TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – 문한나
mhn97@naver.com

복습 - 포인터

포인터의 주소를 받기 위해 더블 포인터를 쓴다

```
52
53 int main(void){
54
55         Stack *top=NULL;
56
57         push(&top,10);
58         push(&top,20);
59         push(&top,30);
60         push(&top,40);
61         print_stack(top);
62
63         pop(&top,10);
64         pop(&top,20);
65         pop(&top,30);
66         print_stack(top);
67
68         return 0;
69
70 }
```

```
20
21 void push(Stack **top,int data){
22
23         Stack *tmp;
24         tmp = *top;
25         (*top) = get_node();
26         (*top)->data=data;
27         (*top)->link = tmp;
28
29 }
```

변수의 주소값를 받기 위해 포인터를 쓴다

```
52
53 int main(void){
54
55         Stack *top=NULL;
56
57         push(&top,10);
58         push(&top,20);
59         push(&top,30);
60         push(&top,40);
61         print_stack(top);
62
63         pop(&top,10);
64         pop(&top,20);
65         pop(&top,30);
66         print_stack(top);
67
68         return 0;
69
70 }
```

```
43
44 void print_stack(Stack *top){
45         Stack *tmp;
46         tmp = top;
47         while(tmp){
48                 printf("%d\n",tmp->data);
49                 tmp = tmp->link;
50         }
51 }
```

복습 - Stack 구현

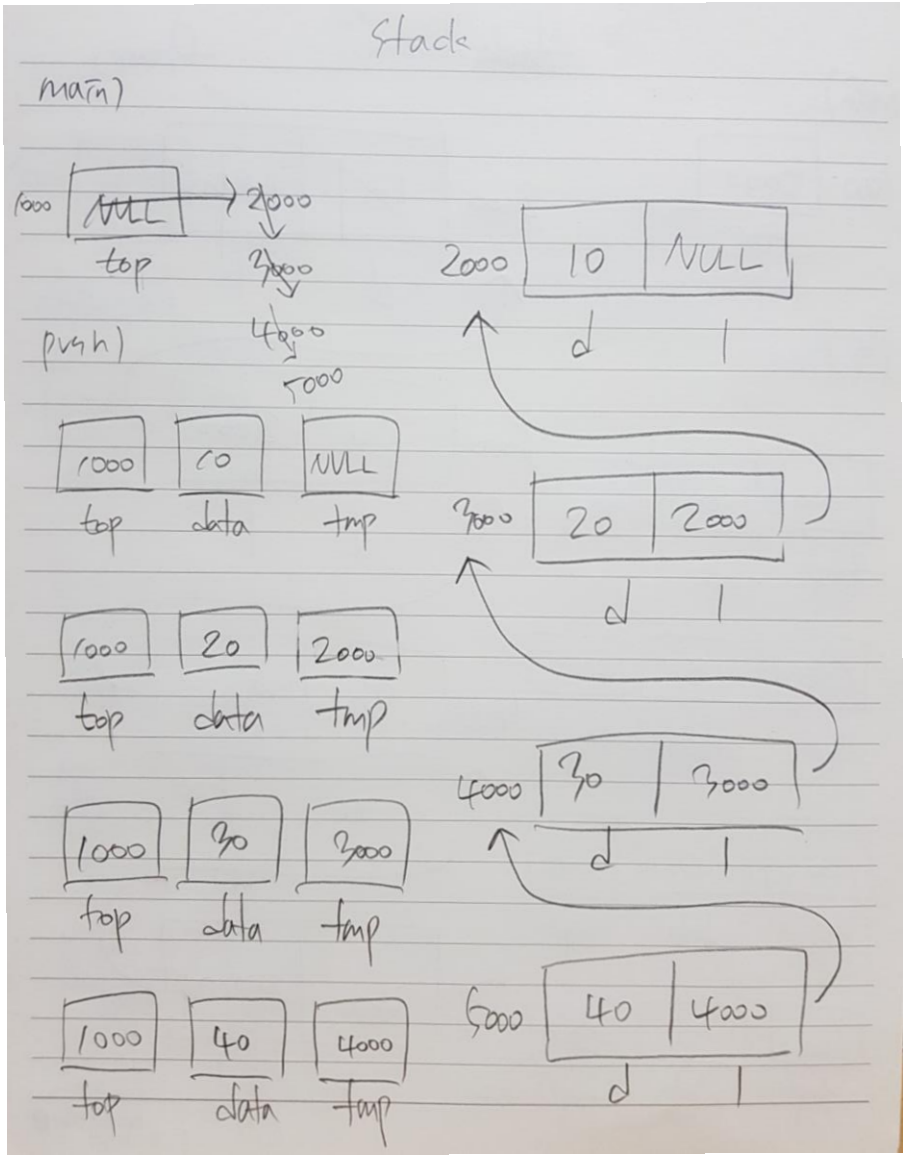
```c
#include <stdio.h>
#include <malloc.h>

struct node{

        int data;
        struct node *link;

};
typedef struct node Stack;

Stack *get_node(){

        Stack *tmp;
        tmp = (Stack *)malloc(sizeof(Stack));
        tmp->link = NULL;
        return tmp;

}

void push(Stack **top,int data){

        Stack *tmp;
        tmp = *top;
        (*top) = get_node();
        (*top)->data=data;
        (*top)->link = tmp;

}

void pop(Stack **top,int data){

        Stack *tmp;
        tmp = *top;
        if(*top == NULL){
                printf("Stack is empty!!!\n");
        }
        if(tmp->data == data){
                free(tmp);
                *top = tmp->link;
        }else pop(&(*top)->link,data);
}

void print_stack(Stack *top){
        Stack *tmp;
        tmp = top;
        while(tmp){
                printf("%d\n",tmp->data);
                tmp = tmp->link;
        }
}
```

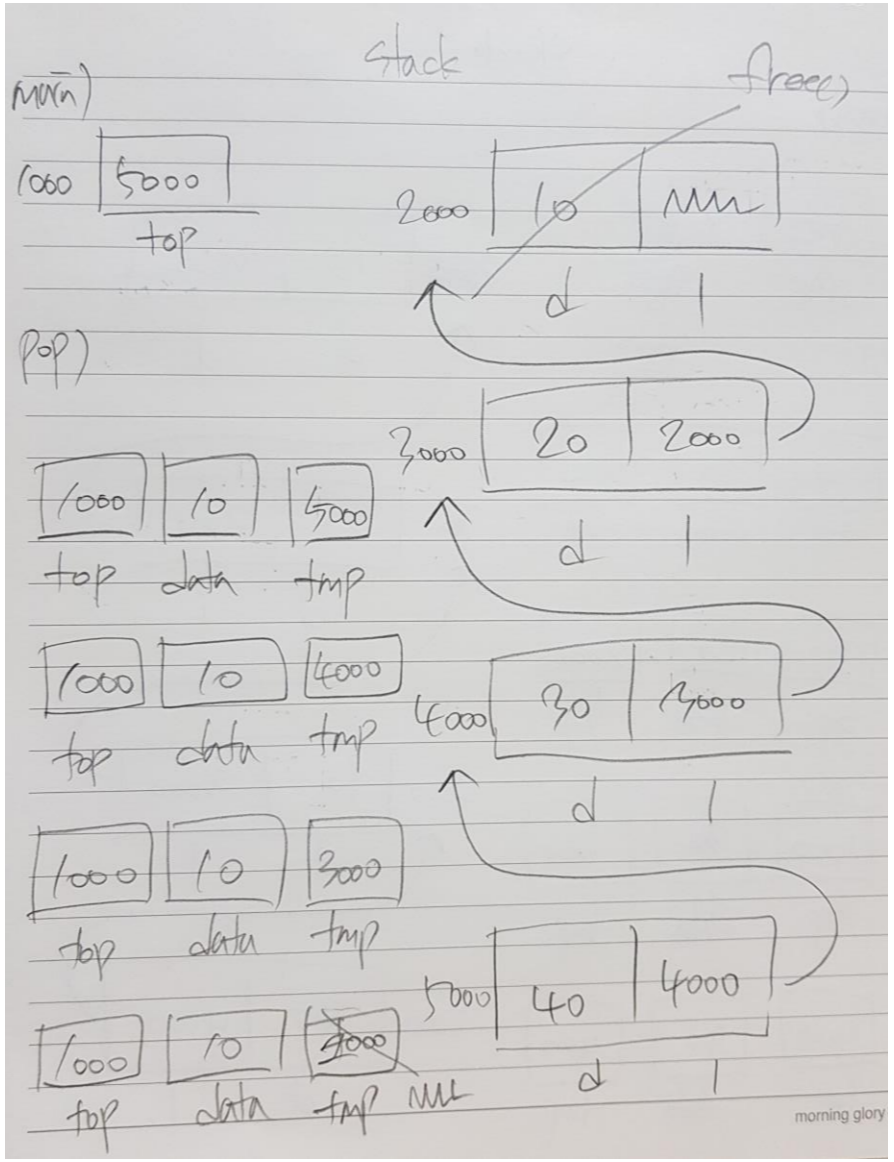
```c

int main(void){

        Stack *top=NULL;

        push(&top,10);
        push(&top,20);
        push(&top,30);
        push(&top,40);
        print_stack(top);

        pop(&top,10);
        pop(&top,20);
        pop(&top,30);
        print_stack(top);

        return 0;

}
```

<결과>

```
mhn@mhn-Z20NH-AS51B5U:~/c/14$ ./a.out
40
30
20
10
40
mhn@mhn-Z20NH-AS51B5U:~/c/14$
```

복습 - Stack 구현


Stack
main)
top
push)
top
data
tmp
pop)


Stack
free()
main)
top
pop)
top
data
tmp
morning glory

복습 - Queue 구현

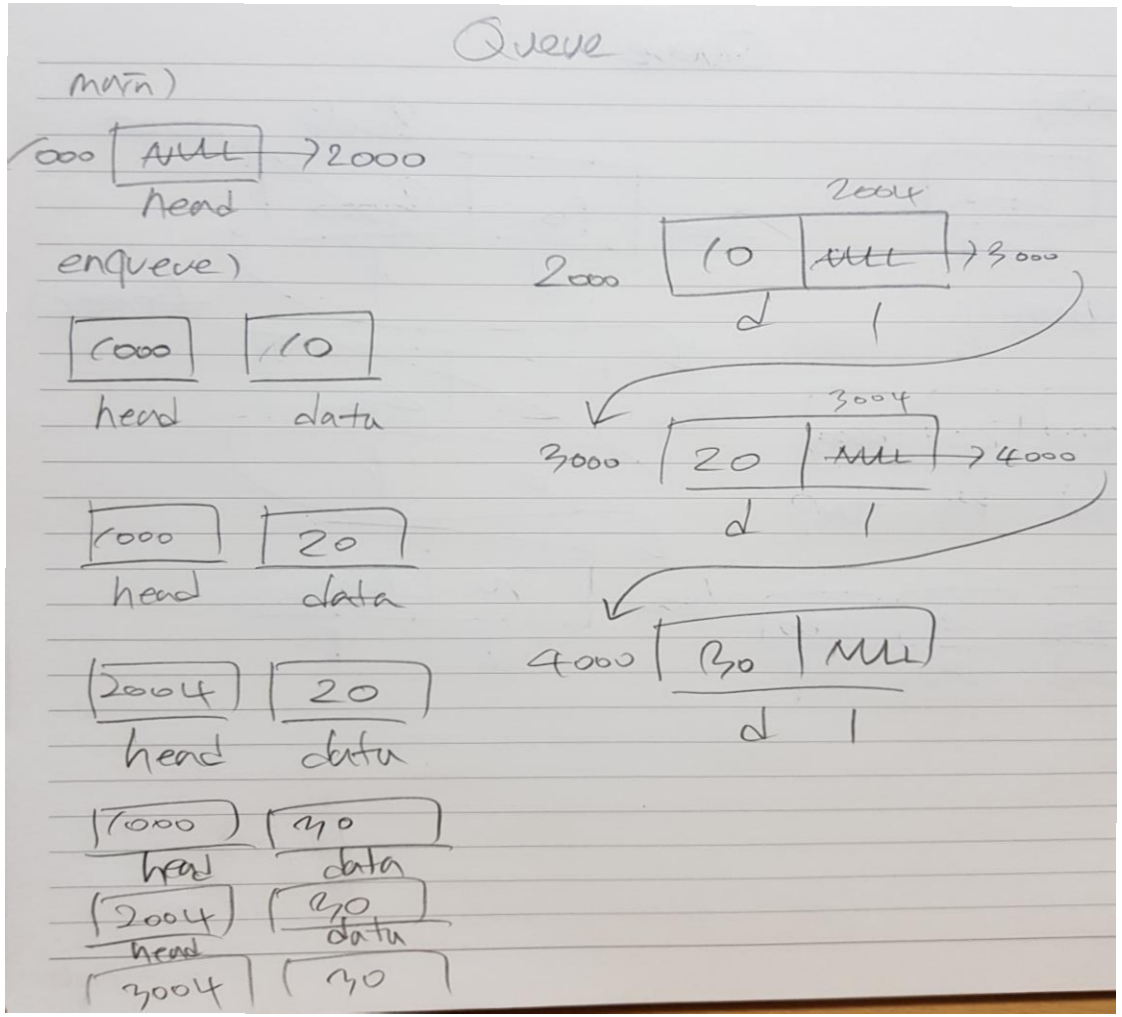
```c
#include <stdio.h>
#include <malloc.h>

struct node{

        int data;
        struct node *link;

};
typedef struct node Queue;

Queue *get_node(){

        Queue *tmp;
        tmp = (Queue *)malloc(sizeof(Queue));
        tmp -> link = NULL;
        return tmp;

}

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

        if(*head == NULL){
        *head = get_node();
        (*head)->data=data;
        return;
        }
        enqueue(&(*head)->link,data);

}

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

        Queue *tmp;
        tmp = *head;
        if((*head) == NULL){
                printf("Queue is Empty!!!");
        }
        if(tmp->data == data){
                free(tmp);
                (*head) = tmp->link;
        }else dequeue(&(tmp->link),data);
}

```

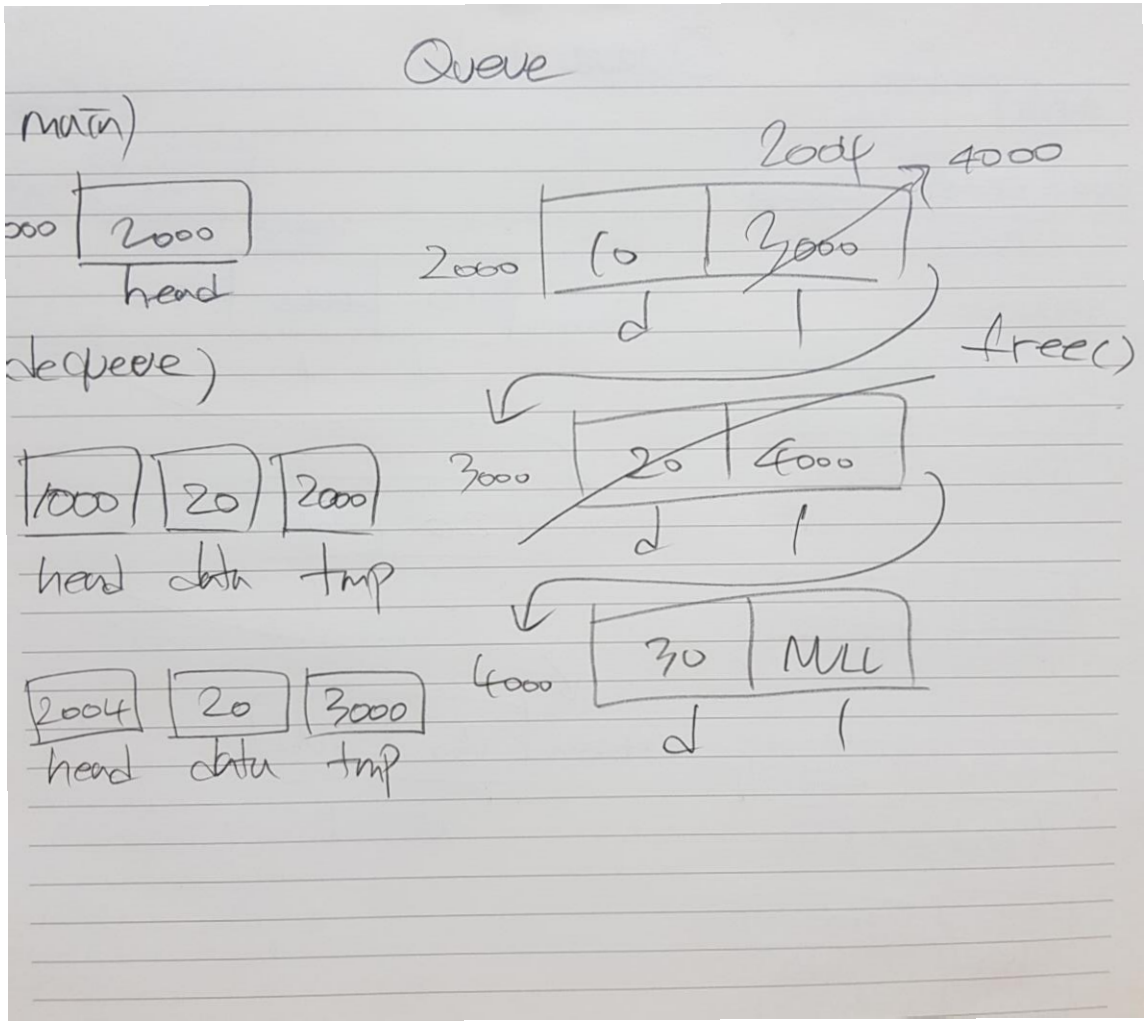
```c

void print_queue(Queue *head){

        Queue *tmp;
        tmp = head;

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

}

int main(void){

        Queue *head = NULL;

        enqueue(&head,10);
        enqueue(&head,20);
        enqueue(&head,30);
        print_queue(head);

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

        return 0;

}
```

<결과>

```
mhn@mhn-Z20NH-AS51B5U:~/c/14$ ./a.out
10
20
30
10
30
mhn@mhn-Z20NH-AS51B5U:~/c/14$
```

복습 - Queue 구현


Queue
main)
NULL
2000
head
enqueue)
1000
10
head
data
20
2004
30
3004
NULL
4000
tail


Queue
main)
2000
head
dequeue)
1000
20
2000
head
data
tmp
2004
3000
10
3000
free()
4000
30
NULL
tail