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

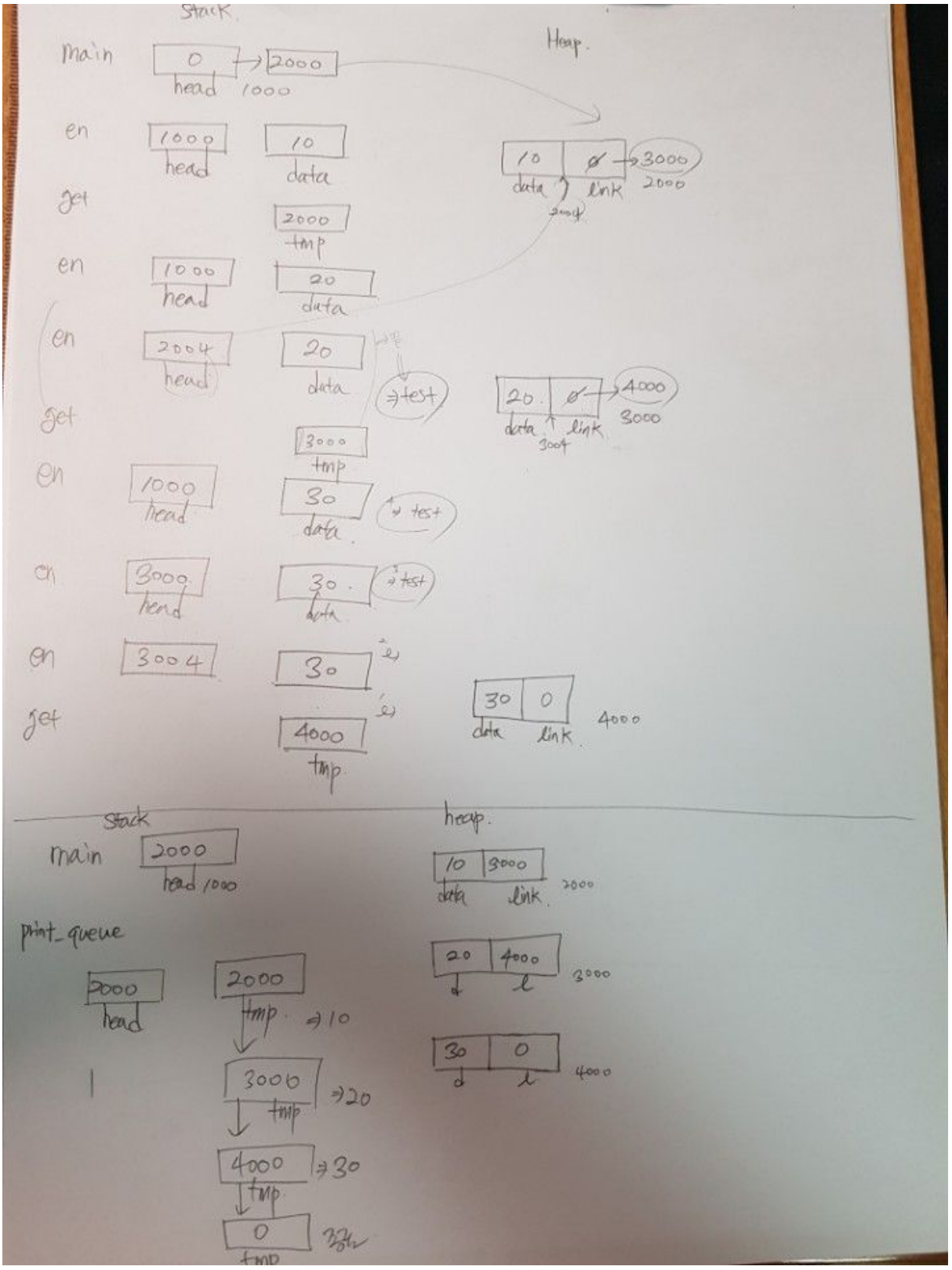
강사 - Innov (이상훈)
gcccompil3r@gmail.com
학생 - 이유성
dbtjd1102@naver.com

9일차 복습

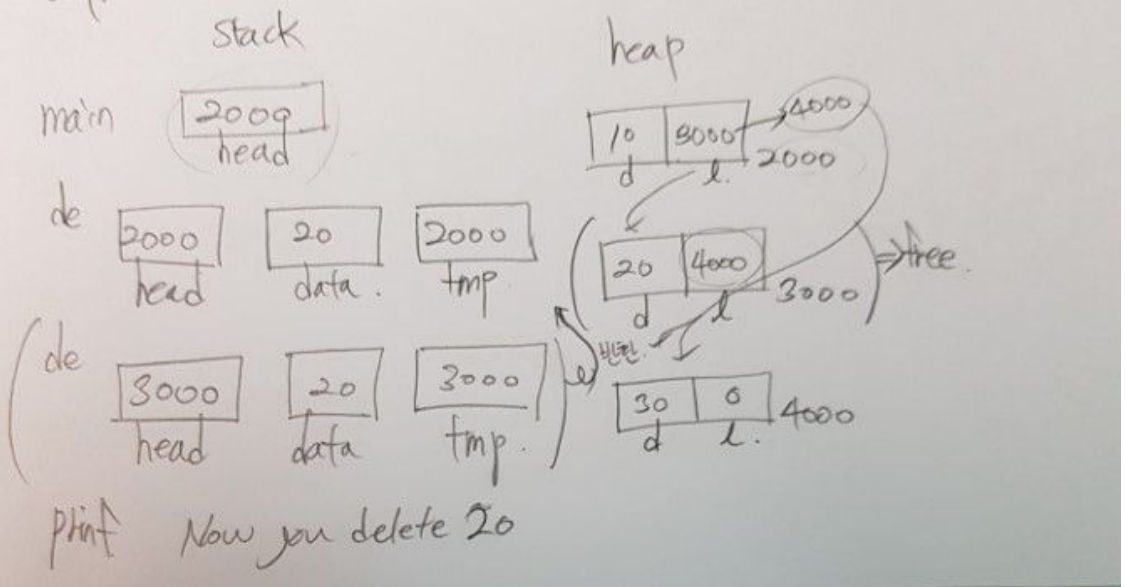
```
#include <stdio.h>
#include <malloc.h>

#define EMPTY 0




struct node
{
	int data;
	struct node *link;

};
typedef struct node Queue;




Queue *get_nude()
{
	Queue *tmp;
	tmp = (Queue *)malloc(sizeof(Queue));
	tmp->link =EMPTY;
	return tmp;

}




void enqueue(Queue **head , int data)
{


	if(*head ==EMPTY)
	{
		*head = get_nude();
		(*head)->data = data;
		return ;
	}
```

```
	enqueue(&(*head)->link,data);

	printf("test\n"); //재귀함수 호출하는 만큼 test
}



void print_queue(Queue *head)
{
	Queue *tmp = head;

	while(tmp)
	{
		printf("%d\n", tmp->data);
		tmp = tmp->link;
	}

}



Queue *dequeue(Queue *head, int data)
{
	Queue *tmp = head;
	if (tmp ==NULL)
		printf("There are no data that you delete\n");
	if(head -> data != data)
		head->link = dequeue (head->link,data);
	else
	{
		//Queue *res = head->link;
		printf("Now you delete %d\n",data);
		free(tmp);
		return head->link;
	}
	return head;
}




int main(void)
{
	Queue *head  =EMPTY ;
	enqueue(&head,10);
	enqueue(&head,20);
	enqueue(&head,30);
```

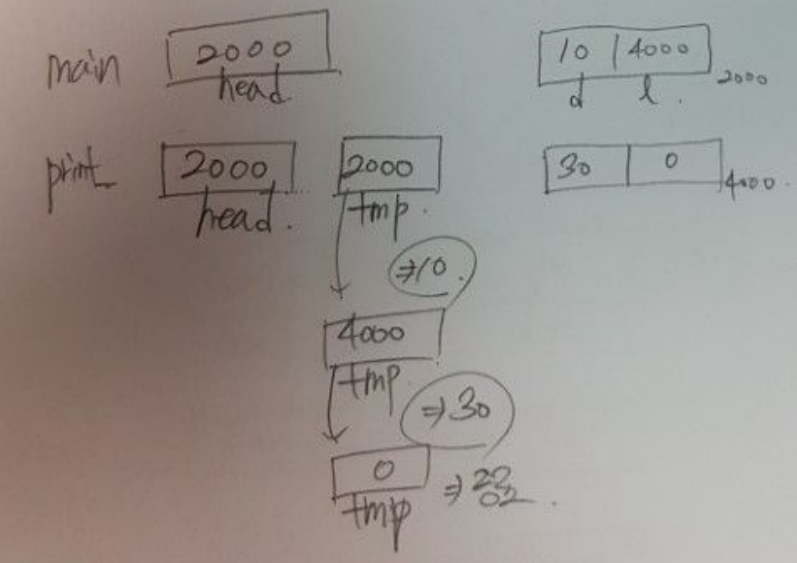
```
        print_queue(head);

        dequeue(head,20);
        print_queue(head);


        return 0;

}
```


Stack
Heap
main
0
2000
head
1000
en
1000
head
10
data
10
data
link
3000
2000
get
2000
tmp
en
1000
head
20
data
en
2004
head
20
data
test
20
data
link
4000
3000
2004
get
3000
tmp
en
1000
head
30
data
test
en
3000
head
30
data
test
en
3004
30
get
4000
tmp
30
0
data
link
4000
stack
heap
main
2000
head
1000
10
3000
data
link
2000
print_queue
2000
head
2000
tmp
10
20
4000
3000
30
0
4000

dequeue

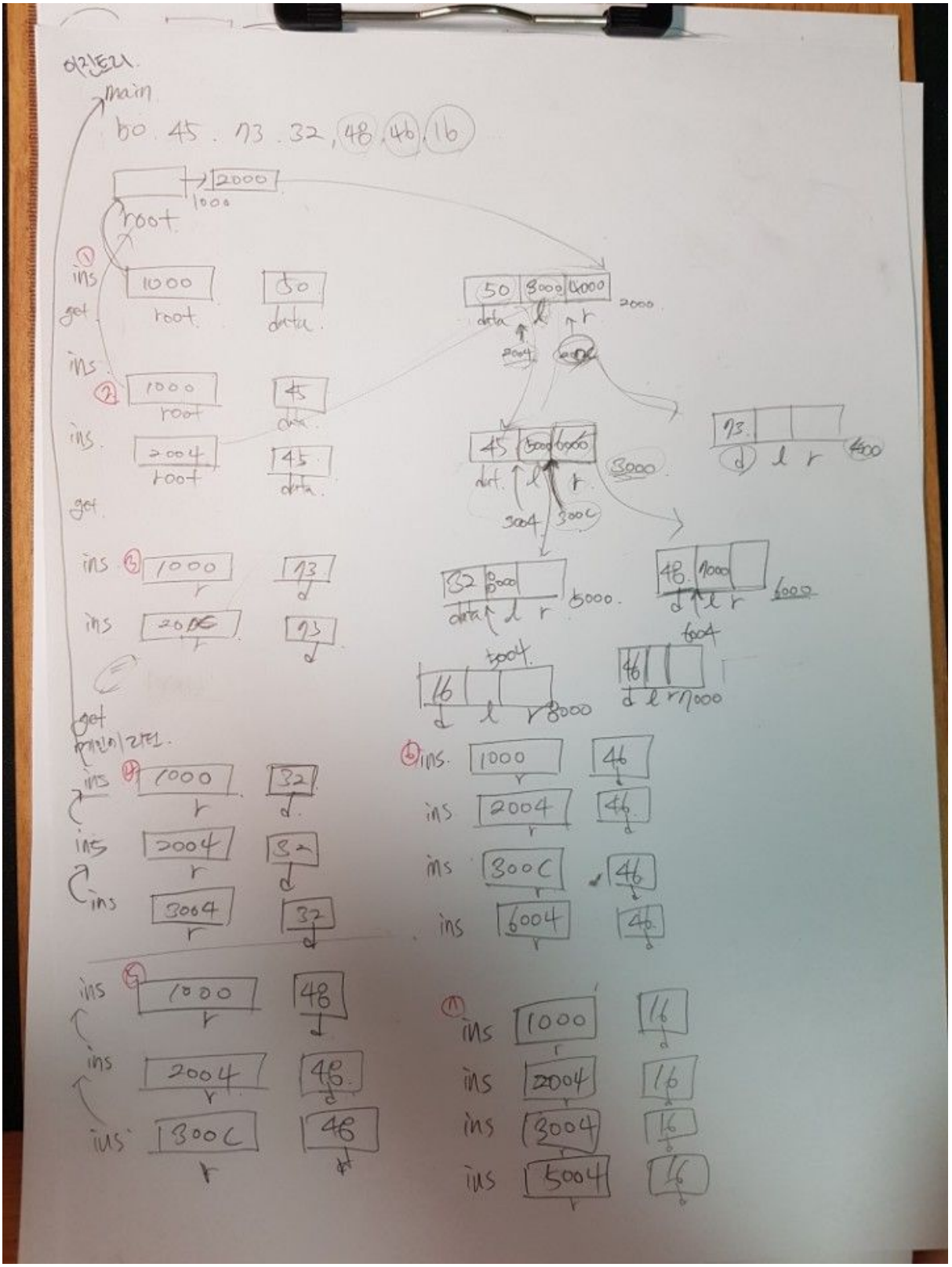


print_queue



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



struct tree{

        int data;
        struct tree *left;
        struct tree *right;
        };



tree *get_node()
{
        tree *tmp;
        tmp = (tree *)malloc(sizeof(tree));
        tmp->left =EMPTY;
        tmp->right =EMPTY;
        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)                                  //여기서 부터 8줄 빼도 상관 없음.
                        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");

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



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;
}
```


main
50 45 73 32 48 46 16
root
1000
2000
ins
get
data
50
45
73
32
48
46
16
3000
4000
5000
6000
3004
300C
5004

print_tree

2000 root	50, 45, 73
3000 root	45, 32, 48
5000 root	32, 16
8000 root	16
6000 root	48, 46
7000 root	46
4000 root	73