시스템프로그래밍기초 실습

Ch9. Structures and Unions

HANYANG UNIVERSITY

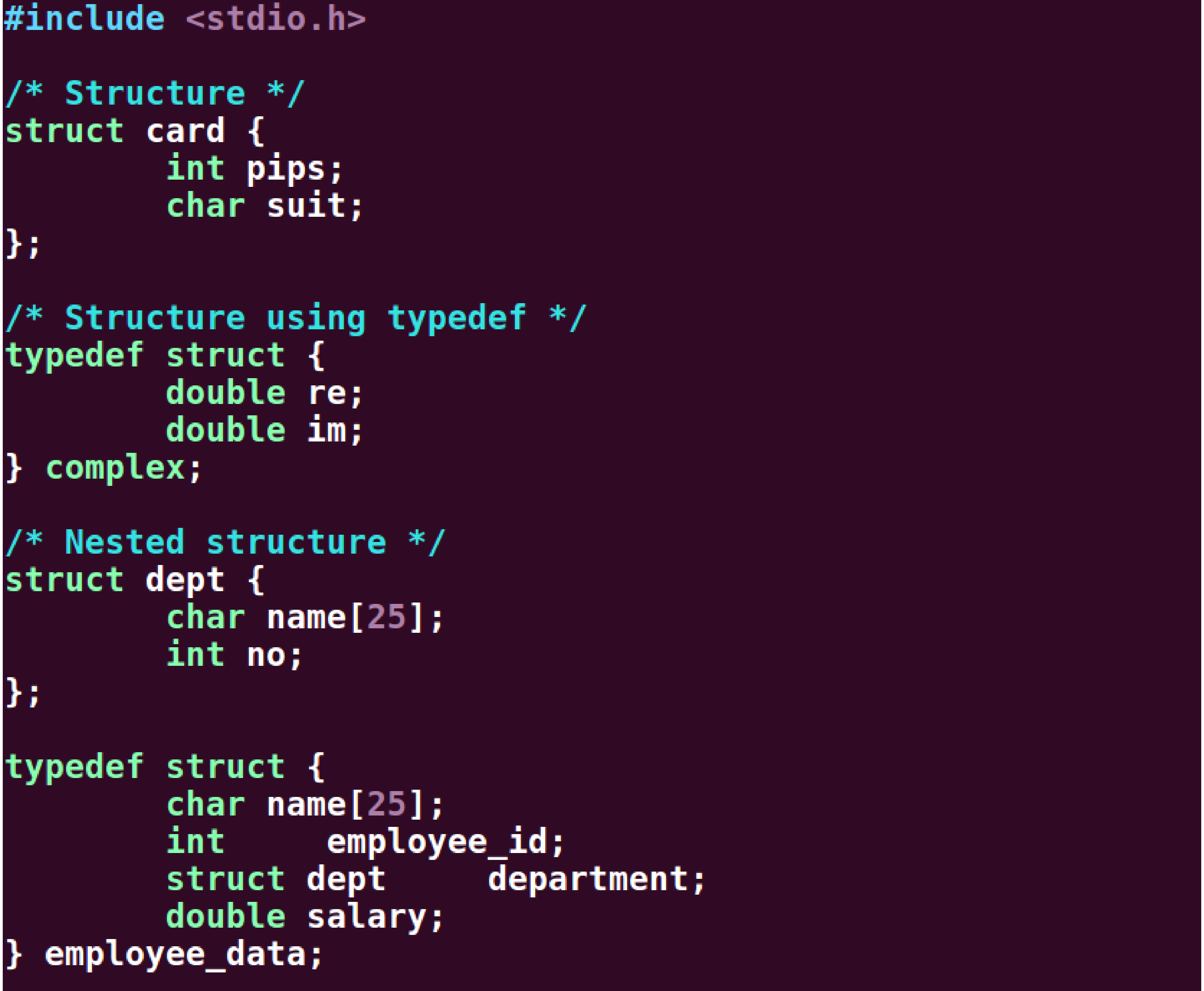
struct

```
struct tag_name {
        int a;
        double b;
        char c;
} var1, var2;
```

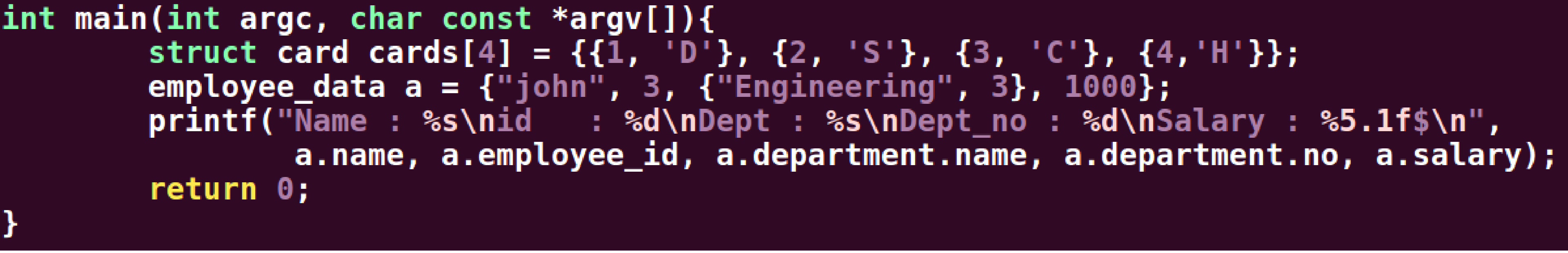
typedef struct

```
typedef struct {
        int a;
        double b;
        char c;
} type_name;
```

Tag_name만을 사용하는 것과 typedef를 사용하는 것은 어떤 차이를 가져오는가?

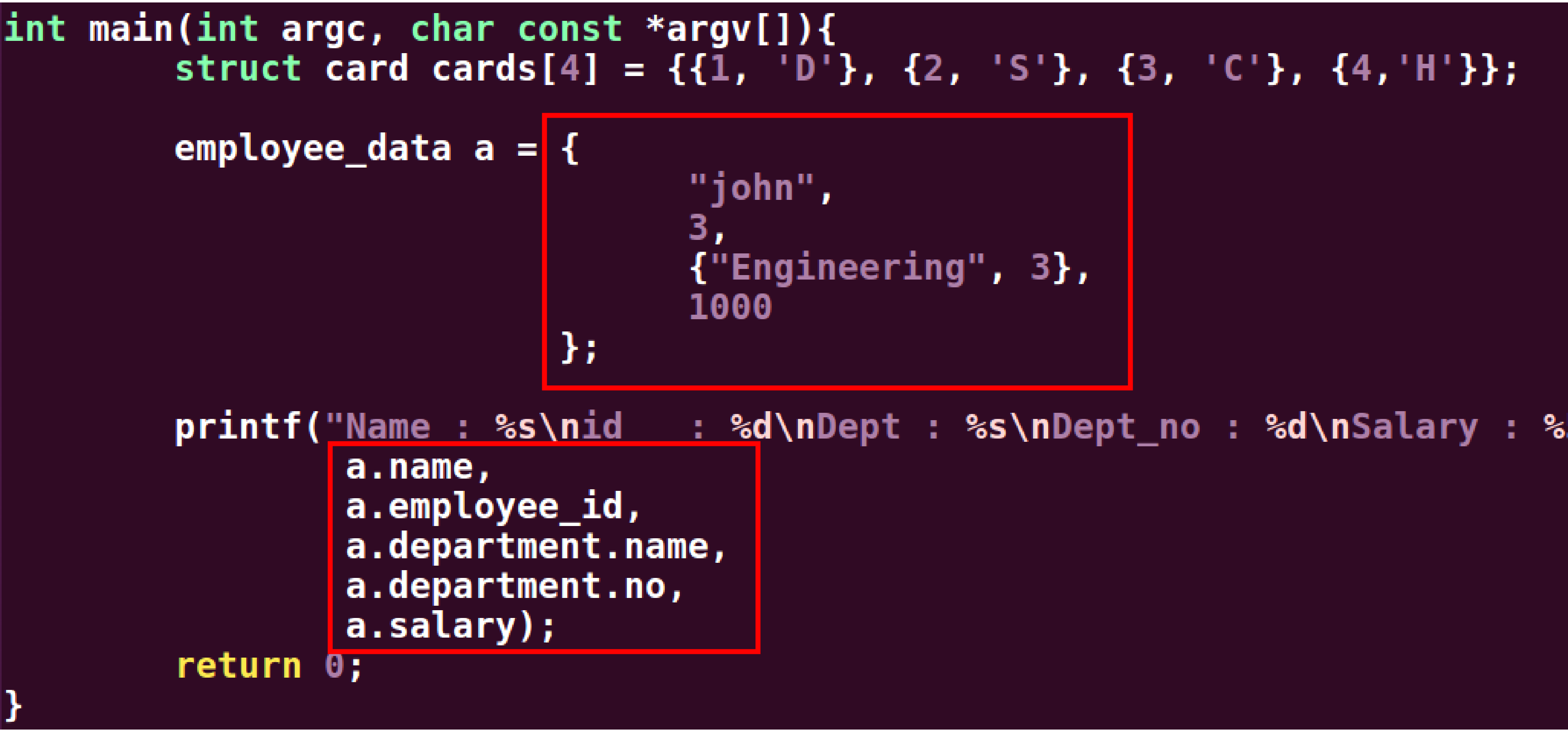


```c
#include <stdio.h>

/* Structure */
struct card {
        int pips;
        char suit;
};

/* Structure using typedef */
typedef struct {
        double re;
        double im;
} complex;

/* Nested structure */
struct dept {
        char name[25];
        int no;
};

typedef struct {
        char name[25];
        int     employee_id;
        struct dept     department;
        double salary;
} employee_data;
```



```c
int main(int argc, char const *argv[]){
        struct card cards[4] = {{1, 'D'}, {2, 'S'}, {3, 'C'}, {4,'H'}};
        employee_data a = {"john", 3, {"Engineering", 3}, 1000};
        printf("Name : %s\nid    : %d\nDept : %s\nDept_no : %d\nSalary : %5.1f$\n",
                a.name, a.employee_id, a.department.name, a.department.no, a.salary);
        return 0;
}
```

다시 정리하면..



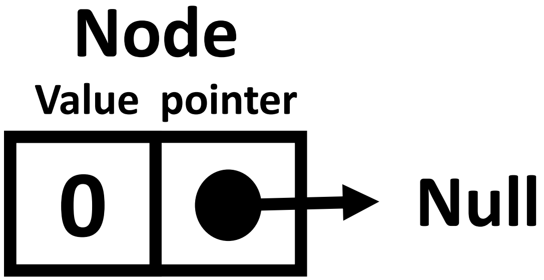
```c
int main(int argc, char const *argv[]){
        struct card cards[4] = {{1, 'D'}, {2, 'S'}, {3, 'C'}, {4,'H'}};

        employee_data a = {
                                "john",
                                3,
                                {"Engineering", 3},
                                1000
                          };

        printf("Name : %s\nid    : %d\nDept : %s\nDept_no : %d\nSalary : %
                a.name,
                a.employee_id,
                a.department.name,
                a.department.no,
                a.salary);
        return 0;
}
```

C에서 타입을 정의하는 방법

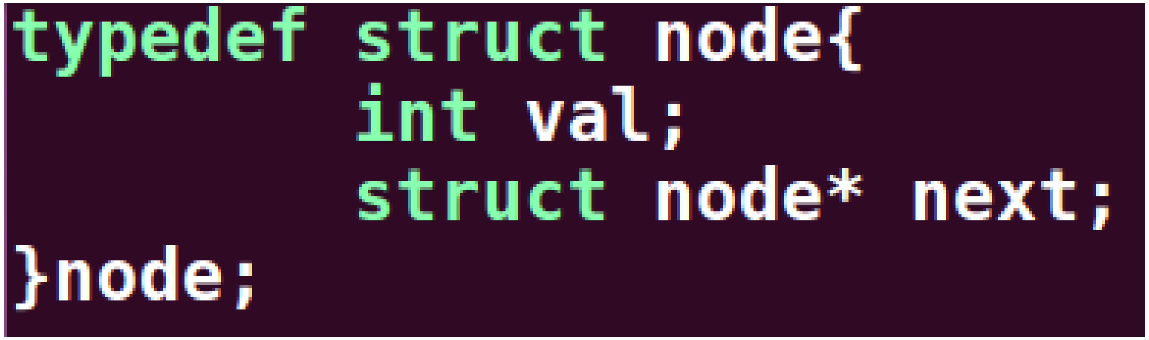
```
     ❶  ❷
⓿int x = 0;
❸x = 5;
```

⓿ **Type definition:**	define a type.
❶ **Declaration:**	declare a variable.
❷ **Initialization:**	initialize a value to a variable. It can be done at the time of declaration.
❸ **Assignment:**	assign value to a variable.

Node

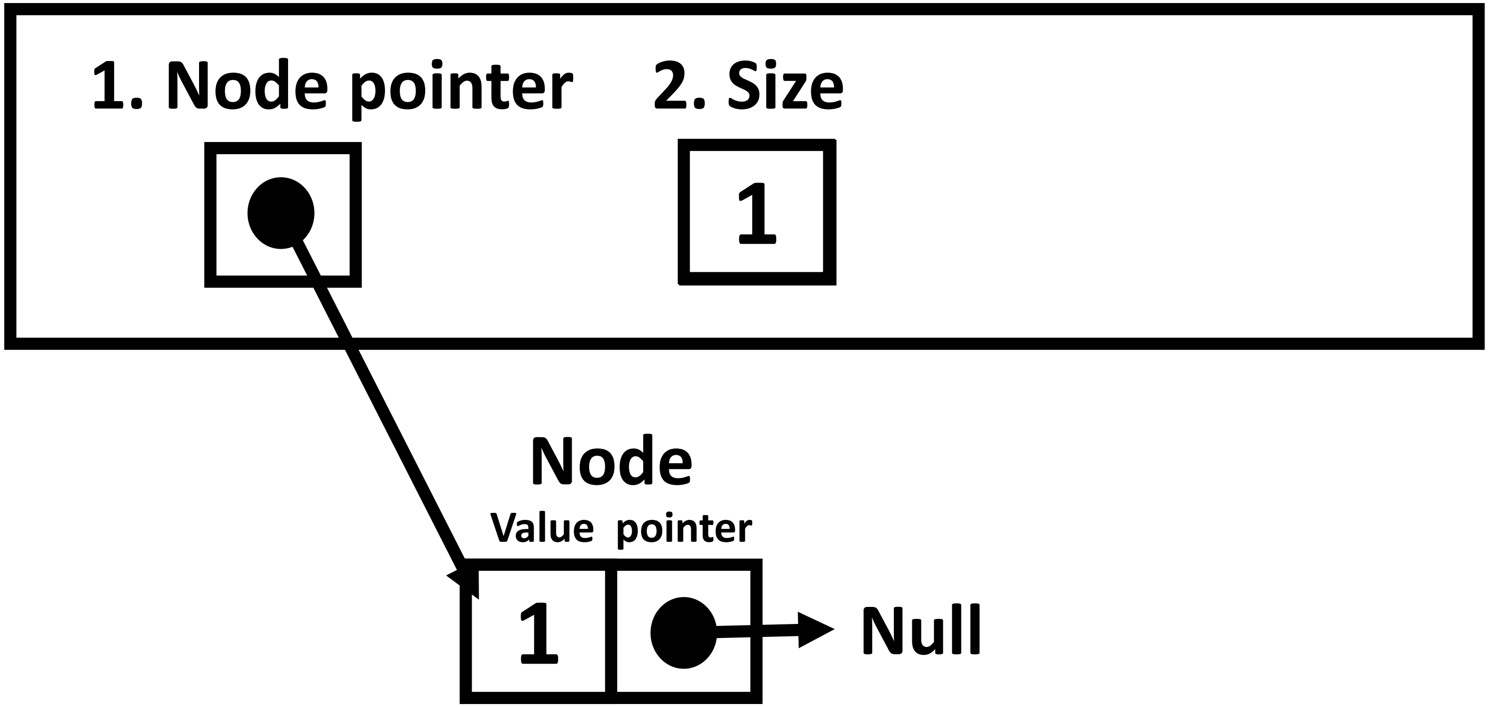


In directory chap9/headers/node.h



```
typedef struct node{
        int val;
        struct node* next;
}node;
```

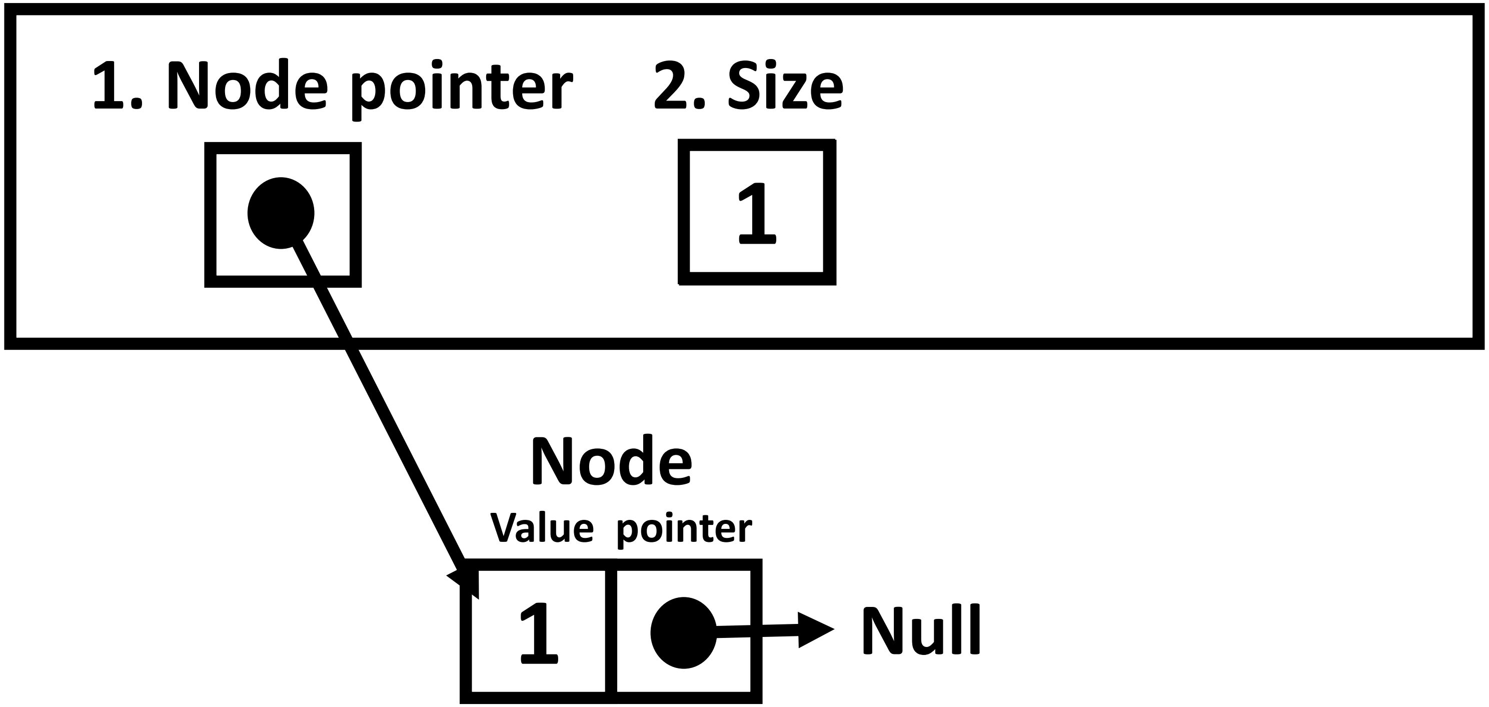
Linked List



appendTo(list,newnode(1));

그렇다면 한 개보다 많은 노드를 추가하면?

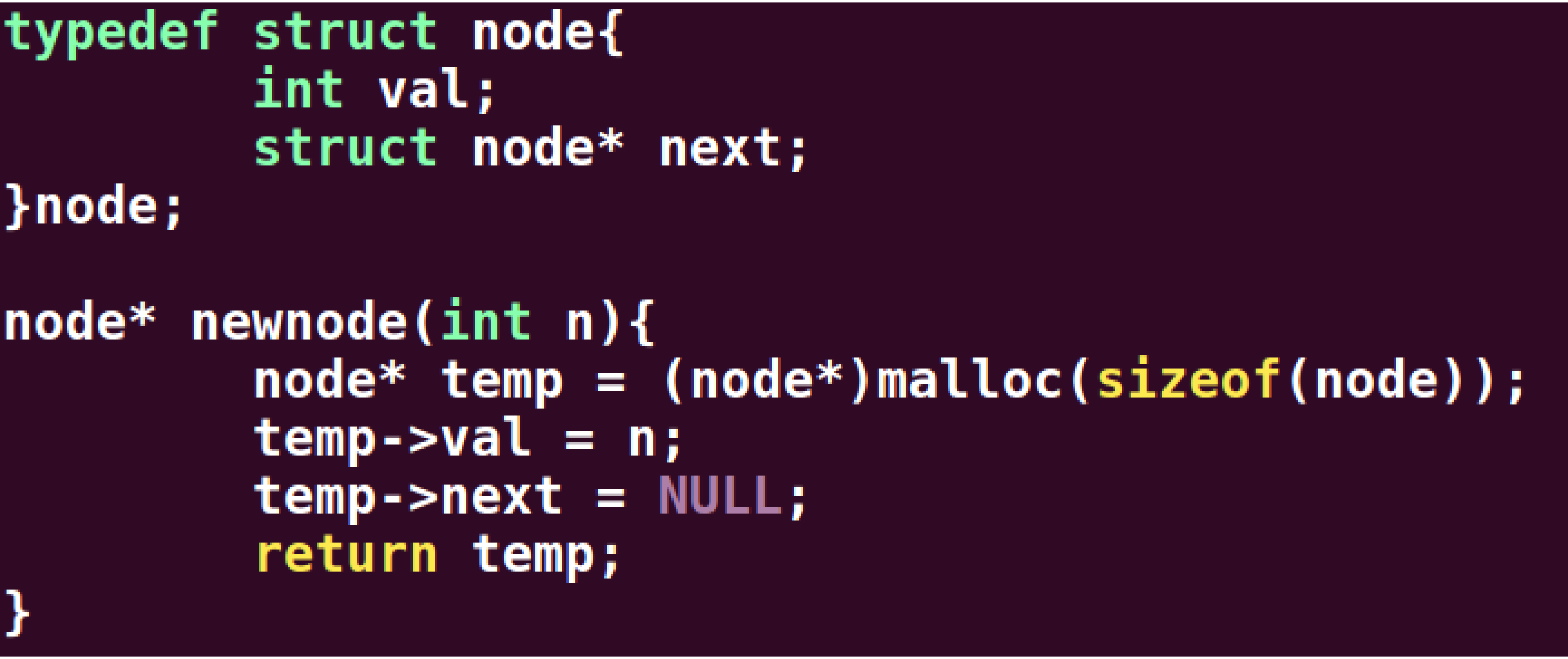
Linked List



delAt(list,0);

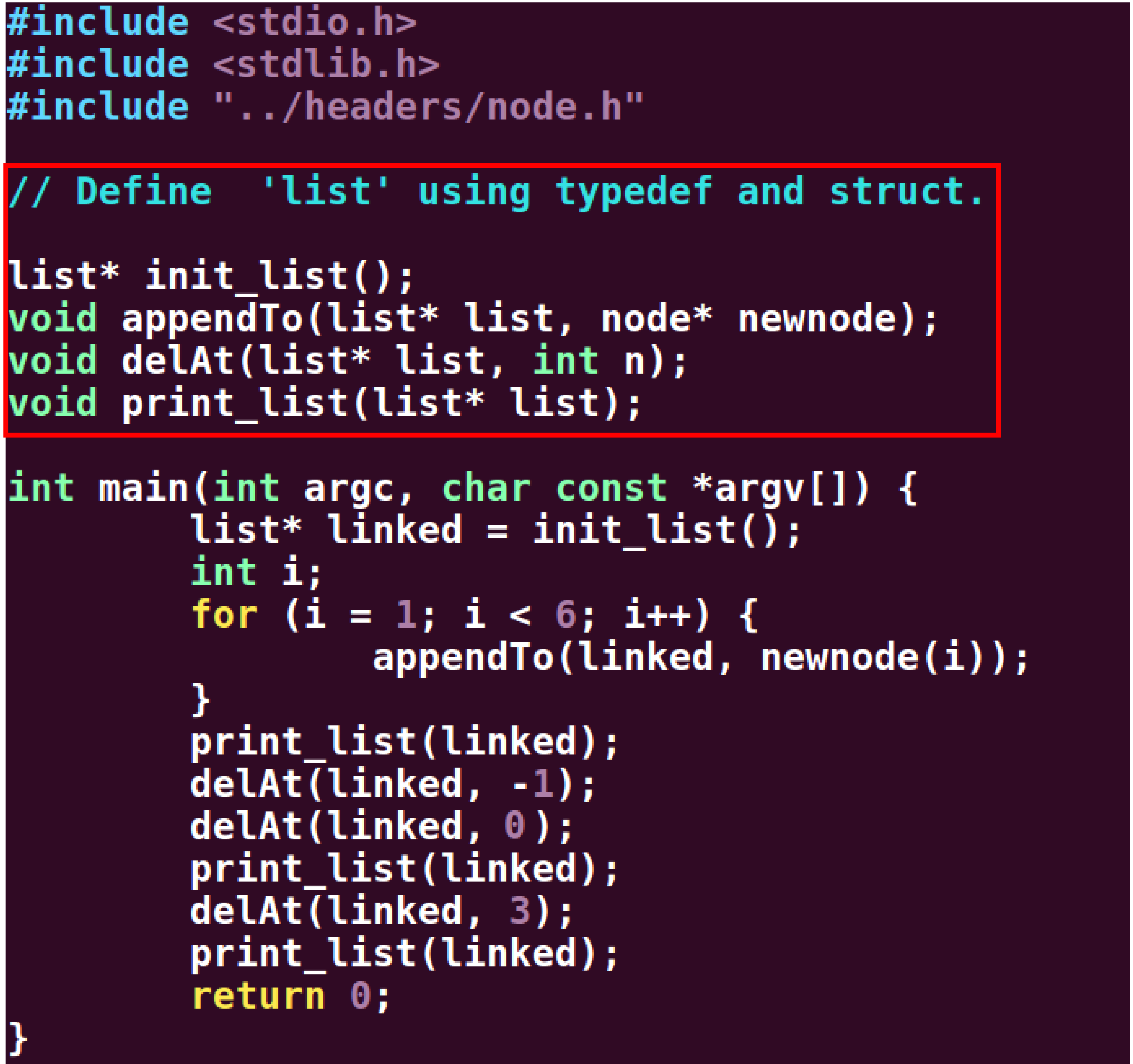
그렇다면 한 개보다 많은 노드를 추가하면?
삭제 시에 0개의 노드를 가진 경우에는?

chap9/headers/node.h



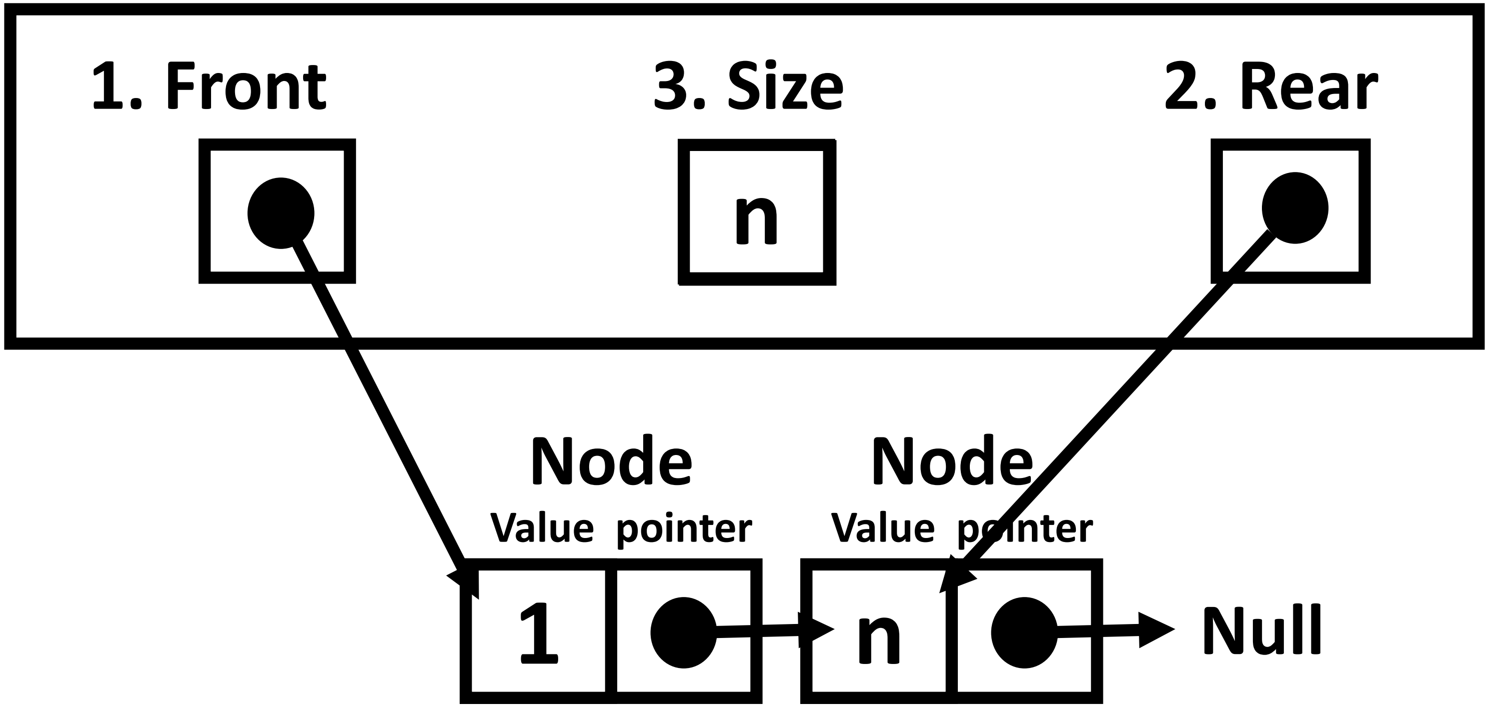
```c
typedef struct node{
        int val;
        struct node* next;
}node;

node* newnode(int n){
        node* temp = (node*)malloc(sizeof(node));
        temp->val = n;
        temp->next = NULL;
        return temp;
}
```

chap9/sources/list.c

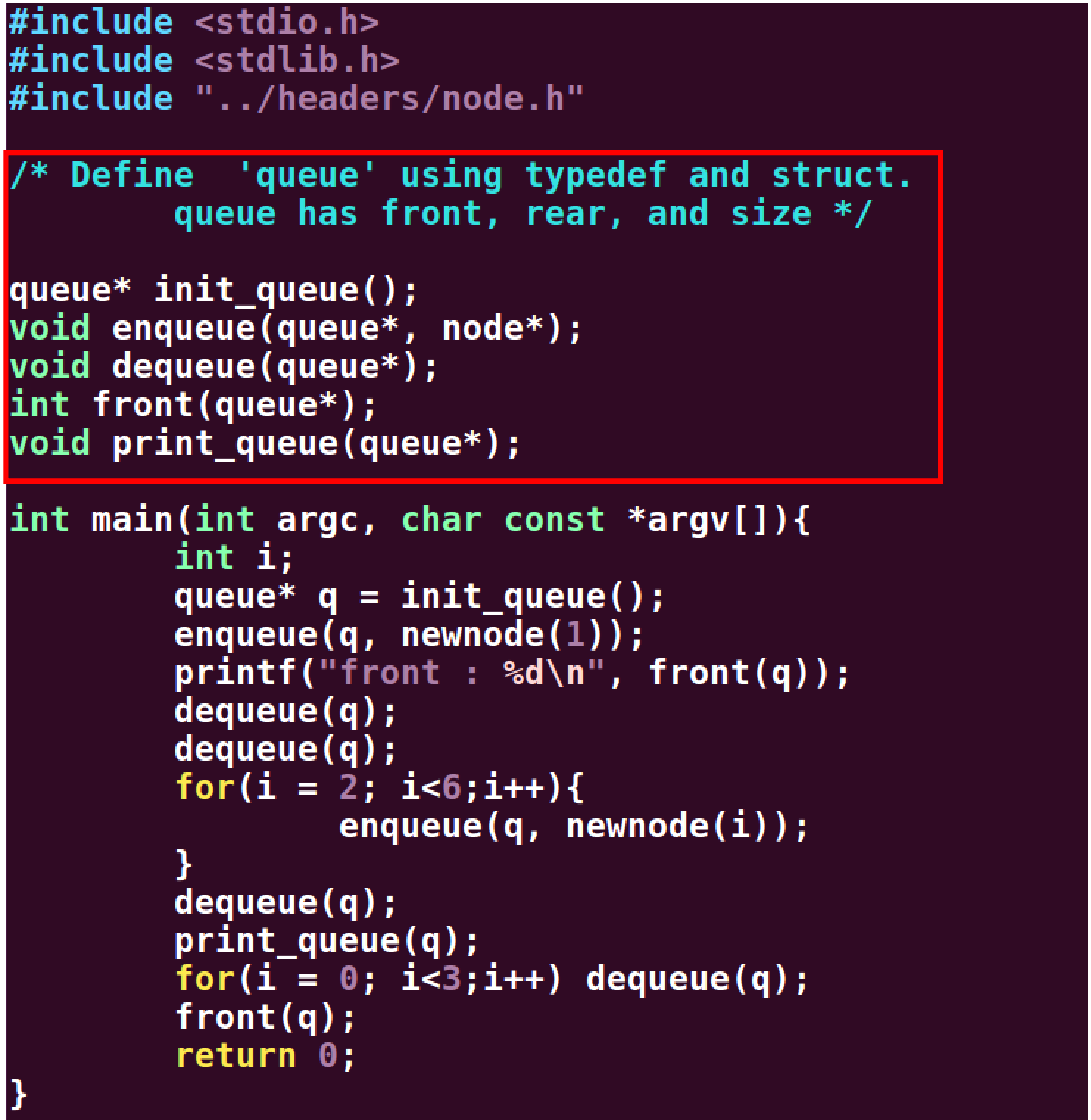


```
#include <stdio.h>
#include <stdlib.h>
#include "../headers/node.h"

// Define  'list' using typedef and struct.

list* init_list();
void appendTo(list* list, node* newnode);
void delAt(list* list, int n);
void print_list(list* list);

int main(int argc, char const *argv[]) {
        list* linked = init_list();
        int i;
        for (i = 1; i < 6; i++) {
                appendTo(linked, newnode(i));
        }
        print_list(linked);
        delAt(linked, -1);
        delAt(linked, 0);
        print_list(linked);
        delAt(linked, 3);
        print_list(linked);
        return 0;
}
```

Queue

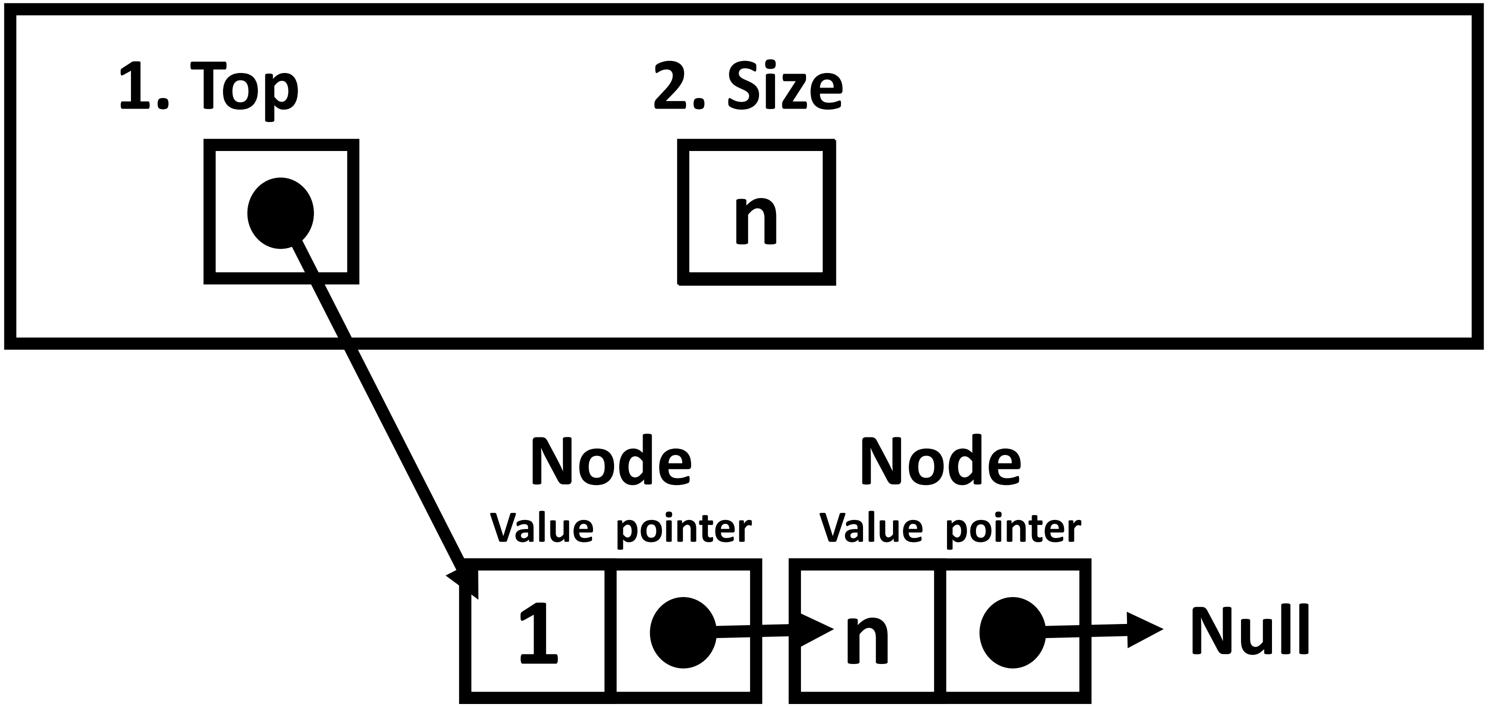

1. Front
3. Size
2. Rear
n
Node
Node
Value pointer
Value pointer
1
n
Null

chap9/sources/queue.c

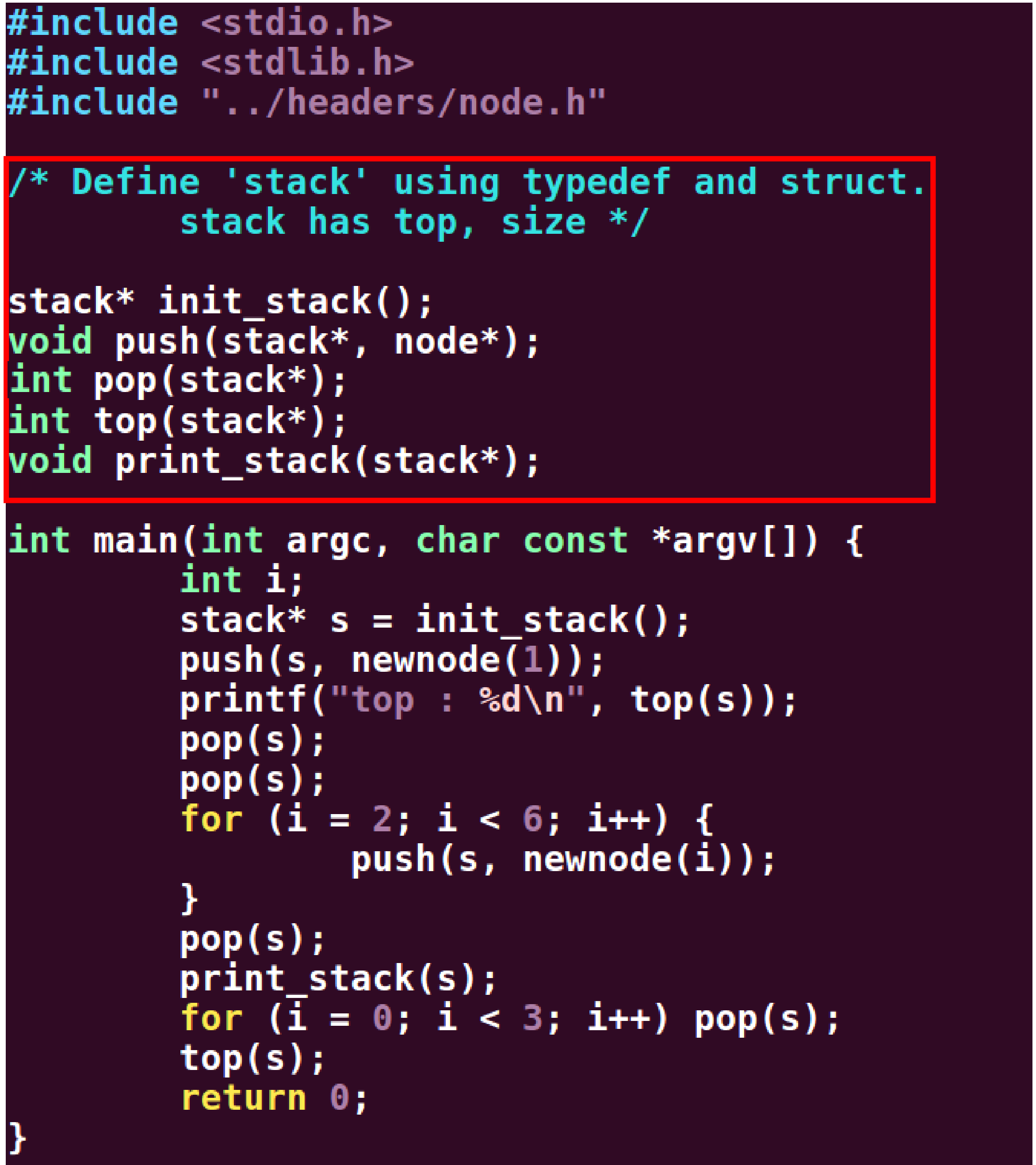


```c
#include <stdio.h>
#include <stdlib.h>
#include "../headers/node.h"

/* Define  'queue' using typedef and struct.
        queue has front, rear, and size */

queue* init_queue();
void enqueue(queue*, node*);
void dequeue(queue*);
int front(queue*);
void print_queue(queue*);

int main(int argc, char const *argv[]){
        int i;
        queue* q = init_queue();
        enqueue(q, newnode(1));
        printf("front : %d\n", front(q));
        dequeue(q);
        dequeue(q);
        for(i = 2; i<6;i++){
                enqueue(q, newnode(i));
        }
        dequeue(q);
        print_queue(q);
        for(i = 0; i<3;i++) dequeue(q);
        front(q);
        return 0;
}
```

Stack

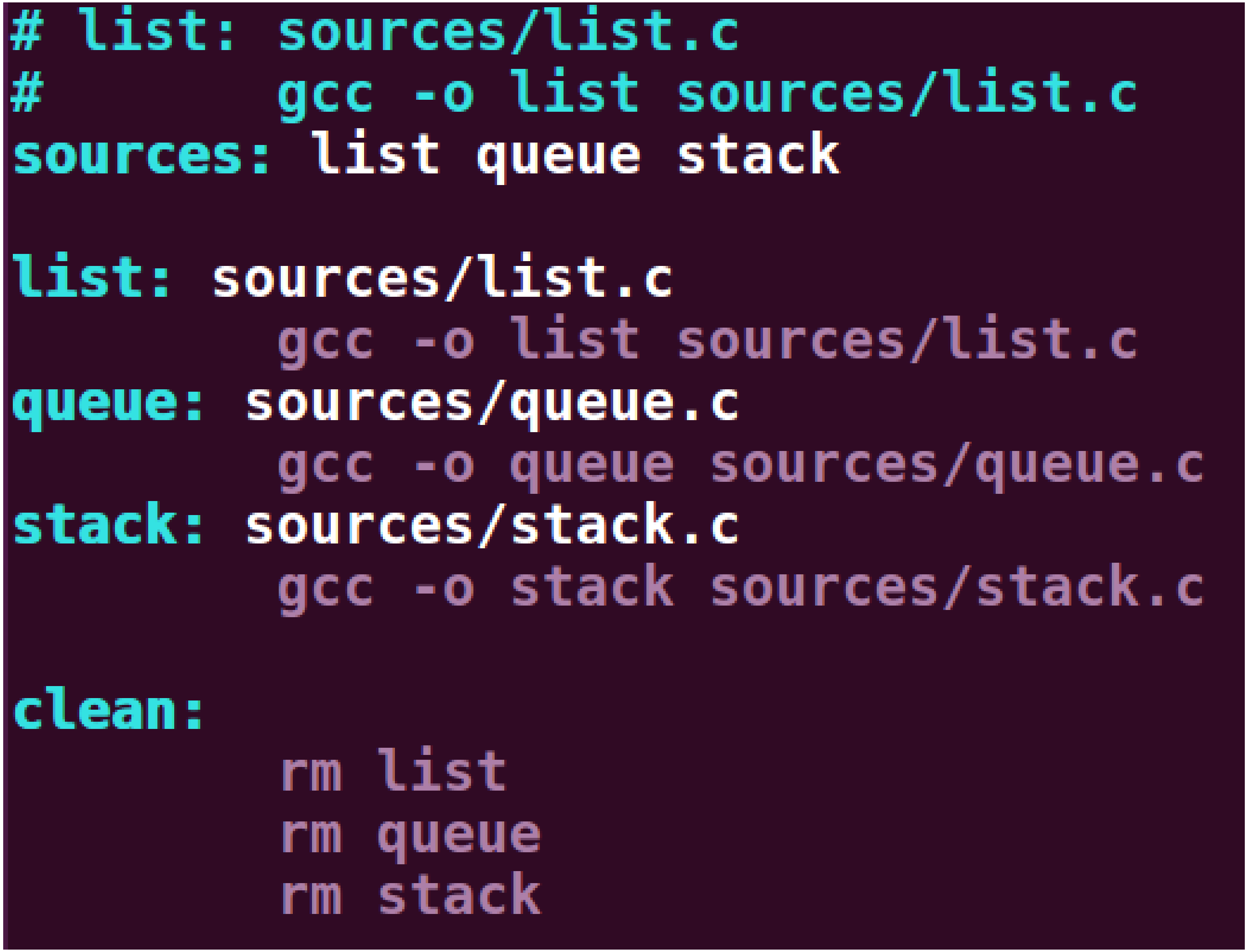


chap9/sources/stack.c



```c
#include <stdio.h>
#include <stdlib.h>
#include "../headers/node.h"

/* Define 'stack' using typedef and struct.
        stack has top, size */

stack* init_stack();
void push(stack*, node*);
int pop(stack*);
int top(stack*);
void print_stack(stack*);

int main(int argc, char const *argv[]) {
        int i;
        stack* s = init_stack();
        push(s, newnode(1));
        printf("top : %d\n", top(s));
        pop(s);
        pop(s);
        for (i = 2; i < 6; i++) {
                push(s, newnode(i));
        }
        pop(s);
        print_stack(s);
        for (i = 0; i < 3; i++) pop(s);
        top(s);
        return 0;
}
```

chap9/Makefile

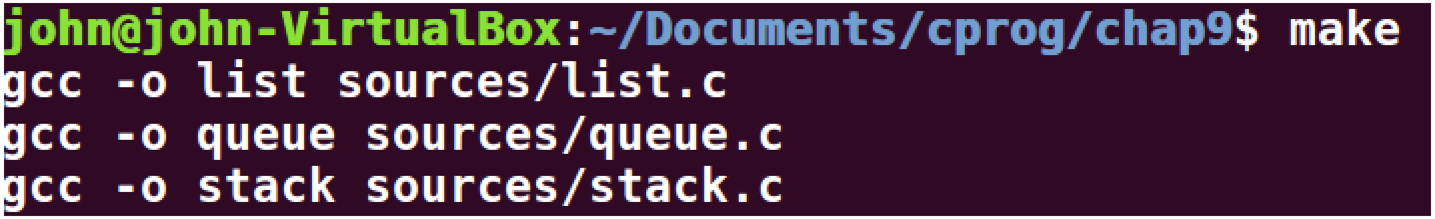


```
# list: sources/list.c
#	gcc -o list sources/list.c
sources: list queue stack

list: sources/list.c
	gcc -o list sources/list.c
queue: sources/queue.c
	gcc -o queue sources/queue.c
stack: sources/stack.c
	gcc -o stack sources/stack.c

clean:
	rm list
	rm queue
	rm stack
```

어떻게 컴파일 하는가?

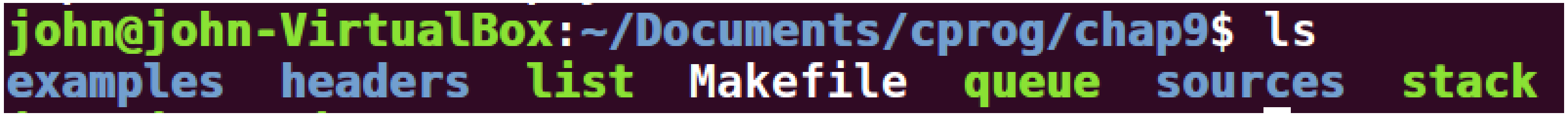
1. Makefile을 작성하여라.

2. ‘Make’



```
john@john-VirtualBox:~/Documents/cprog/chap9$ make
gcc -o list sources/list.c
gcc -o queue sources/queue.c
gcc -o stack sources/stack.c
```

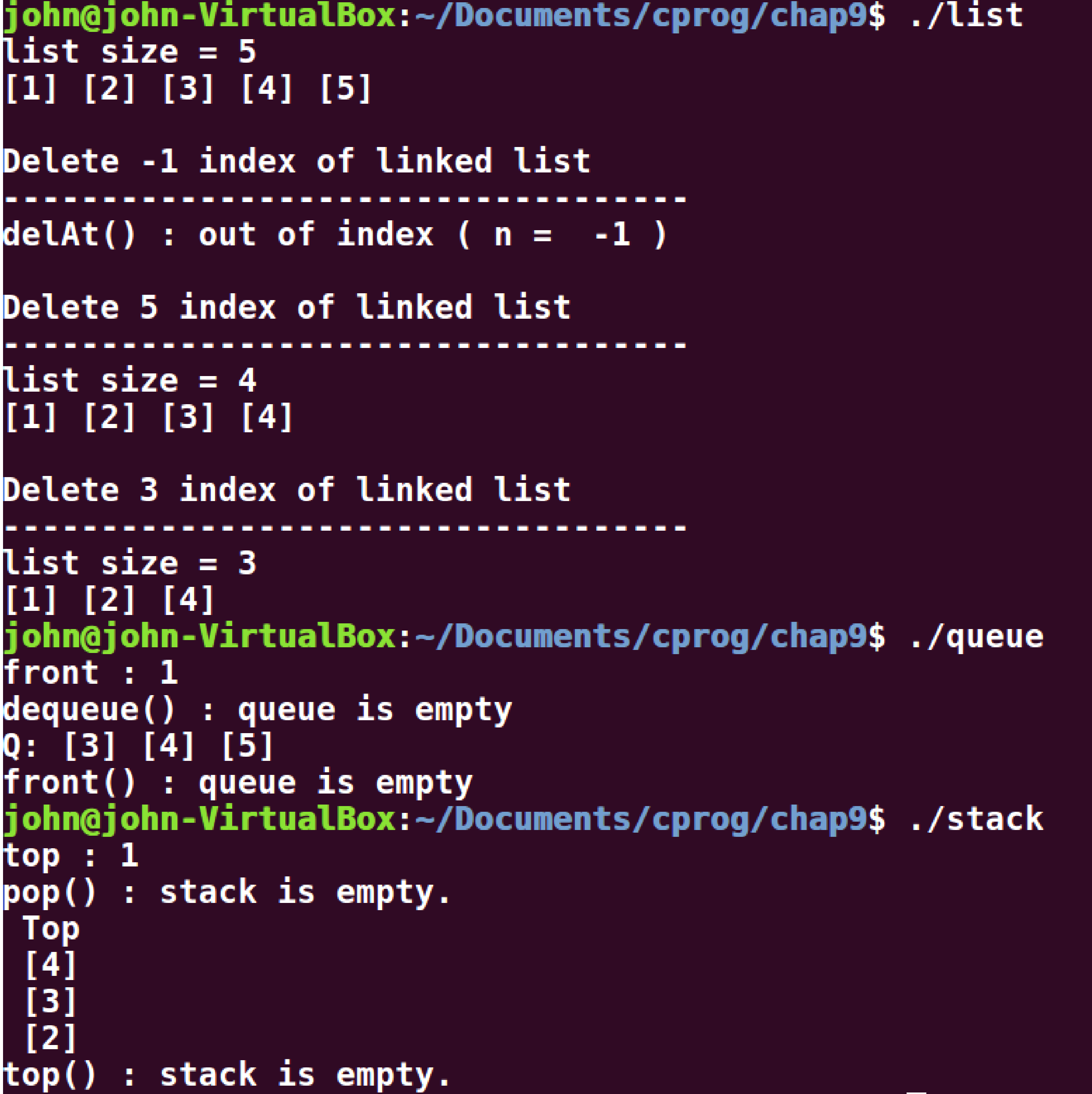
2. 각 실행파일이 생성된다.



```
john@john-VirtualBox:~/Documents/cprog/chap9$ ls
examples  headers  list  Makefile  queue  sources  stack
```

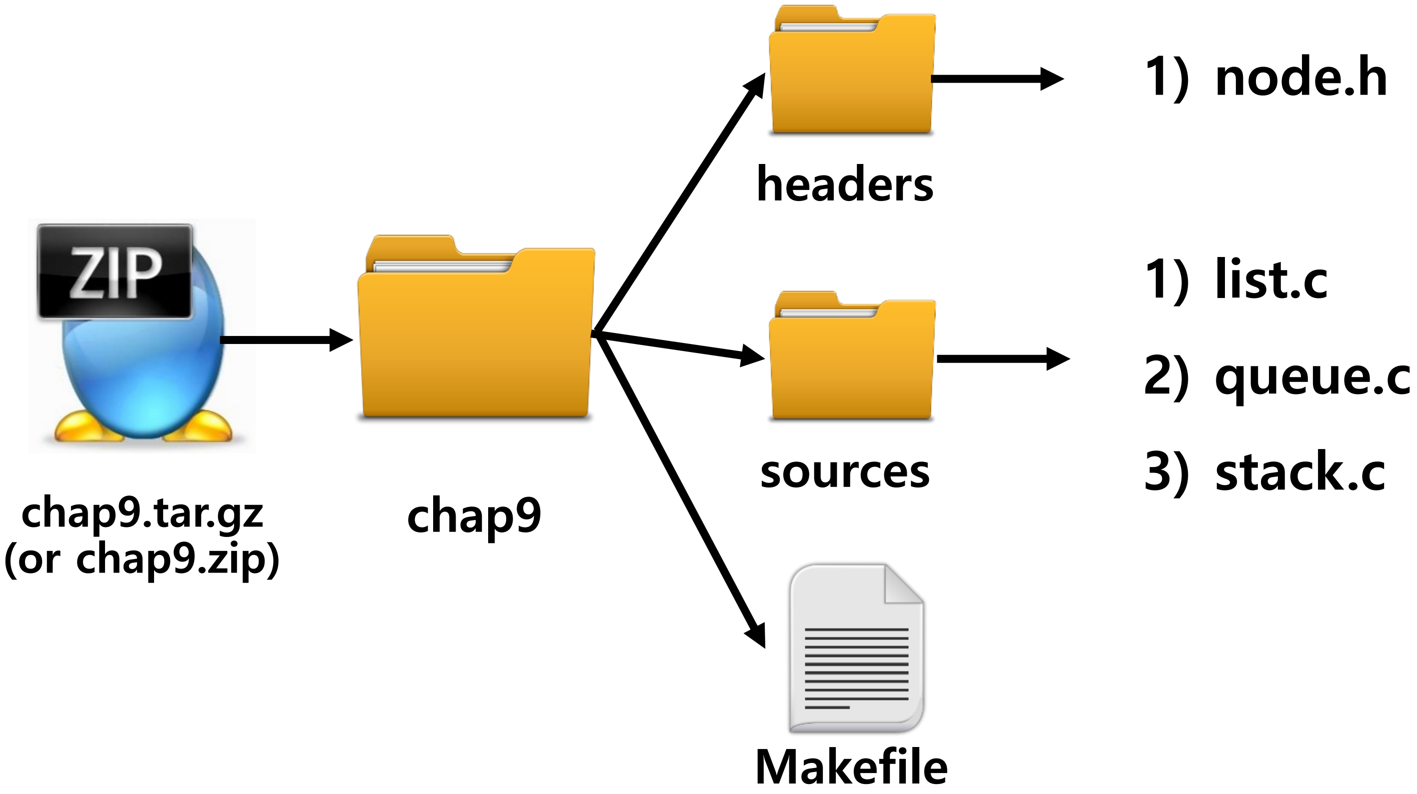
green color : list, queue, stack

결과화면



```
john@john-VirtualBox:~/Documents/cprog/chap9$ ./list
list size = 5
[1] [2] [3] [4] [5]

Delete -1 index of linked list
-----------------------------------
delAt() : out of index ( n =  -1 )

Delete 5 index of linked list
-----------------------------------
list size = 4
[1] [2] [3] [4]

Delete 3 index of linked list
-----------------------------------
list size = 3
[1] [2] [4]
john@john-VirtualBox:~/Documents/cprog/chap9$ ./queue
front : 1
dequeue() : queue is empty
Q: [3] [4] [5]
front() : queue is empty
john@john-VirtualBox:~/Documents/cprog/chap9$ ./stack
top : 1
pop() : stack is empty.
 Top
 [4]
 [3]
 [2]
top() : stack is empty.
```

< 과제 정리 형식 >



chap9.tar.gz
(or chap9.zip)

chap9

headers

1) node.h

sources

1) list.c

2) queue.c

3) stack.c

Makefile