2018. 3. 6 화 – 9 회차

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

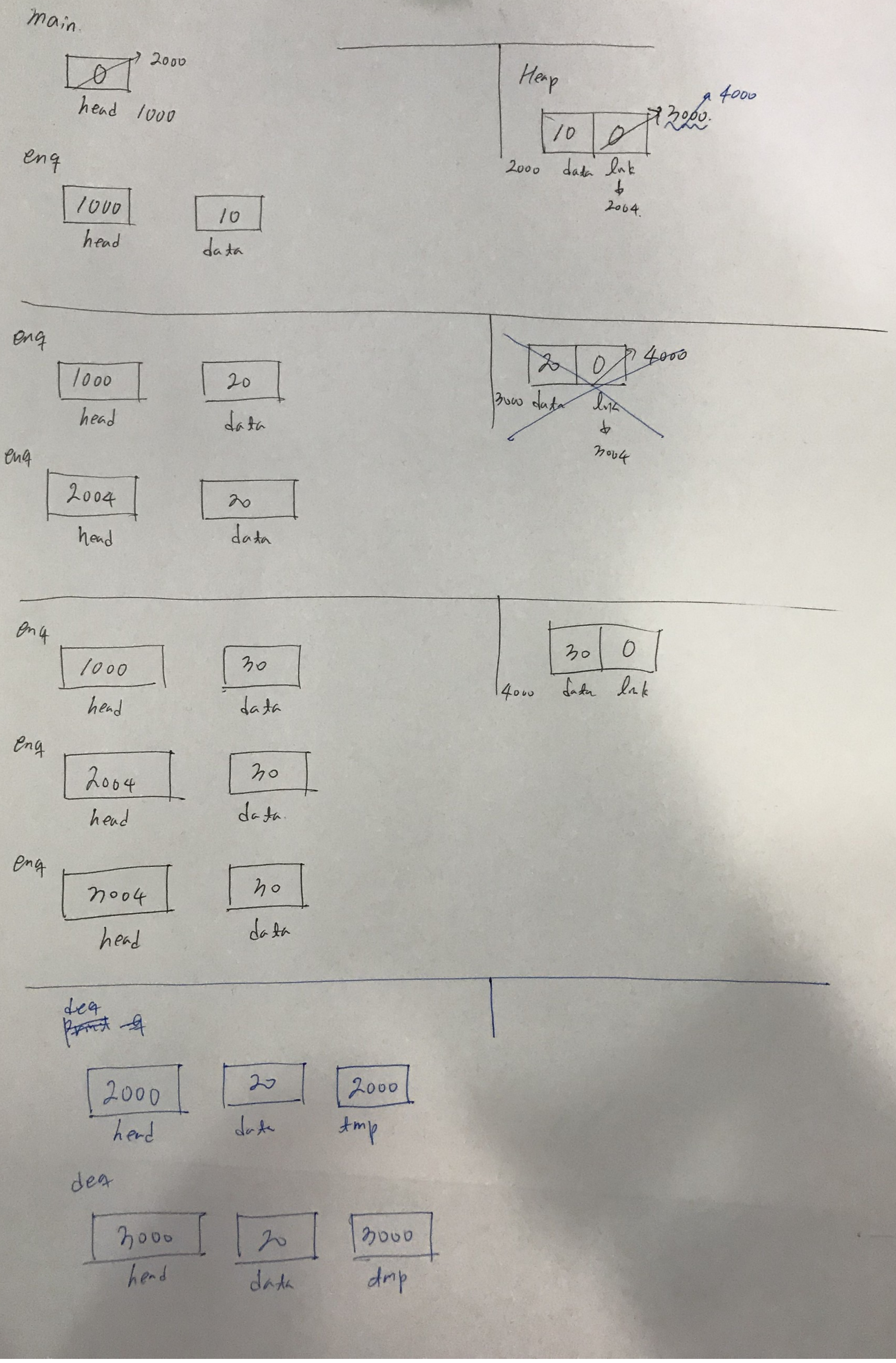
Prof. 이상훈
Stu. 정상용

자료구조_2

1. dequeue

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

#define EMPTY 0

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

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

void enqueue(queue **head, int data)
{
    if(*head == EMPTY)
    {
        *head = get_node();
        (*head) -> data = data;
        return ;
    }
    enqueue(&(*head) -> link, data);
}

int print_queue(queue *head)
{
    queue *tmp;
    tmp = head;
    while(tmp)
    {
         printf("%d\n", tmp -> data);
         tmp = tmp -> link;
    }
    return 0;
}

queue *dequeue(queue *head, int data)
{
    queue *tmp = head;

    if(tmp == NULL)
        printf("There is no data that u delete\n");

    if(head -> data != data)
        head -> link = dequeue(head -> link, data);
    else
    {
        printf("Now u 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);

    head = dequeue(head, 20);

    print_queue(head);

    return 0;
}
```

Sol)



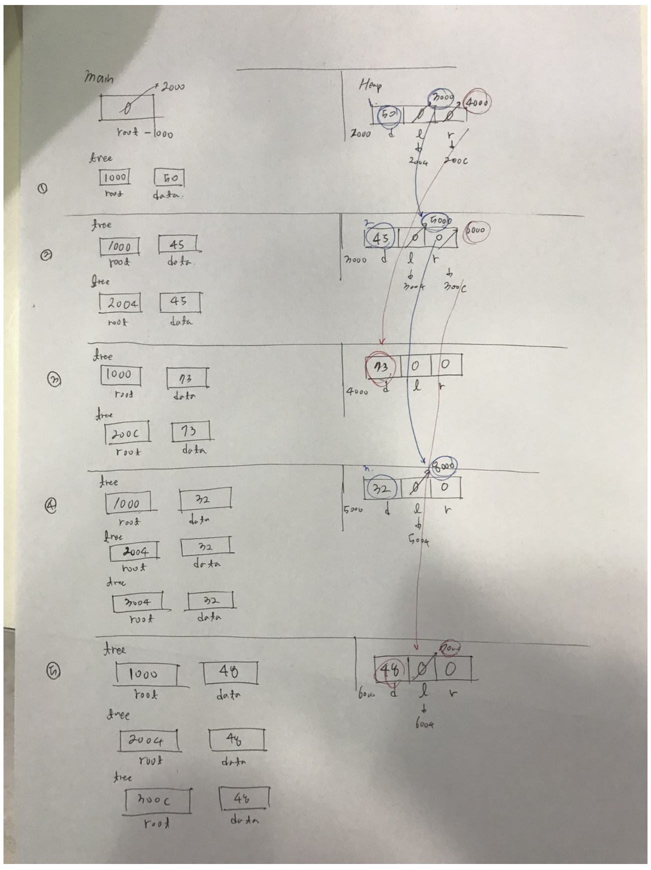
main
0
2000
head 1000
enq
1000
head
10
data
Heap
10
0
2000
4000
2000 data link
2004
enq
1000
head
20
data
20
0
4000
3000 data link
3004
enq
2004
head
20
data
enq
1000
head
30
data
30
0
4000 data link
enq
2004
head
30
data
enq
3004
head
30
data
deq
~~print~~ q
2000
head
20
data
2000
tmp
deq
3000
head
20
data
3000
tmp

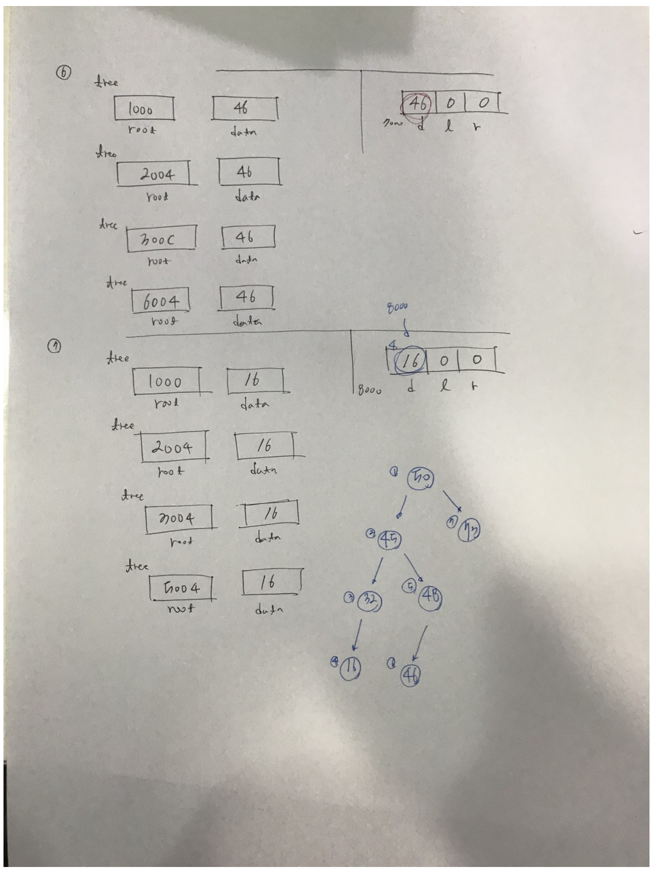
2. tree_ins(data & print)

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

#define EMPTY 0

struct tree
{
   int data;
   struct tree *left;
   struct tree *right;
};
typedef struct tree queue;

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

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

   else if(data > (*head) -> data)
   {
      enqueue(&(*head) -> right, data);
   }
   else if(data < (*head) -> data)
   {
      enqueue(&((*head) -> left), data);
   }
   else if(data == (*head) -> data)
   {
      return;
   }
}

int print_queue(queue *head)
{
  if(head)
  {
     printf("data = %d\n,", head -> data);
     print_queue(head -> left);
```

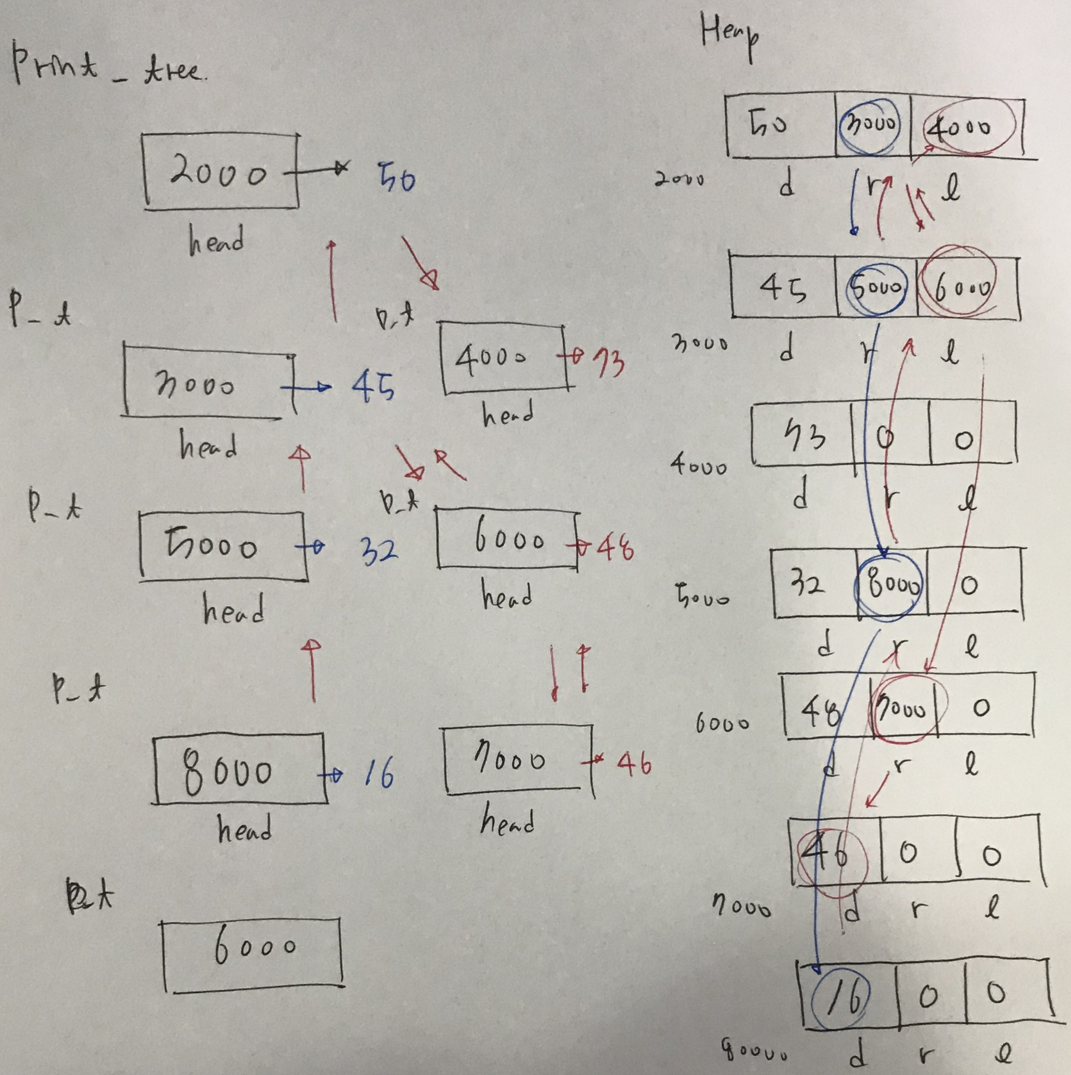
```
        print_queue(head -> right);
    }
}




int main(void)
{
    queue *head = EMPTY;

    int i;
    int arr[13] = {50, 45, 73, 32, 48, 46, 16, 37, 120, 47, 130, 127, 124};
    for(i = 0; i < 13; i++)
    {
        enqueue(&head, arr[i]);
    }

    print_queue(head);

    return 0;
}
```

Sol)





Print_tree.



Heap
2000
head
50
p_t
3000
45
p_t
4000
73
head
p_t
5000
32
p_t
6000
48
head
p_t
8000
16
7000
46
head
p_t
6000
50 | 3000 | 4000
2000 d r l
45 | 5000 | 6000
3000 d r l
73 | 0 | 0
4000 d r l
32 | 8000 | 0
5000 d r l
48 | 7000 | 0
6000 d r l
46 | 0 | 0
7000 d r l
16 | 0 | 0
8000 d r l