2018. 3. 5 월 – 8 회차

과정 : TI, DSP, Xilinx Znq FPGA, MCU 기반의 프로그래밍 전문가 과정

Prof. 이상훈
Stu. 정상용

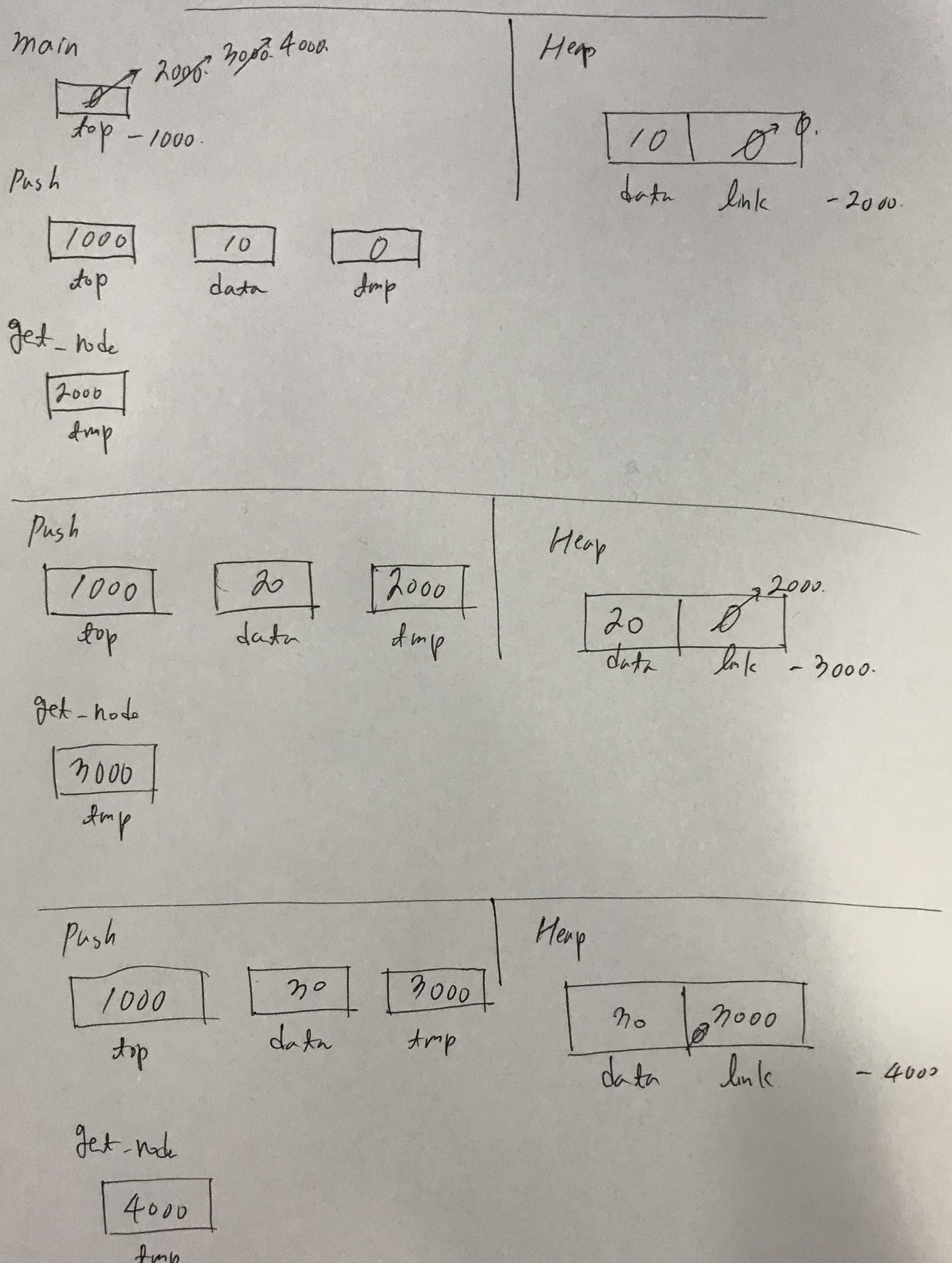
자료구조_1

Ex> Push & pop

```
#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 = *top;
   *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);
```

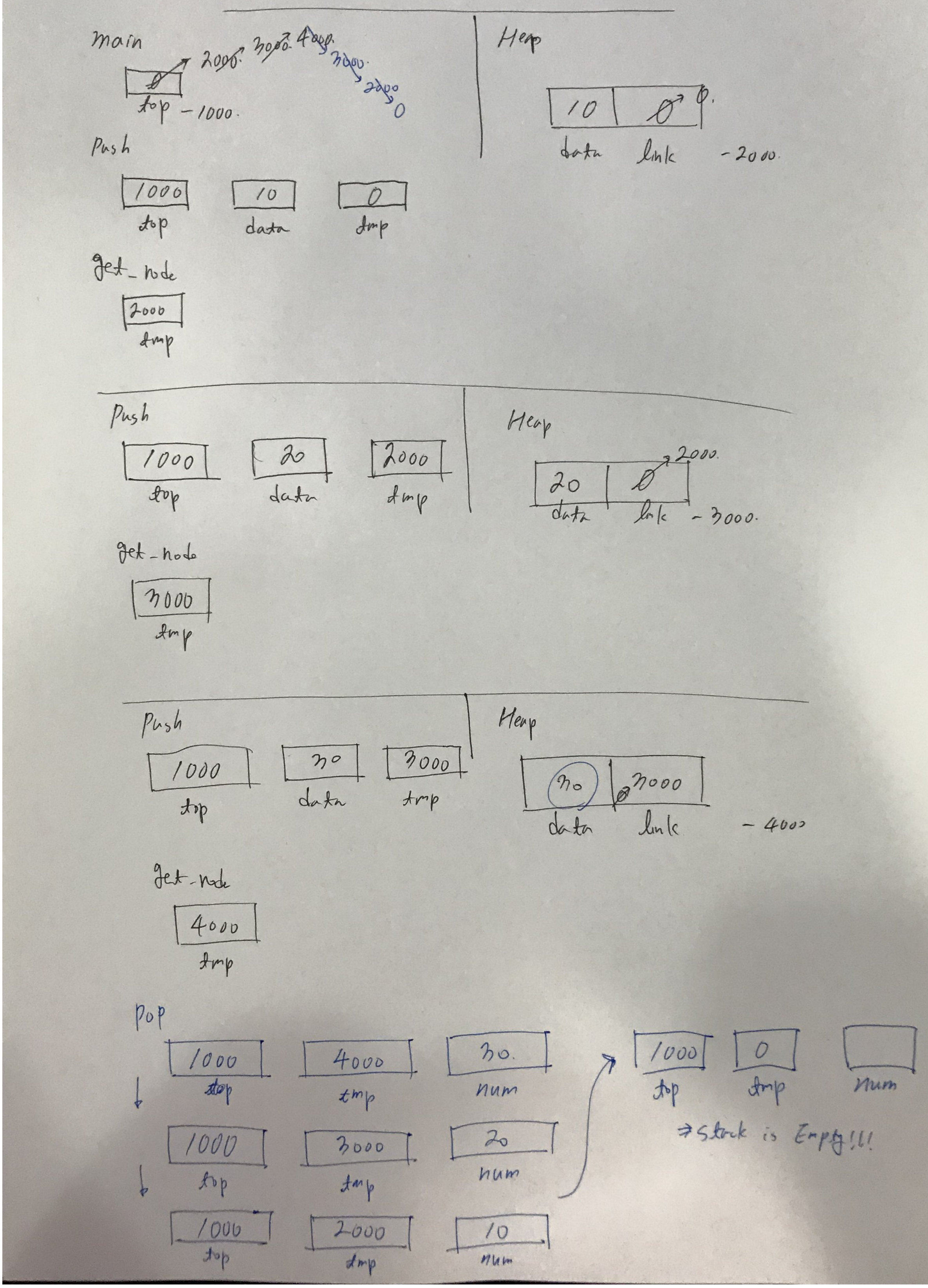
```
    return num;
}


int main(void)
{
    Stack *top = EMPTY;
    push(&top, 10);
    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;
}
```

Push & Pop.



main
0
2000
3000
4000
top
- 1000
Heap
10
0
φ
data
link
- 2000
Push
1000
top
10
data
0
tmp
get_node
2000
tmp
Push
1000
top
20
data
2000
tmp
Heap
20
0
2000
data
link
- 3000
get_node
3000
tmp
Push
1000
top
30
data
3000
tmp
Heap
30
0
3000
data
link
- 4000
get_node
4000
tmp

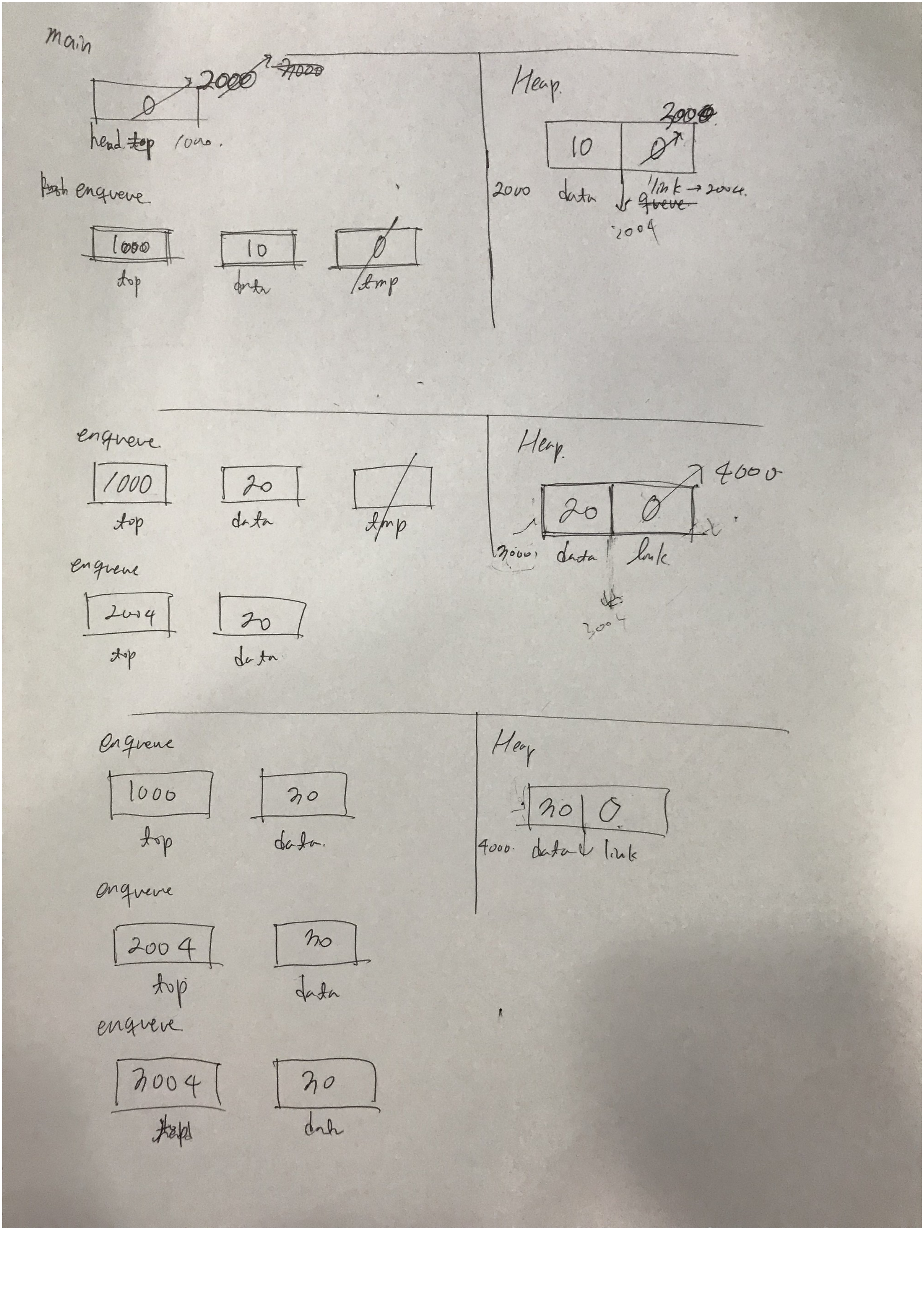
Push & Pop.



⇒ Stack is Empty!!!

Ex> queue

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

#define EMPTY 0

struct __queue
{
    int data;
    struct __queue *link;
};
typedef struct __queue queue;

queue *get_node()
{
    queue *tmp;
    tmp =(queue *)malloc(sizeof(queue));
    tmp -> link = EMPTY;
    return tmp;
}

void enqueue(queue **head, int data)
{
    if(*head == NULL)
    {
         *head = get_node();
         (*head) -> data = data;

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

int print_queue(queue *head)
{
    queue *tmp = head;
    while(tmp)
    {
          printf("%d\n", tmp -> data);
          tmp = tmp -> link;
    }
}




int main(void)
{
    queue *head = EMPTY;
    enqueue(&head, 10);
    enqueue(&head, 20);
    enqueue(&head, 30);
    print_queue(head);
    return 0;
}
```


main
0
2000
~~1000~~
head ~~top~~
1000.
enqueue
1000
10
0
top
data
tmp
Heap.
2000
10
0
2000
data
link → 2004.
~~queue~~
2004
enqueue
1000
20
top
data
tmp
Heap.
4000
20
0
3000
data
link
3004
enqueue
2004
20
top
data
enqueue
1000
30
top
data.
Heap
30
0
4000.
data
link
enqueue
2004
30
top
data
enqueue
3004
30
data