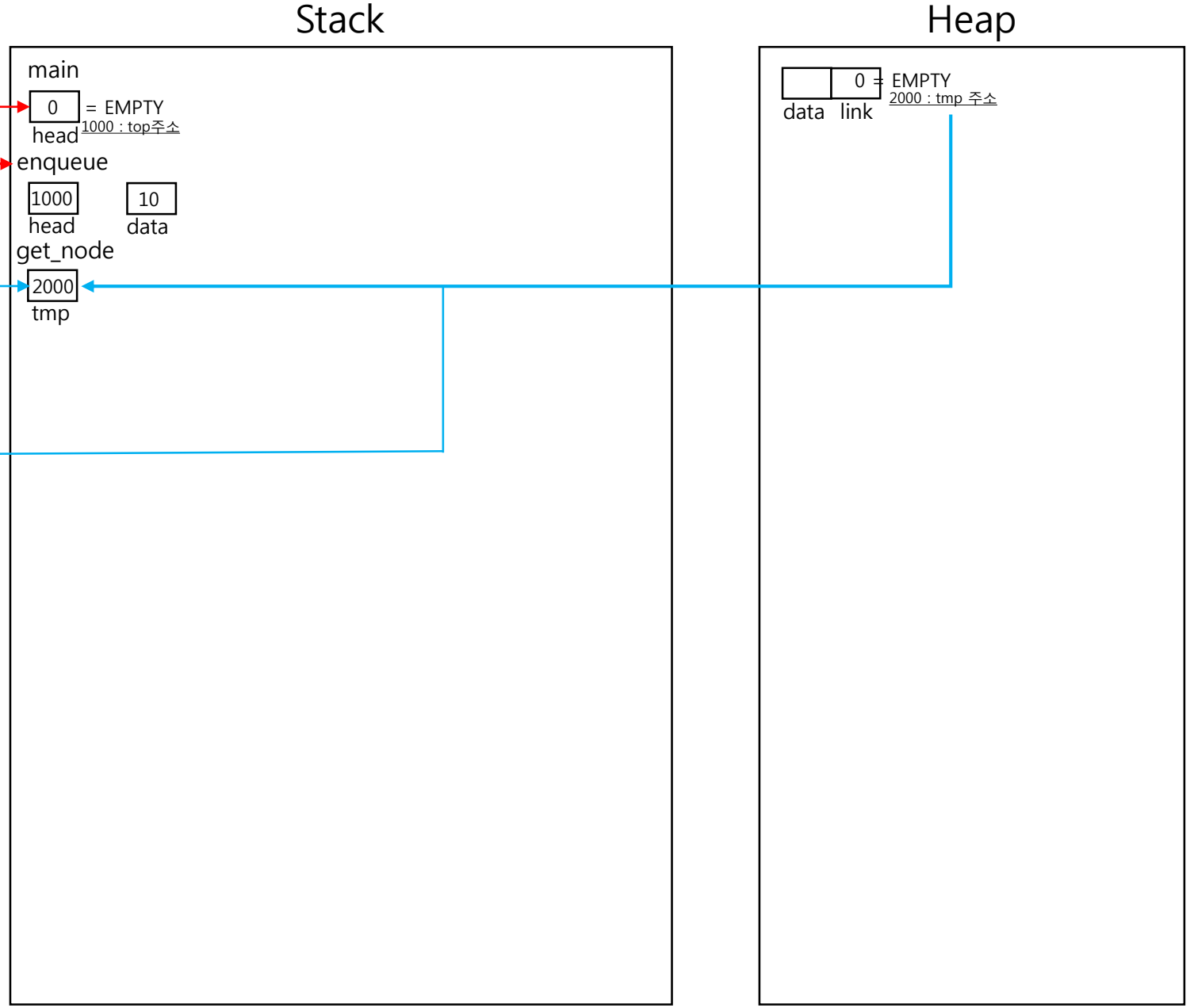
TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – GJ (박현우)
uc820@naver.com

1.큐(Queue) 코드 이해(그림 그리기) – (enqueue 1)

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

#define EMPTY 0

struct node{

int data;
struct node *link;
};

typedef struct node Queue;

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

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

Queue *tmp;
if( *head == EMPTY){

printf("QUEUE IS EMPTY₩n");
}

tmp = *head;
if(tmp->data ==data){

*head = tmp->link;
free(tmp);
}else
dequeue( &(tmp->link), data);
}
```

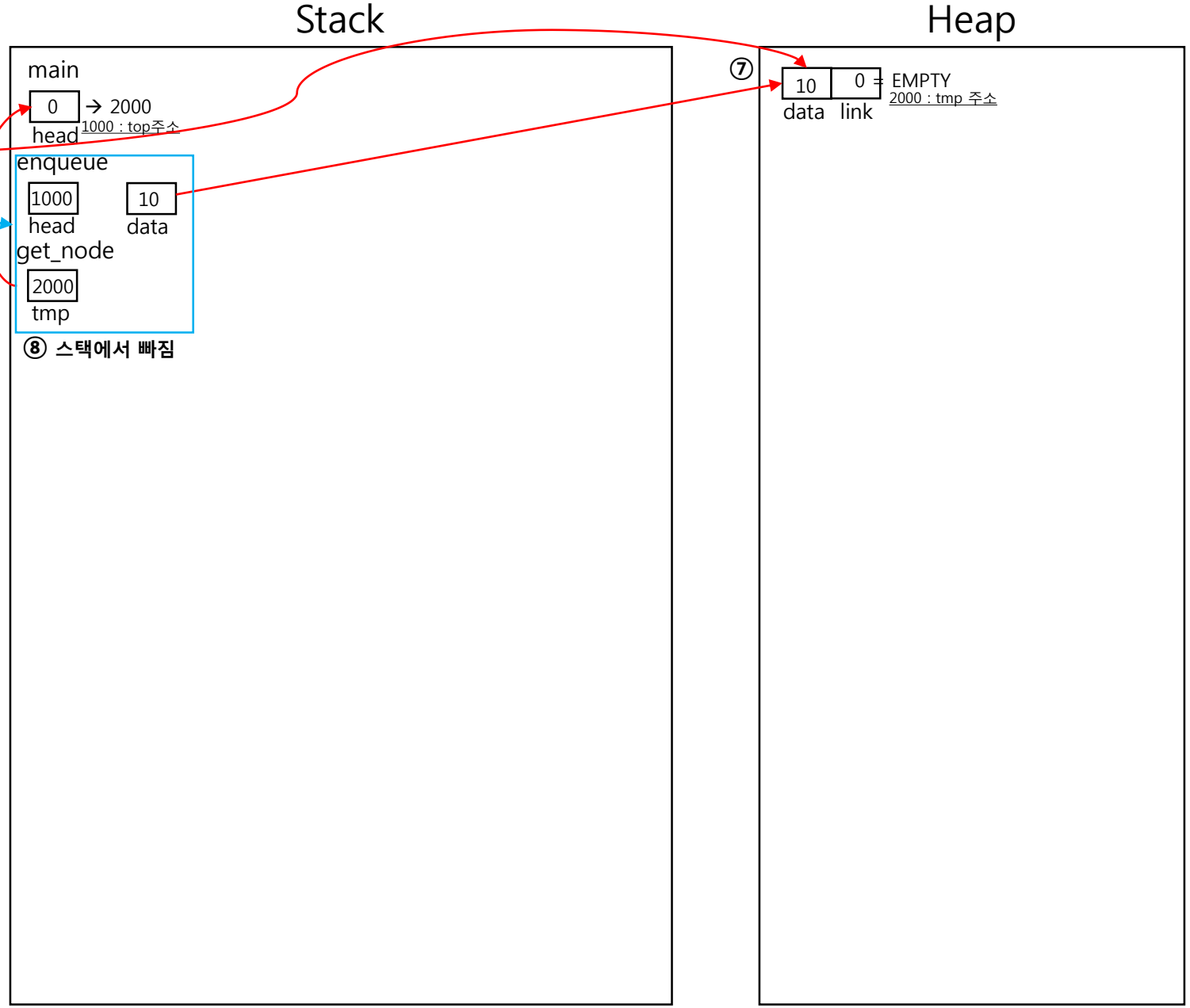
```
void enqueue(Queue **head, int data){
if( *head == EMPTY){

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

void 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);


dequeue(&head,20);


print_queue(&head);

return 0;
}
```



1.큐(Queue) 코드 이해(그림 그리기) – (enqueue 2)

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

#define EMPTY 0

struct node{

int data;
struct node *link;
};

typedef struct node Queue;

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

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

Queue *tmp;
if( *head == EMPTY){

printf("QUEUE IS EMPTY₩n");
}

tmp = *head;
if(tmp->data ==data){

*head = tmp->link;
free(tmp);
}else
dequeue( &(tmp->link), data);
}
```

```
void enqueue(Queue **head, int data){
if( *head == EMPTY){

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

void 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);


dequeue(&head,20);


print_queue(&head);

return 0;
}
```

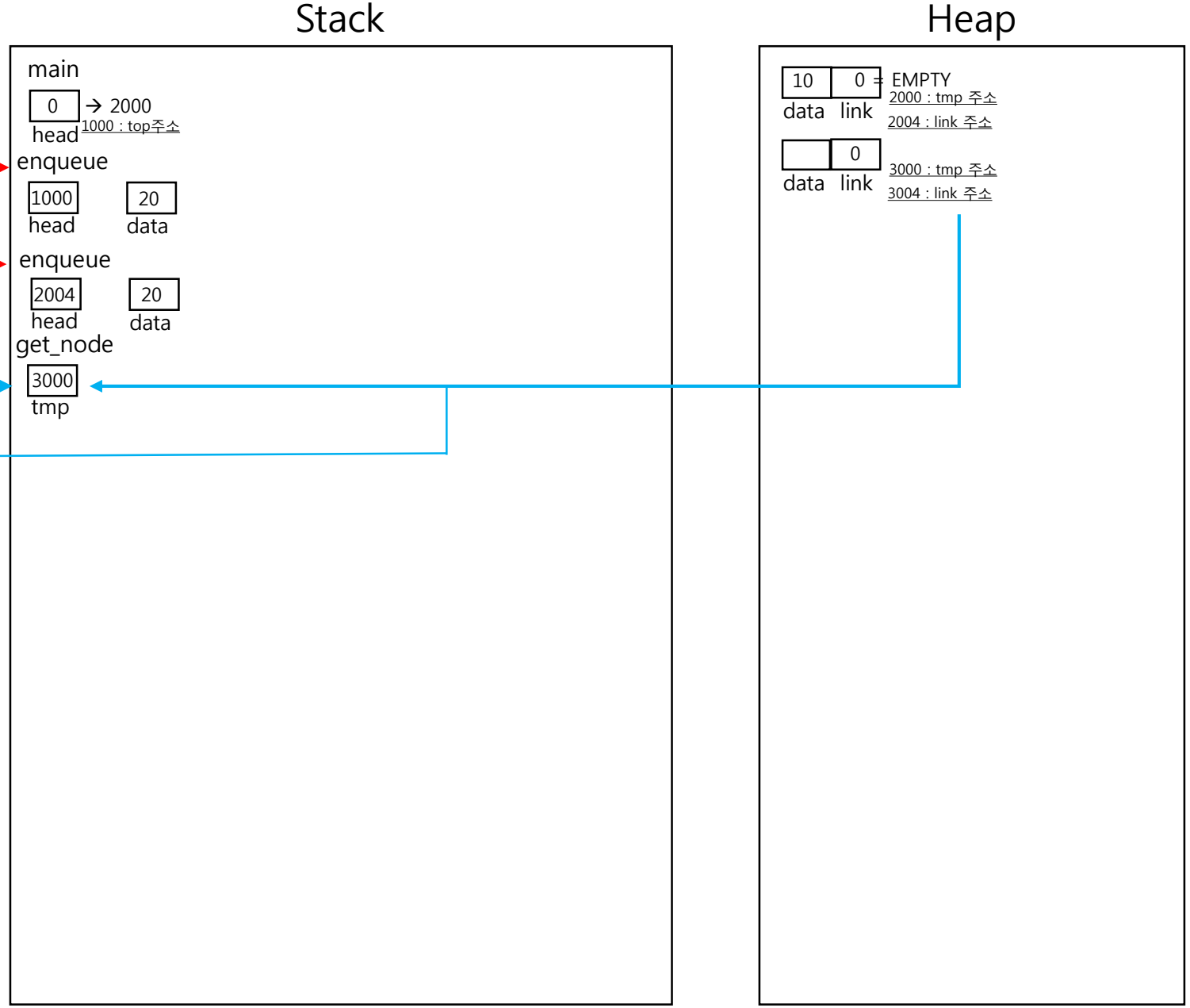


1.큐(Queue) 코드 이해(그림 그리기) – (enqueue 3)

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

#define EMPTY 0

struct node{

int data;
struct node *link;
};

typedef struct node Queue;

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

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

Queue *tmp;
if( *head == EMPTY){

printf("QUEUE IS EMPTY₩n");
}

tmp = *head;
if(tmp->data ==data){

*head = tmp->link;
free(tmp);
}else
dequeue( &(tmp->link), data);
}
```

```
void enqueue(Queue **head, int data){
if( *head == EMPTY){

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

void 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);


dequeue(&head,20);


print_queue(&head);

return 0;
}
```

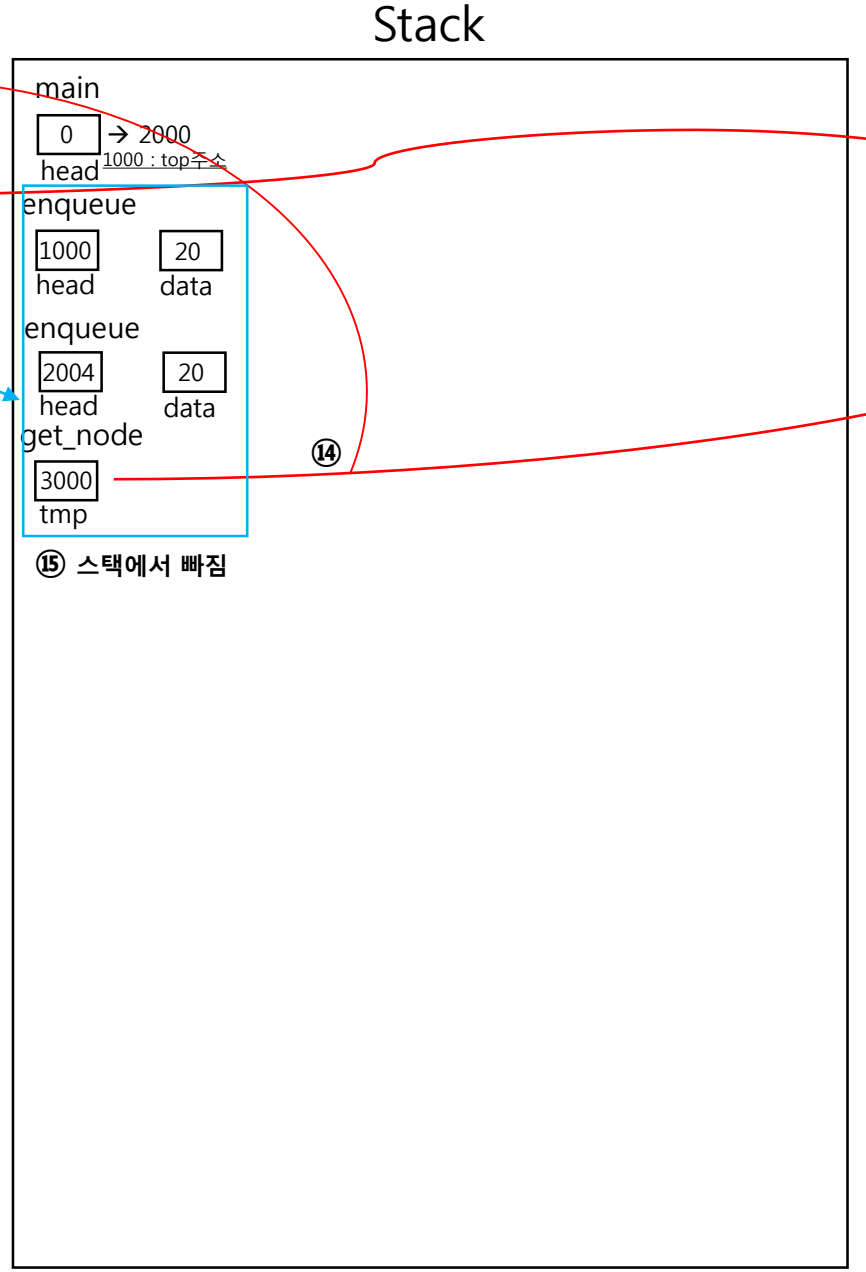
⑨ ⑩ ⑪ ⑫ ⑬

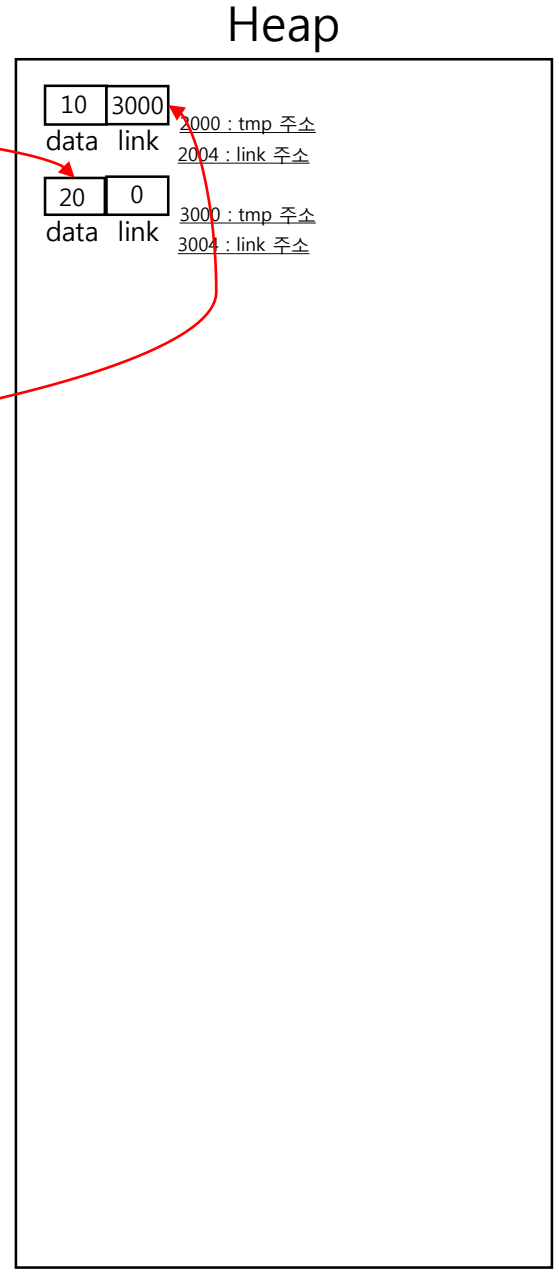


1.큐(Queue) 코드 이해(그림 그리기) – (enqueue 4)

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

#define EMPTY 0

struct node{

int data;
struct node *link;
};

typedef struct node Queue;

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

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

Queue *tmp;
if( *head == EMPTY){

printf("QUEUE IS EMPTY₩n");
}

tmp = *head;
if(tmp->data ==data){

*head = tmp->link;
free(tmp);
}else
dequeue( &(tmp->link), data);
}
```

```
void enqueue(Queue **head, int data){
if( *head == EMPTY){

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

void 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);


dequeue(&head,20);


print_queue(&head);

return 0;
}
```

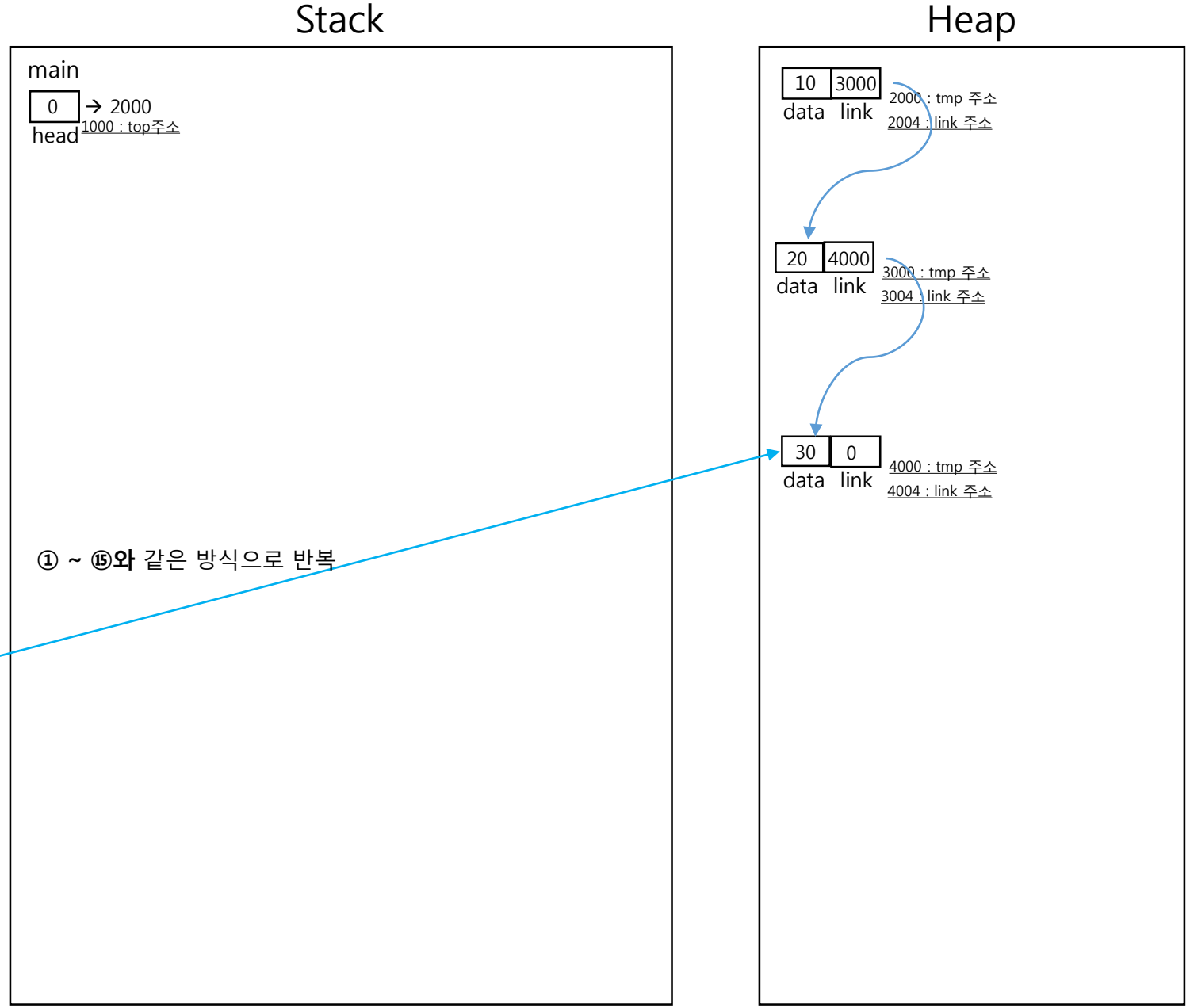




1.큐(Queue) 코드 이해(그림 그리기) – (enqueue 5)

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

#define EMPTY 0

struct node{

int data;
struct node *link;
};

typedef struct node Queue;

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

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

Queue *tmp;
if( *head == EMPTY){

printf("QUEUE IS EMPTY₩n");
}

tmp = *head;
if(tmp->data ==data){

*head = tmp->link;
free(tmp);
}else
dequeue( &(tmp->link), data);
}
```

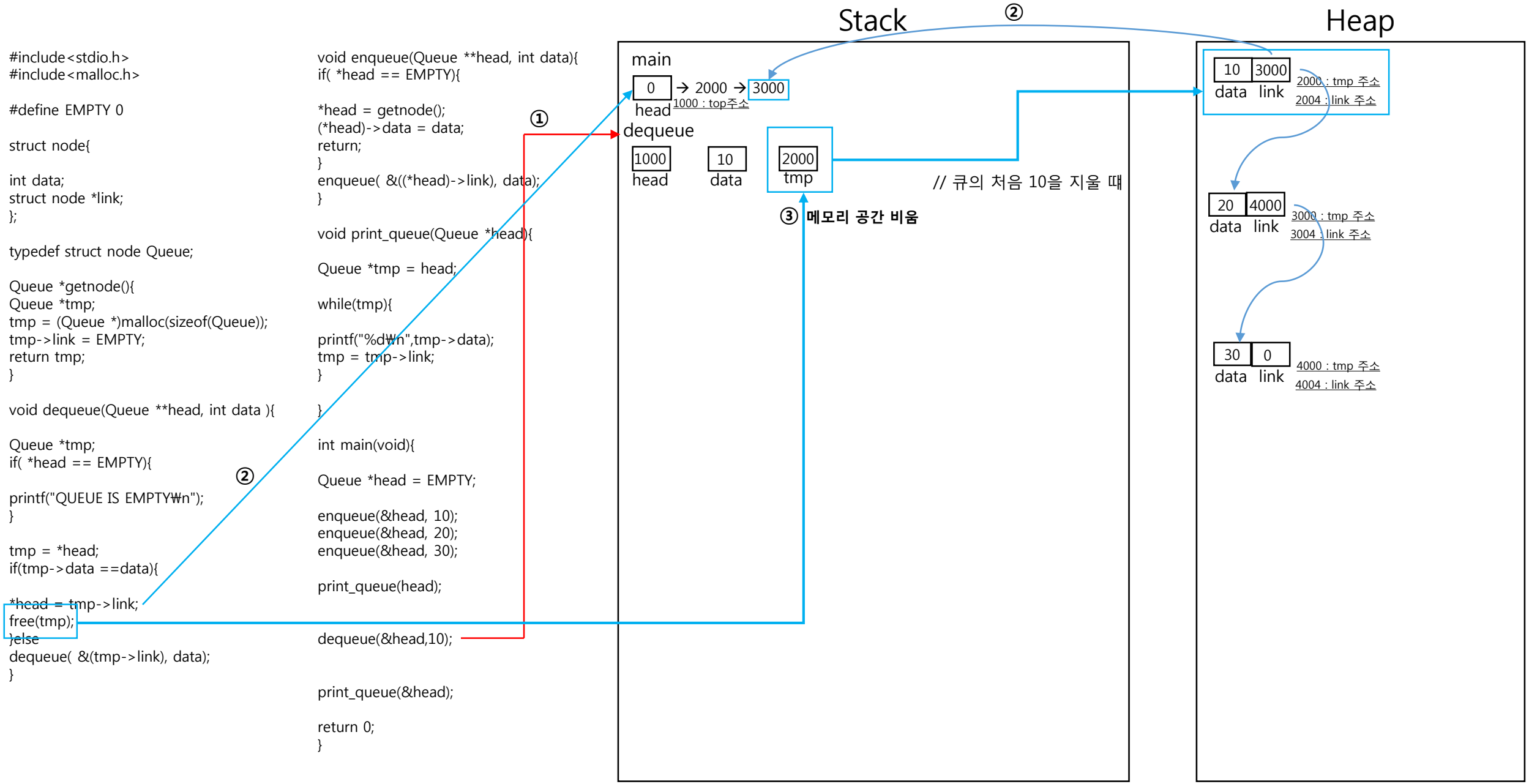
```
void enqueue(Queue **head, int data){
if( *head == EMPTY){

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

void 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);


dequeue(&head,20);


print_queue(&head);

return 0;
}
```



1.큐(Queue) 코드 이해(그림 그리기) – (dequeue 1)

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

#define EMPTY 0

struct node{

int data;
struct node *link;
};

typedef struct node Queue;

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

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

Queue *tmp;
if( *head == EMPTY){

printf("QUEUE IS EMPTY₩n");
}

tmp = *head;
if(tmp->data ==data){

*head = tmp->link;
free(tmp);
}else
dequeue( &(tmp->link), data);
}
```

```
void enqueue(Queue **head, int data){
if( *head == EMPTY){

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

void 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);


dequeue(&head,10);


print_queue(&head);

return 0;
}
```

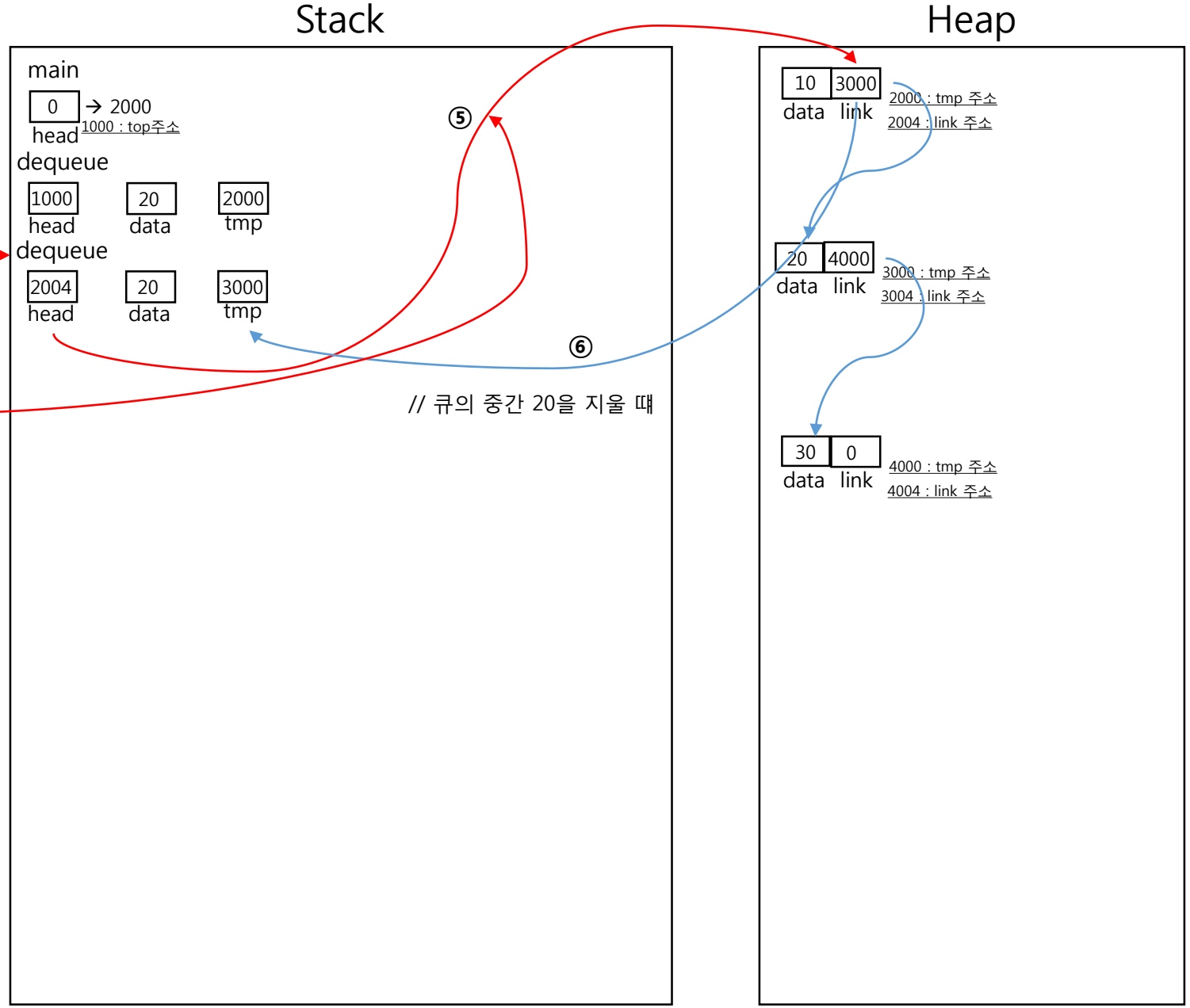


1.큐(Queue) 코드 이해(그림 그리기) – (dequeue 2)

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

#define EMPTY 0

struct node{

int data;
struct node *link;
};

typedef struct node Queue;

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

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

Queue *tmp;
if( *head == EMPTY){

printf("QUEUE IS EMPTY₩n");
}

tmp = *head;
if(tmp->data ==data){

*head = tmp->link;
free(tmp);
}else
dequeue( &(tmp->link), data);
}
```

④

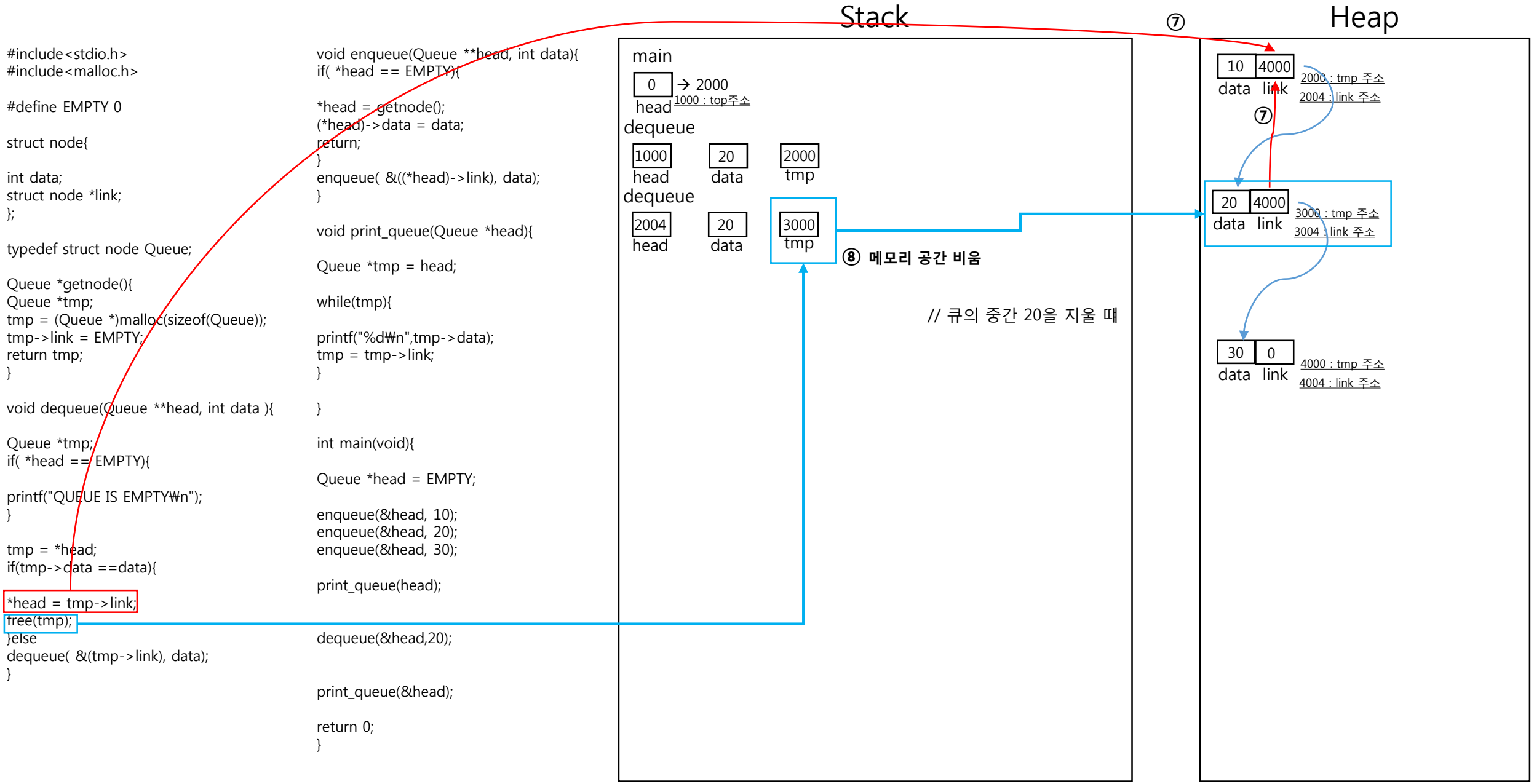
```
void enqueue(Queue **head, int data){
if( *head == EMPTY){

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

void 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);


dequeue(&head,20);


print_queue(&head);

return 0;
}
```



1.큐(Queue) 코드 이해(그림 그리기) – (dequeue 3)

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

#define EMPTY 0

struct node{

int data;
struct node *link;
};

typedef struct node Queue;

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

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

Queue *tmp;
if( *head == EMPTY){

printf("QUEUE IS EMPTY₩n");
}

tmp = *head;
if(tmp->data ==data){

*head = tmp->link;
free(tmp);
}else
dequeue( &(tmp->link), data);
}
```

```
void enqueue(Queue **head, int data){
if( *head == EMPTY){

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

void 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);


dequeue(&head,20);


print_queue(&head);

return 0;
}
```



1.큐(Queue) 코드 이해(그림 그리기) – (dequeue 4)

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

#define EMPTY 0

struct node{

int data;
struct node *link;
};

typedef struct node Queue;

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

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

Queue *tmp;
if( *head == EMPTY){

printf("QUEUE IS EMPTY₩n");
}

tmp = *head;
if(tmp->data ==data){

*head = tmp->link;
free(tmp);
}else
dequeue( &(tmp->link), data);
}
```

```
void enqueue(Queue **head, int data){
if( *head == EMPTY){

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

void 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);


dequeue(&head,20);


print_queue(&head);

return 0;
}
```