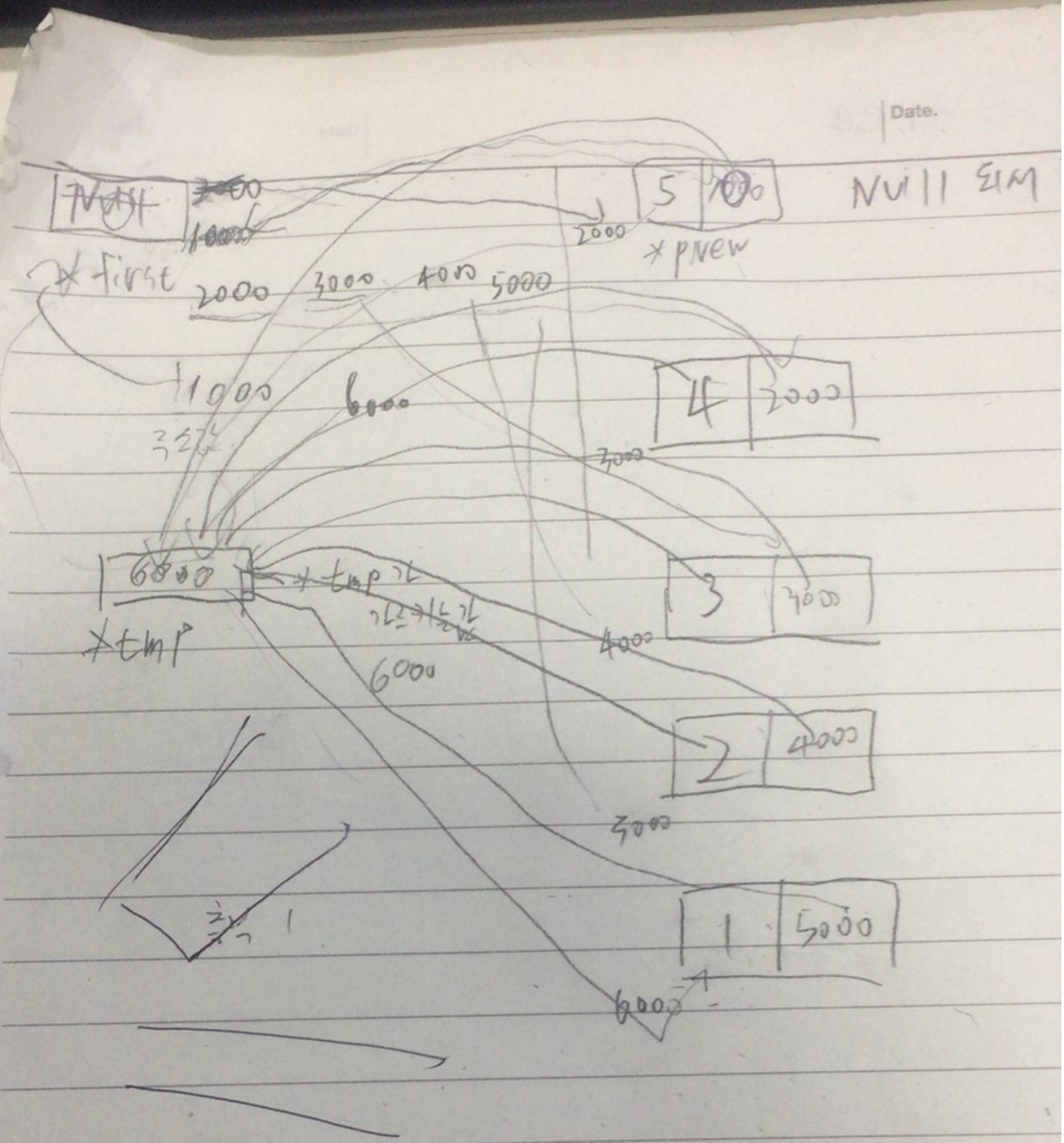
TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – 윤연성
whatmatters@naver.com

```
/*



void non_recur_tree_ins(tree **root, int data)

{
   tree **tmp = root;

   while(*tmp)
   {
      if((*tmp) ->data > data)
         tmp = &(*tmp) ->left;
      else if((*tmp)->data <data)
         tmp = &(tmp) ->right;

   }

   *tmp = get_tree_node();
   (*tmp)->data = data;

}

void non_recur_delete_tree(tree **root, int data)

{
   tree **tmp = root;
   int num;
   while(*tmp)
   {
      if((*tmp) ->data > data)
         tmp = &(*tmp)->left;
      else if((*tmp)->data <data)
         tmp = &(*tmp)->right;
      else if((*tmp)->left &&(*tmp)->right)
      {
         find_max(&(*tmp)->left, &num);
         (*tmp)->data = num;
         return;
      }
      else
      {
            (*tmp) = chg_node(*tmp);
            return;
      }
   }
   printf("Not Found");
}
```

```
int main(void)
{

int num1 = 3, num2 = 7;

int *temp = NULL;
int *num1_p = &num1;
int *num2_p = &num2;
int **num_p_p = &num1_p;
    printf("*num1_p = %d\n", *num1_p);
    printf("*num2_p = %d\n", *num2_p);

temp = num1_p;                  //num1_p    =  *num_p_p 같음 이게 헷갈려서 스택보다가 내려옴
*num_p_p = num2_p;                  //**pp 는   *pp 는 **pp 의 윗단계
num2_p = temp;

    printf("*num1_p = %d\n", *num1_p);
    printf("*num2_p = %d\n", *num2_p);

return 0;
}


typedef struct listNode
{
    int data;
    struct listNode *next;
} listNode;

typedef struct listNode *listPtr;
struct listNode
{
    int data;
    listPtr next;  // listPtr 포인터선언해줬는데 next 는 포인터 안씀
};


                    //void *data;   는 어떤 데이터를 받지 모른다 하면 void *를 써주면됨!
```

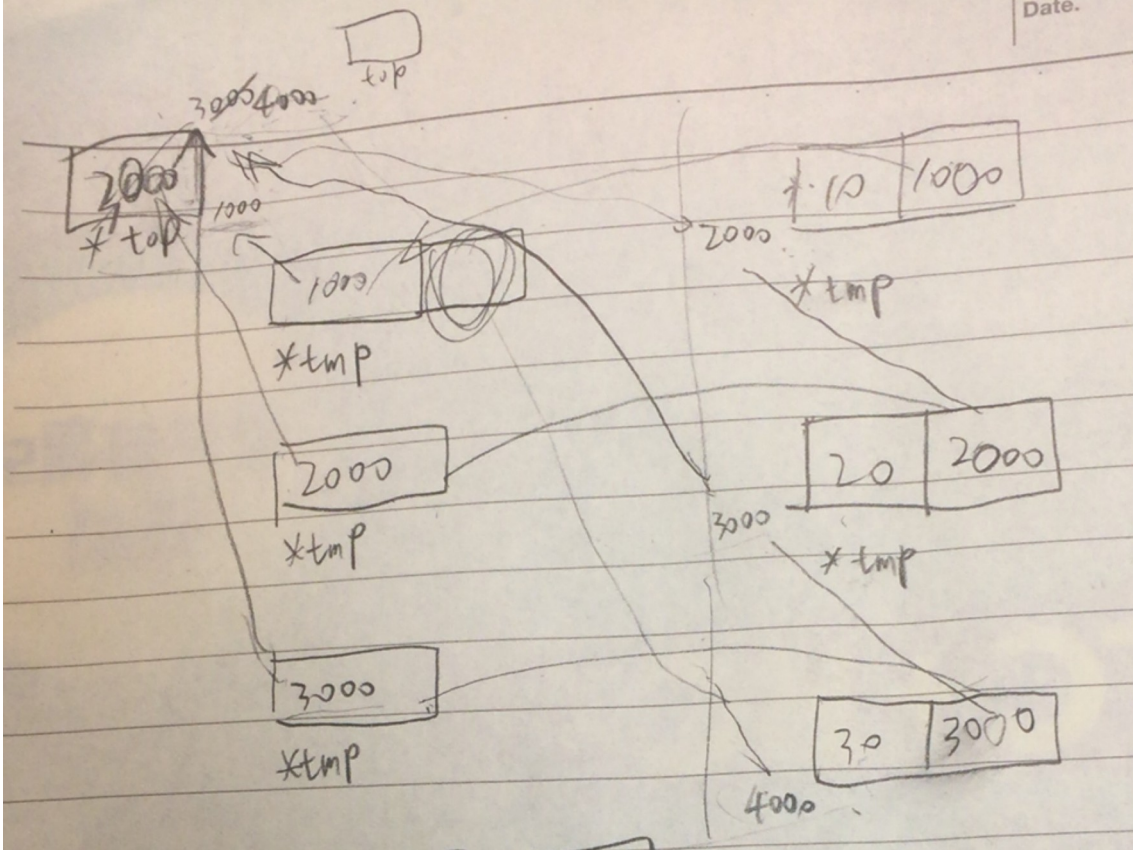
```
#include <stdio.h>
#include <stdlib.h>

typedef struct listNode
{
   int data;
   struct listNode *next;
} listNode;

listNode *first = NULL;                //*first     == NULL

void insert_head(int num)
{
   listNode *pNew = (listNode *)malloc(sizeof(listNode));          //pNew 포인터 선언해줘서 밑에는 안해줬음 (헷갈렸던
거임)
   pNew->data = num;
   pNew->next = first;
   first = pNew;                  //pNew 의 주소값을 퍼스트에 넣음
}                     //*pnew 는 heap 영역 first 는 스택 영역


void remove_head()
{
   listNode *tmp = first;
   if(first)
   {
      first = first->next;
      if (!first)
         last = NULL;
      free(tmp);
   }
}

void printList()
{
   listNode *tmp = first;                // first 도 포인터임 *tmp 라는 변수에 넣은거임 틀은 listNode
   while(tmp)                                  // 0 이 될때까지 게속 돌림
   {
      printf("%4d\n", tmp->data);            //tmp 가 가르키는 주소값을 찾아가서 구조체안의 data 값을 출력
      tmp = tmp->next;                  //tmp 가 가르키는 주소값을 찾아가서 구조체의 next 를 tmp 에 넣음
   }
   printf("\n");
}

int main()
{
   int num;
   int i;
   for (i = 0; i < 5; i++)
   {
      scanf("%d", &num);                 //for 문이라 5 번 num 을 입력받음
      insert_head(num);                  // 5 번 입력받은 num 을 계속 삽입함
      remove_head();

   }
   printList();                  //printfList 로    출력을 계속함
   return 0;
```

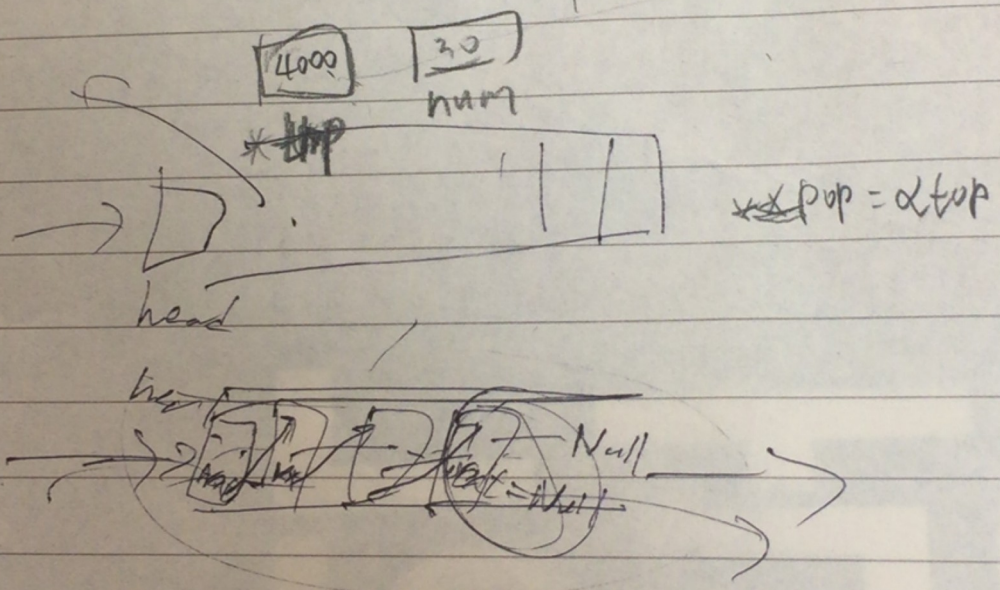

NULL
2000
*first
2000
3000
4000
5000
5
*pNew
NULL
4
3
2
1
6000
*tmp
*tmp
6000

```
#include <stdio.h>
#include <malloc.h>
#define EMPTY 0
struct node
{
   int data;
   struct node *link;
};
typedef struct node Stack;

Stack *get_node()
{
   Stack *tmp;
   tmp = (Stack *)malloc(sizeof(Stack));
   tmp->link = EMPTY;
   return tmp;
}
void push(Stack **top, int data)
{
   Stack *tmp;            //*tmp 라는 Stack 형식의 변수선언
   tmp = *top;            //   *top 값을 tmp 에 넣기
   *top = get_node();
   (*top)->data = data;
   (*top)->link = tmp;
}
int pop(Stack **top)
{
   Stack *tmp;
   int num;
   tmp = *top;
   if (*top == EMPTY)
   {
```

```
        printf("Stack is empty!!!\n");
        return 0;
    }
    num = tmp->data;
    *top = (*top)->link;
    free(tmp);
    return num;
}
int main(void)
{
    Stack *top = EMPTY;                 //typedef struct node Stack; node 를 Stack 로 새로운 정의 = 0; 비어있음
    push(&top, 10);                 //top 의 주소값을 푸시에 인자로 데이터에는 10 을
    push(&top, 20);
    push(&top, 30);
    printf("%d\n", pop(&top));
    printf("%d\n", pop(&top));
    printf("%d\n", pop(&top));
    printf("%d\n", pop(&top));
    return 0;
}
```

Date.


top
2000
*top
1000
1000
*tmp
2000
*tmp
3000
*tmp
*10
1000
2000
*tmp
20
2000
3000
*tmp
30
3000
4000


4000
*tmp
30
num
*pop = &top
head
Null