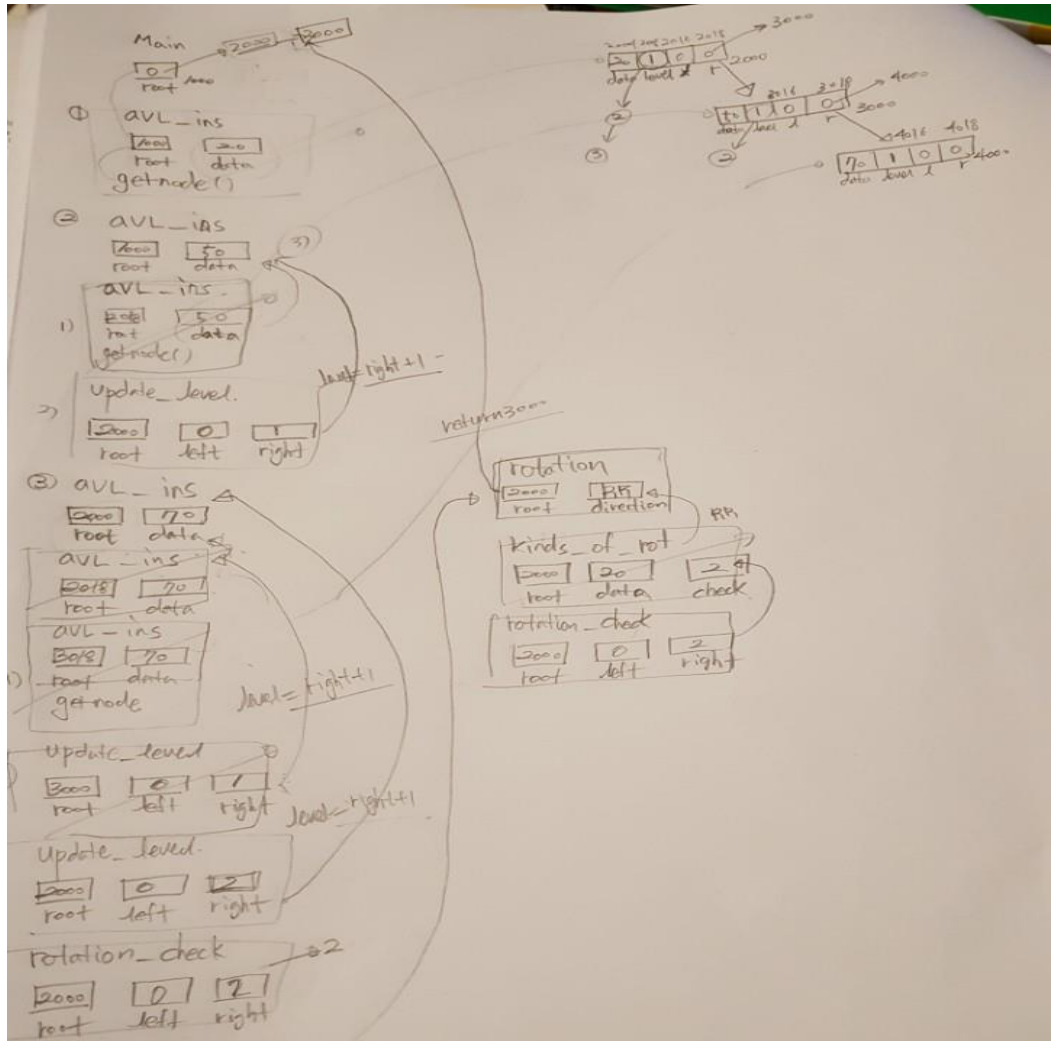
TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – GJ (박현우)
uc820@naver.com

1.AVL TREE 코드 이해 1(그림 그리기)

✓ 이진트리
사용 이유 :검색을 '빨리'하기 위해서 사용

✓ AVL트리
사용 이유 : '완전한 이진 트리'를 구성하기 위해서 사용



20 -> 50 -> 70 순으로 입력

1.AVL TREE 코드 이해 2(코드)

```
#include<stdio.h>
#include<stdlib.h>
#include<malloc.h>
#define EMPTY 0

typedef enum __rot
{ RR,
RL,
LL,
LR
}rot;

typedef struct node{
int level;
int data;
struct node *left;
struct node *right;
}avl;

avl *getnode(void){
avl *tmp;
tmp = (avl *)malloc(sizeof(avl));
tmp->left = EMPTY;
tmp->right = EMPTY;
tmp->level = 1;
return tmp;
}

int update_level(avl *root){

int left = root->left ? root->left->level : 0;
int right = root->right ? root->right->level :0;
if(left > right)
return left + 1;

return right + 1;
}
int rotation_check(avl *root){

int left = root->left ? root->left->level : 0;
int right = root->right ? root->right->level :0;
return right - left;
}
```

```
int kinds_of_rot(avl *root, int data){
int check = rotation_check(root);
printf("data = %d₩n", data);
// for RR and RL
if(check > 1){
if(root->right->data > data)
return RL;

return RR;
}
//for LL and LR
else if(check < -1){
if(root->left->data < data)
return LR;

return LL;
}
}
avl *rr_rot(avl *root, avl *right){

avl *tmp = root;
right->left = getnode();
right->left->data = root->data;
free(tmp);

return right;
}
avl *rl_rot(avl *root, avl *right){


return root;
}
avl *lr_rot(avl *root, avl *left){

return left;
}
avl *ll_rot(avl *root, avl *left){

avl *tmp = root;
left->right = getnode();
left->right->data = root->data;
free(tmp);

return left;
}
```

```
avl *rotation(avl *root,int
direction){
avl *tmp = root;

switch(direction){
case LL:
printf("LL Rotation₩n");
return ll_rot(root, root->left);
case RL:
printf("RL Rotation₩n");
return rl_rot(root, root->left);
case RR:
printf("RR Rotation₩n");
return rr_rot(root, root->right);
case LR:
return lr_rot(root, root->right);
printf("LR Rotation₩n");
}
}

void avl_ins(avl **root, int data){

if(!(*root)){
(*root) = getnode();
(*root)->data = data;
return;
}
if((*root)->data > data)
avl_ins( &(*root)->left, data);
else if((*root)->data < data)
avl_ins( &(*root)->right, data);

(*root)->level =
update_level(*root);

if( abs(rotation_check(*root)) >1
){
printf("Rotation₩n");
*root = rotation(*root,
kinds_of_rot(*root, data));
}
}
```

```
void print_avl(avl *root){
if(root){
printf("data = %d  level = %d₩n",root->data, root-
>level);
print_avl(root->left);
print_avl(root->right);
}
}
int main(void){
avl *root = EMPTY;
int data[] = {3,2,1};
int i, size = 0;
size = sizeof(data)/sizeof(int);
for(i=0; i<size; i++){
avl_ins(&root, data[i]);
}
print_avl(root);
return 0;
}
```