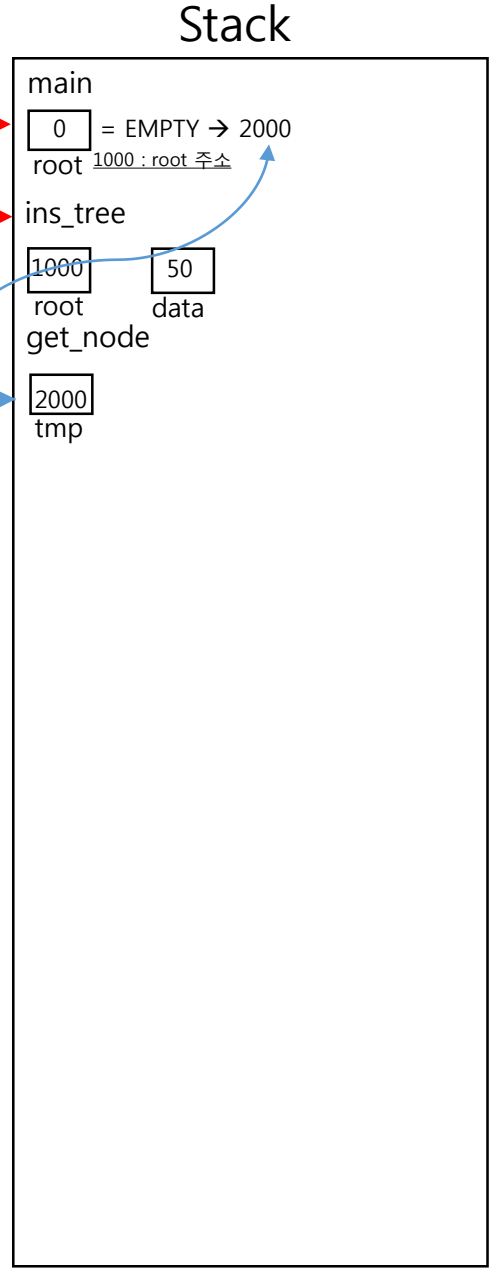
TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

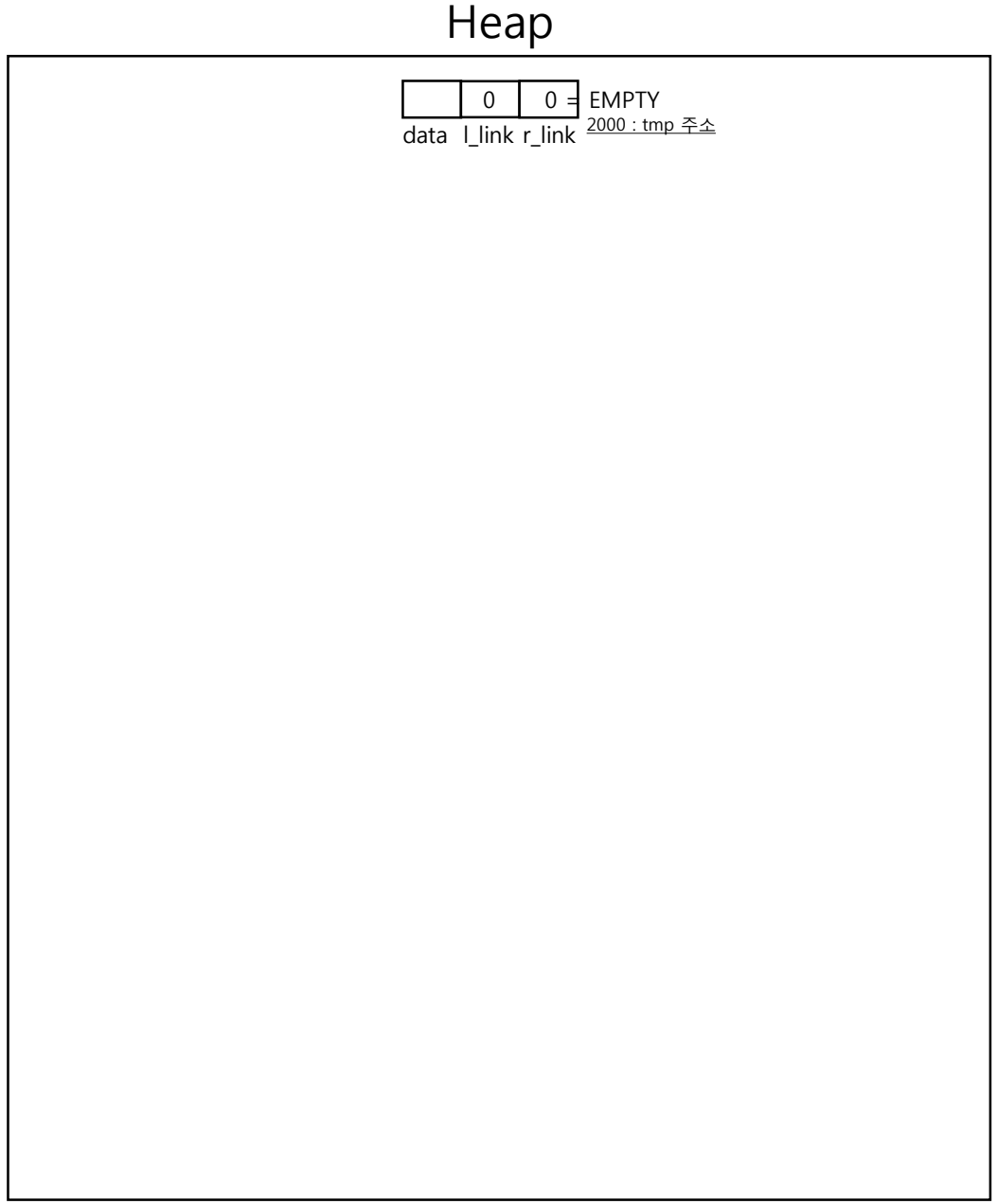
강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – GJ (박현우)
uc820@naver.com

1. 트리(Tree) 코드 이해(그림 그리기) – (ins_tree 1)

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

#define EMPTY 0

struct node{

int data;

struct node *l_link;
struct node *r_link;

};

typedef struct node Tree;

Tree *getnode(){

Tree *tmp = (Tree *)malloc(sizeof(Tree));
tmp->l_link = EMPTY;
tmp->r_link = EMPTY;

return tmp;

}

void ins_tree(Tree **root, int data){

if( *root == EMPTY){

*root = getnode();
(*root)->data = data;
return;
}

if( (*root)->data < data)
ins_tree( &((*root)->r_link), data);

if( (*root)->data > data)
ins_tree( &((*root)->l_link), data);

}
```

```
void print_tree(Tree *root){

Tree *tmp = root;

if(tmp != NULL){

printf("%d₩n", tmp->data);

/* if(root->l_link)
printf("left = %d, ", root->l_link->data);
else
printf("left = NULL, ");

if(root->r_link)
printf("left = %d, ", root->r_link->data);
else
printf("left = NULL, ");
*/

print_tree(tmp->l_link);
print_tree(tmp->r_link);
}

}
int main(void){

Tree *root = EMPTY;
int data[13] = {50, 45, 73, 32, 48, 46, 16, 37,
120, 47, 130, 127, 124};
int i, num;

num = sizeof(data)/sizeof(int);

for( i=0; i<num; i++){
ins_tree(&root, data[i]);
}

print_tree(root);

printf("after del₩n");
root = del_tree(root,32);

print_tree(root);

return 0;
}
```

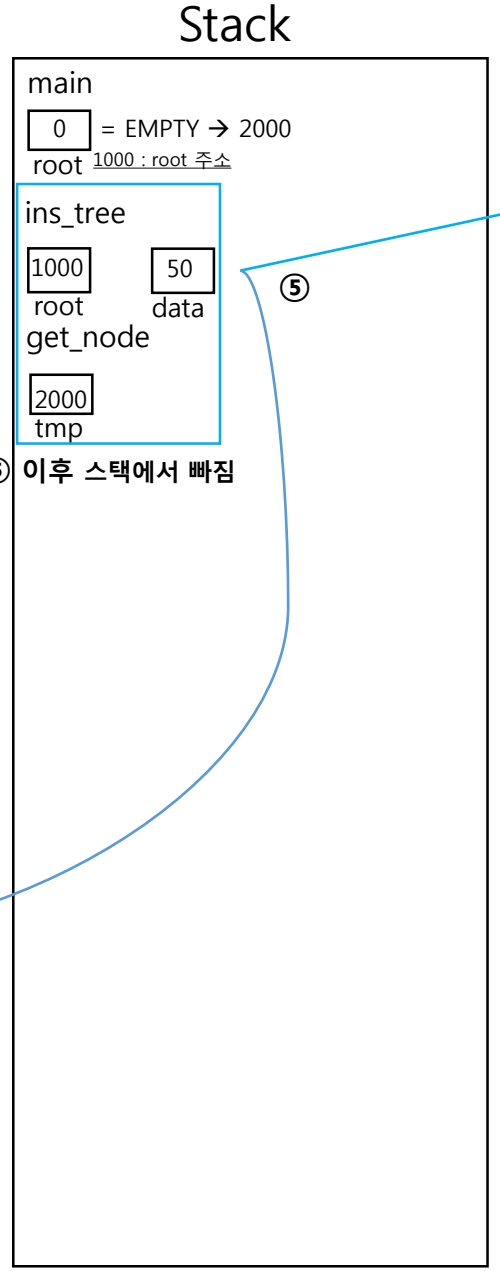


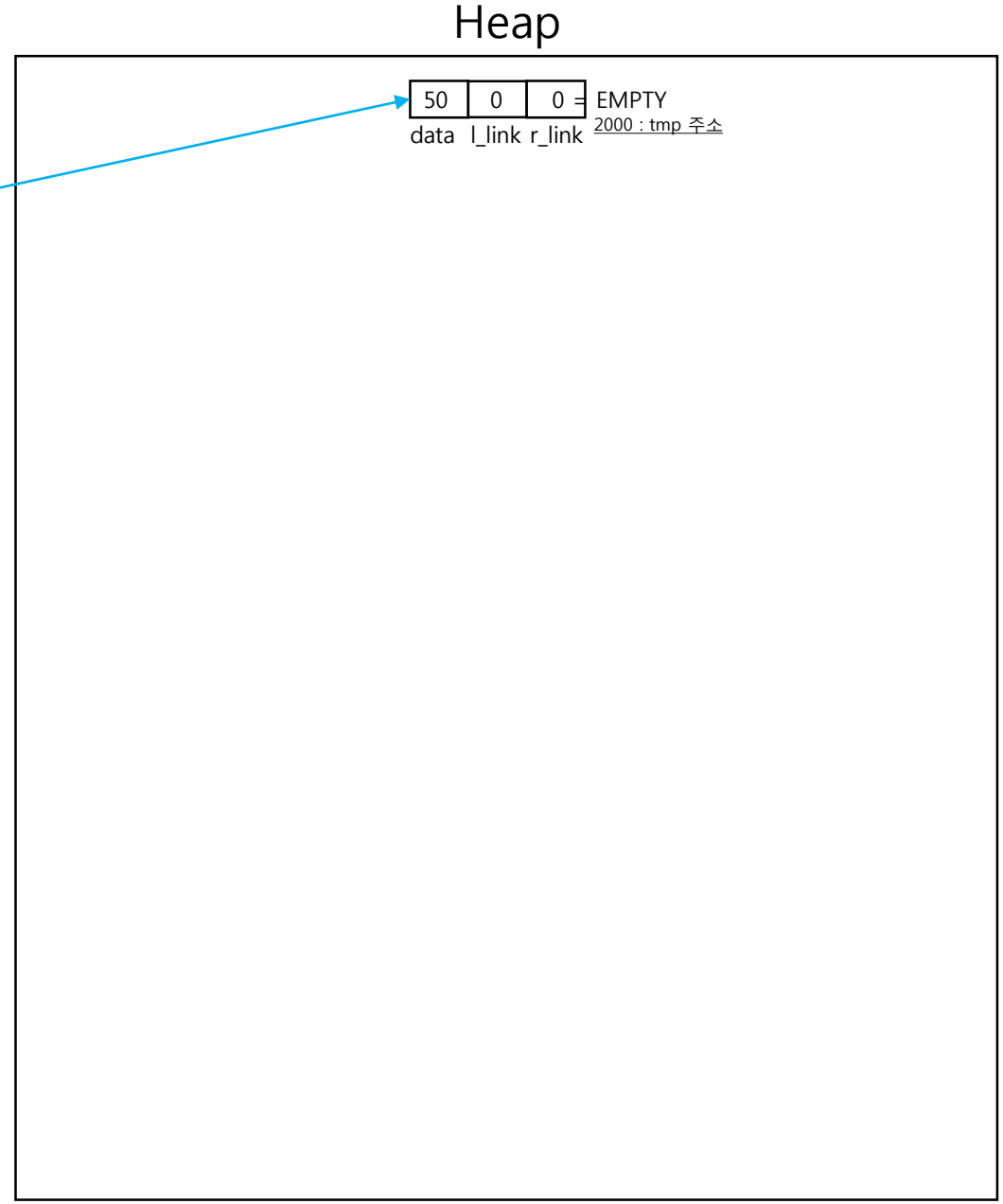


1. 트리(Tree) 코드 이해(그림 그리기) – (ins_tree 2)

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

#define EMPTY 0

struct node{

int data;

struct node *l_link;
struct node *r_link;

};

typedef struct node Tree;

Tree *getnode(){

Tree *tmp = (Tree *)malloc(sizeof(Tree));
tmp->l_link = EMPTY;
tmp->r_link = EMPTY;

return tmp;

}

void ins_tree(Tree **root, int data){

if( *root == EMPTY){

*root = getnode();
(*root)->data = data;
return;
}

if( (*root)->data < data)
ins_tree( &((*root)->r_link), data);

if( (*root)->data > data)
ins_tree( &((*root)->l_link), data);

}
```

```
void print_tree(Tree *root){

Tree *tmp = root;

if(tmp != NULL){

printf("%d₩n", tmp->data);

/* if(root->l_link)
printf("left = %d, ", root->l_link->data);
else
printf("left = NULL, ");

if(root->r_link)
printf("left = %d, ", root->r_link->data);
else
printf("left = NULL, ");
*/

print_tree(tmp->l_link);
print_tree(tmp->r_link);
}

}
int main(void){

Tree *root = EMPTY;
int data[13] = {50, 45, 73, 32, 48, 46, 16, 37,
120, 47, 130, 127, 124};
int i, num;

num = sizeof(data)/sizeof(int);

for( i=0; i<num; i++){
ins_tree(&root, data[i]);
}

print_tree(root);

printf("after del₩n");
root = del_tree(root,32);

print_tree(root);

return 0;
}
```

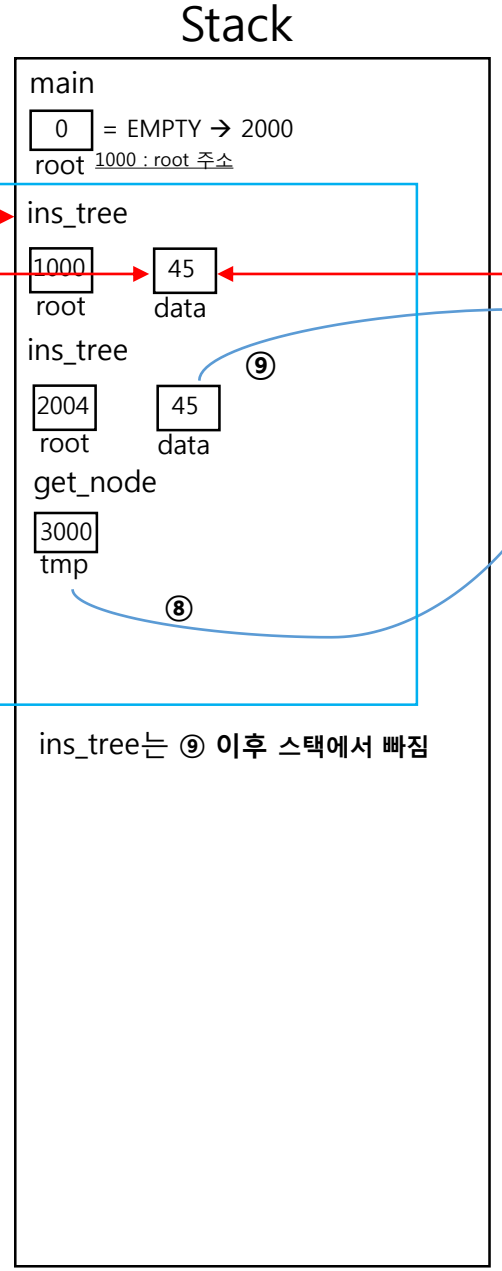


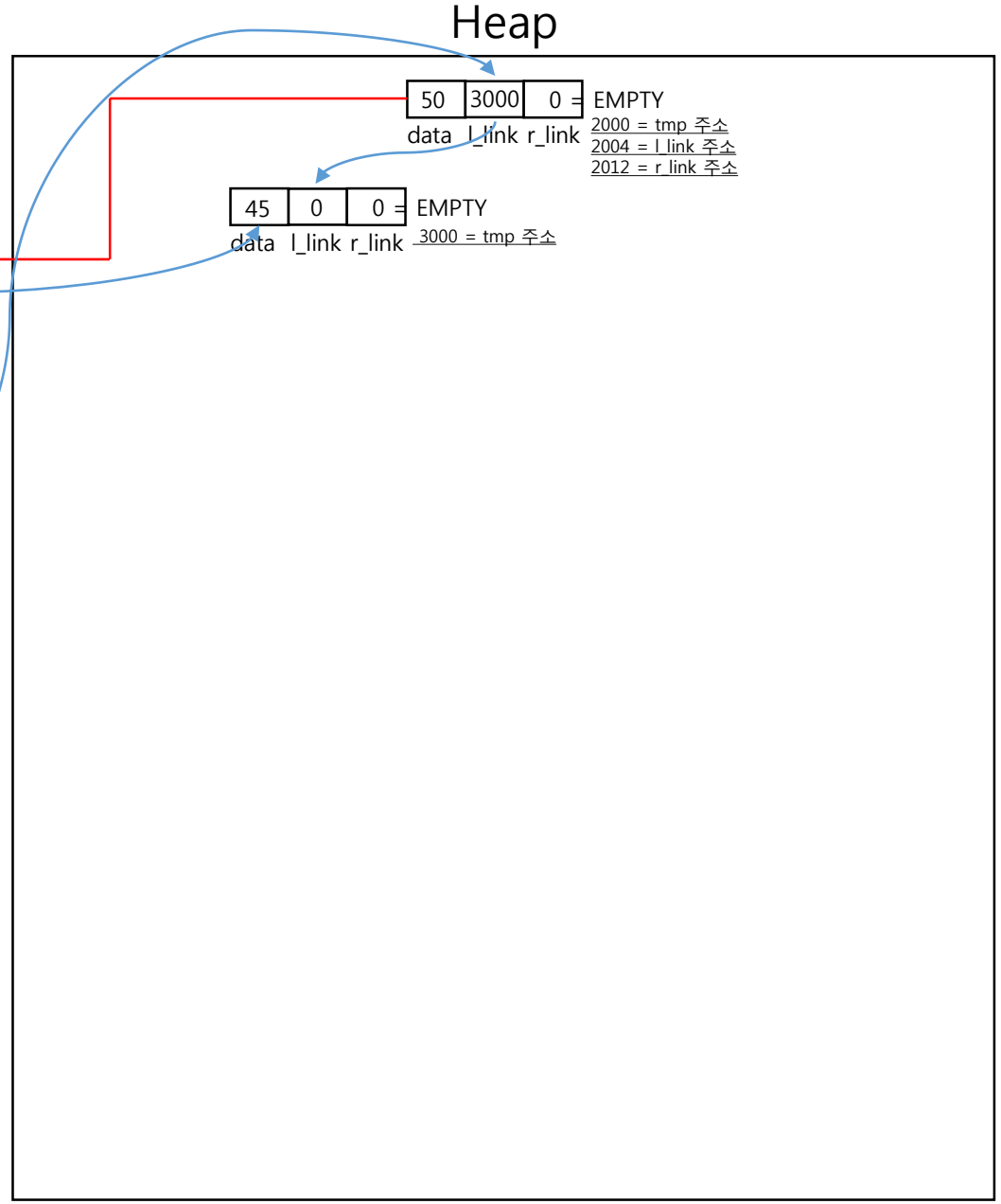


1. 트리(Tree) 코드 이해(그림 그리기) – (ins_tree 3)

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

#define EMPTY 0

struct node{

int data;

struct node *l_link;
struct node *r_link;

};

typedef struct node Tree;

Tree *getnode(){

Tree *tmp = (Tree *)malloc(sizeof(Tree));
tmp->l_link = EMPTY;
tmp->r_link = EMPTY;

return tmp;

}

void ins_tree(Tree **root, int data){

if( *root == EMPTY){

*root = getnode();
(*root)->data = data;
return;
}

⑦
if( (*root)->data < data)
ins_tree( &((*root)->r_link), data);

if( (*root)->data > data)
ins_tree( &((*root)->l_link), data);

}
```

```
void print_tree(Tree *root){

Tree *tmp = root;

if(tmp != NULL){

printf("%d₩n", tmp->data);

/* if(root->l_link)
printf("left = %d, ", root->l_link->data);
else
printf("left = NULL, ");

if(root->r_link)
printf("left = %d, ", root->r_link->data);
else
printf("left = NULL, ");
*/

print_tree(tmp->l_link);
print_tree(tmp->r_link);
}

}
int main(void){

Tree *root = EMPTY;
int data[13] = {50, 45, 73, 32, 48, 46, 16, 37,
120, 47, 130, 127, 124};
int i, num;

num = sizeof(data)/sizeof(int);

for( i=0; i<num; i++){
ins_tree(&root, data[i]); ⑥
}

print_tree(root);

printf("after del₩n");
root = del_tree(root,32);

print_tree(root);

return 0;
}
```





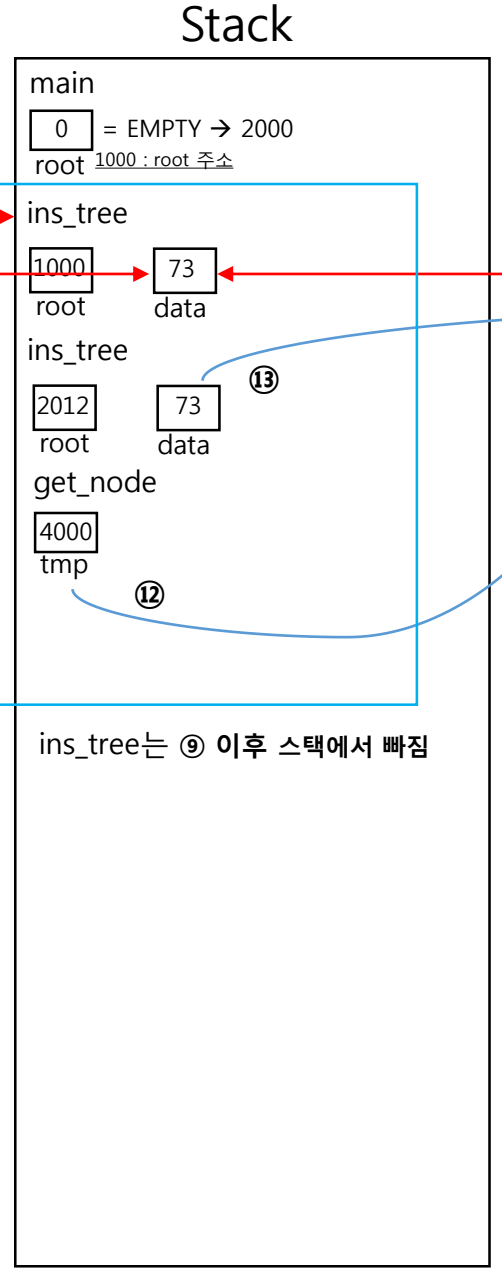
1. 트리(Tree) 코드 이해(그림 그리기) – (ins_tree 4)

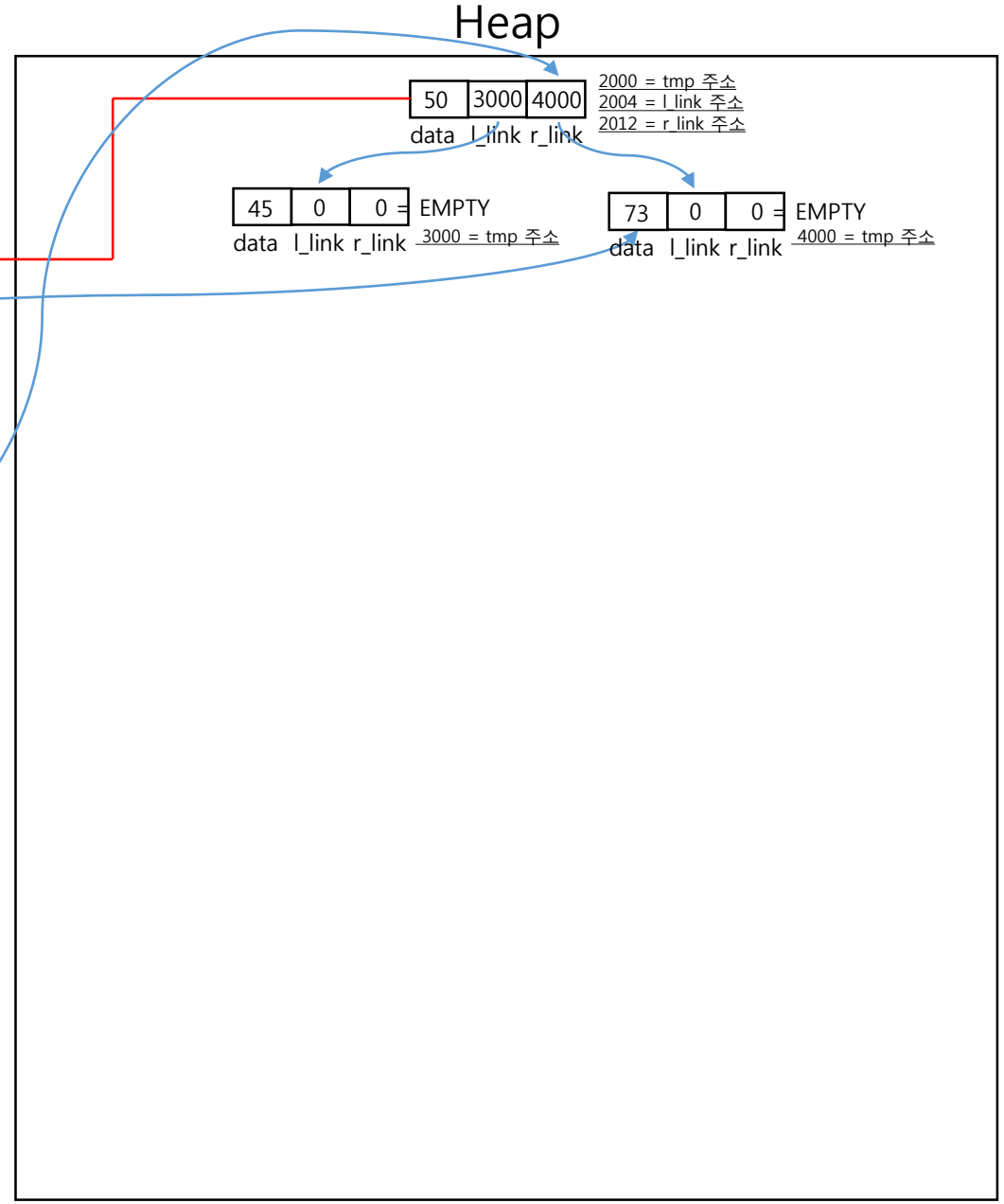
```
#include<stdio.h>
#include<malloc.h>

#define EMPTY 0

struct node{

int data;

struct node *l_link;
struct node *r_link;

};

typedef struct node Tree;

Tree *getnode(){

Tree *tmp = (Tree *)malloc(sizeof(Tree));
tmp->l_link = EMPTY;
tmp->r_link = EMPTY;

return tmp;

}

void ins_tree(Tree **root, int data){

if( *root == EMPTY){

*root = getnode();
(*root)->data = data;
return;
}

if( (*root)->data < data)
ins_tree( &((*root)->r_link), data);

if( (*root)->data > data)
ins_tree( &((*root)->l_link), data);

}
```

⑪

```
void print_tree(Tree *root){

Tree *tmp = root;

if(tmp != NULL){

printf("%d₩n", tmp->data);

/* if(root->l_link)
printf("left = %d, ", root->l_link->data);
else
printf("left = NULL, ");

if(root->r_link)
printf("left = %d, ", root->r_link->data);
else
printf("left = NULL, ");
*/

print_tree(tmp->l_link);
print_tree(tmp->r_link);
}

}
int main(void){

Tree *root = EMPTY;
int data[13] = {50, 45, 73, 32, 48, 46, 16, 37,
120, 47, 130, 127, 124};
int i, num;

num = sizeof(data)/sizeof(int);

for( i=0; i<num; i++){
ins_tree(&root, data[i]);
}

print_tree(root);

printf("after del₩n");
root = del_tree(root,32);

print_tree(root);

return 0;
}
```

⑩





1. 트리(Tree) 코드 이해(그림 그리기) – (ins_tree 5)

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

#define EMPTY 0

struct node{

int data;

struct node *l_link;
struct node *r_link;

};

typedef struct node Tree;

Tree *getnode(){

Tree *tmp = (Tree *)malloc(sizeof(Tree));
tmp->l_link = EMPTY;
tmp->r_link = EMPTY;

return tmp;

}

void ins_tree(Tree **root, int data){

if( *root == EMPTY){

*root = getnode();
(*root)->data = data;
return;
}

if( (*root)->data < data)
ins_tree( &((*root)->r_link), data);

if( (*root)->data > data)
ins_tree( &((*root)->l_link), data);

}
```

```c
void print_tree(Tree *root){

Tree *tmp = root;

if(tmp != NULL){

printf("%d₩n", tmp->data);

/* if(root->l_link)
printf("left = %d, ", root->l_link->data);
else
printf("left = NULL, ");

if(root->r_link)
printf("left = %d, ", root->r_link->data);
else
printf("left = NULL, ");
*/

print_tree(tmp->l_link);
print_tree(tmp->r_link);
}

}
int main(void){

Tree *root = EMPTY;
int data[13] = {50, 45, 73, 32, 48, 46, 16, 37,
120, 47, 130, 127, 124};
int i, num;

num = sizeof(data)/sizeof(int);

for( i=0; i<num; i++){   ⑭
ins_tree(&root, data[i]);
}

print_tree(root);

printf("after del₩n");
root = del_tree(root,32);

print_tree(root);

return 0;
}
```