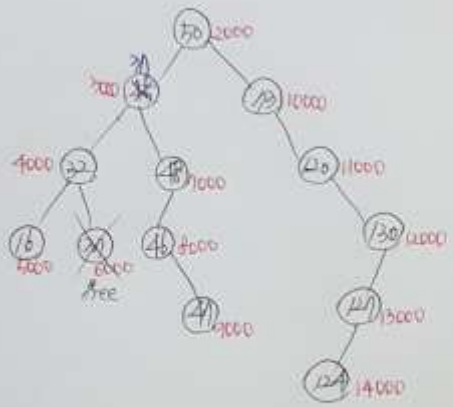
```c
#include <stdio.h>
#include <stdlib.h>

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

tree *get_node(void)
{
	tree *tmp;
	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);
}

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

		print_tree(root->left);
		print_tree(root->right);
	}
}

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;
}
```

```
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;
}

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->right, data);
	else if (root->left && root->right)
	{
		root->left = find_max(root->left, &num);
		root->data = num;
	}
	else
		root = chg_node(root);
	return root;
}

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);
	delete_tree(root, 50);
	printf("After Delete\n");
	print_tree(root);
	return 0;
}
```

<Delete_tree : 45, 50,127에 대해 이해하기>



2. 트리삭제 (45)

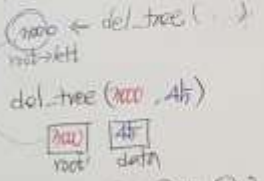


main

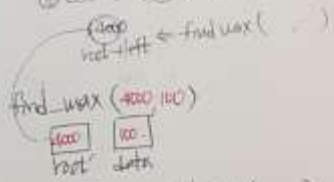
del_tree (2000, 45)

50 > 45

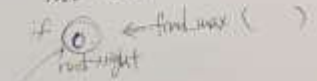
① else if



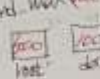
② else if ( ) && ( )



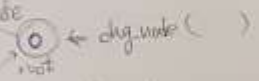
if



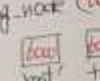
find_max (6000, 100)



else



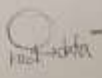
chg_node (6000)



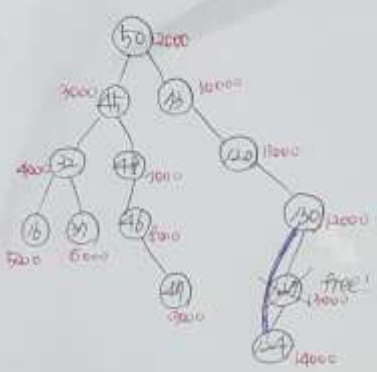
if

free (6000)

del_tree 재귀종료



3. 트리삭제 (40)



main

root

del_tree (root, 40)

root data main

① else if 40 < 44

← del_tree( )

root->right

del_tree (root, 40)

root data

② else if 40 < 46

← del_tree( )

del_tree (root, 40)

③ else if 40 < 49

← del_tree( )

del_tree (root, 40)

④ else if 130 > 49

← del_tree( )

root->left

del_tree (root, 40)

⑤ else

← chg_node(root)

root

chg_node (root)

root temp

if

del_tree 끝

root->data