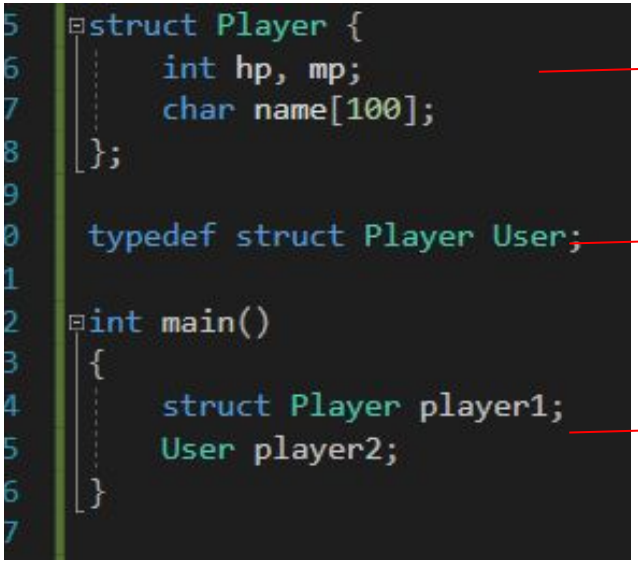
구조체



struct Player 라는 자료형 생성
int형 2개, char형 100개 -> 108 Byte

struct Player 자료형을 User 라고 새 정의

변수의 선언.
struct Player, User 동시 사용 가능

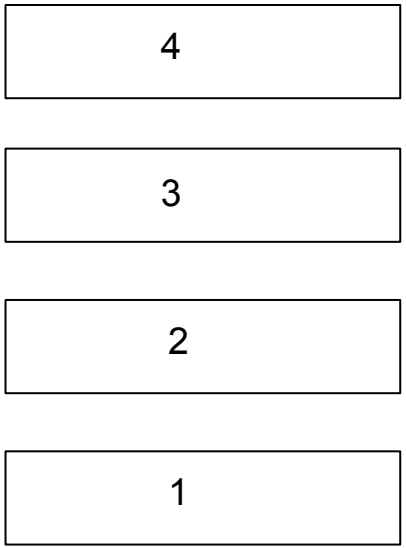
```c
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <string.h> // strcpy() 함수 내포

typedef struct Human Person;
struct Human {
    char name[100];
    int age;
    int weight;
};

int main()
{
    Person person1;

    //첫번째 인자에 두번째 인자의 문자열 복사
    strcpy(person1.name, "정태준");
    person1.age = 24;
    person1.weight = 58;

    printf("Name : %s\n", person1.name);
    printf("Age : %d\n", person1.age);
    printf("Weight : %d\n", person1.weight);
}
```

구조체변수 . 내부변수

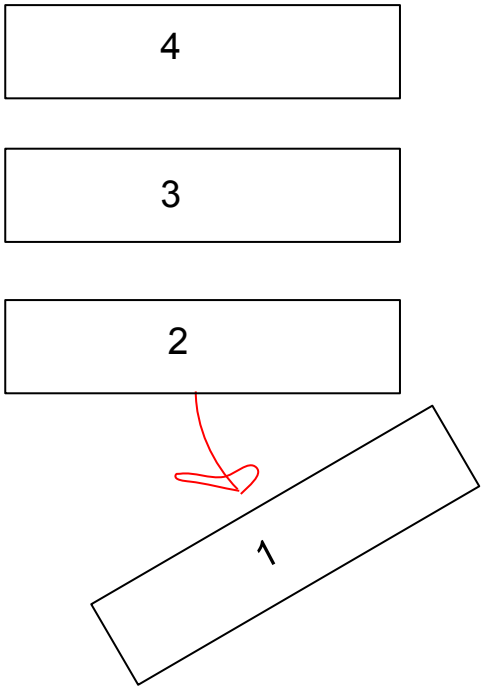
방식으로 '.' 을 통해 구조체 내부에 접근

```c
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <string.h>

typedef struct Human Person;
struct Human {
    char name[100];
    int age;
    int weight;
};

int main()
{
    Person person1;
    Person* ptr = &person1;

    strcpy(ptr->name, "정태준");
    ptr->age = 24;
    ptr->weight = 58;

    printf("Name : %s\n", ptr->name);
    printf("Age : %d\n", ptr->age);
    printf("Weight : %d\n", ptr->weight);
}
```

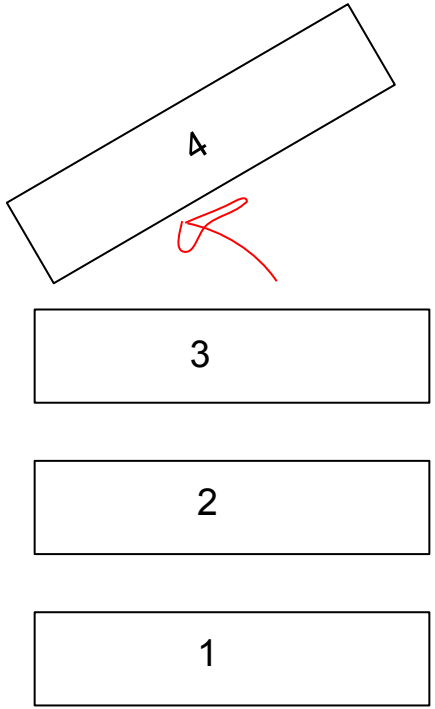
포인터변수 -> 내부변수
=
(*포인터변수).내부변수



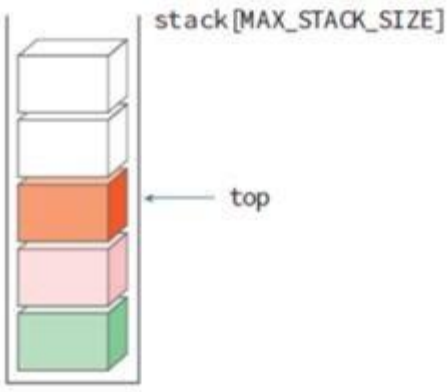
큐 (QUEUE) : 선입선출(FIFO)



스택 (STACK) : 후입선출(LIFO)



스택 구현



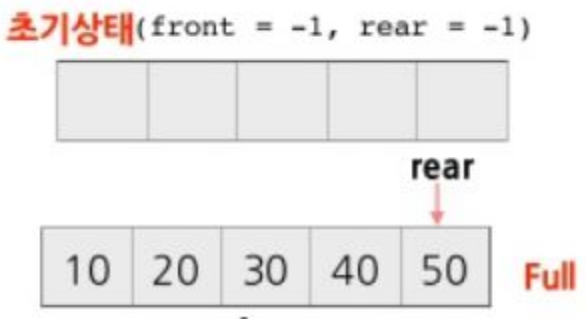
1. void push(), int pop() 함수 구현
2. 스택이 꽉찼을때 push() 함수 예외처리 구현
3. 스택이 비었을때 pop()함수 예외처리 구현

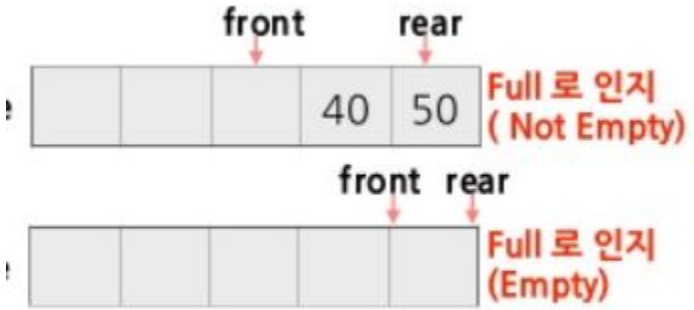
<tip>
top 변수는 데이터가 들어있는 가장 끝 인덱스 값을 저장

이 **top** 변수를 통해 알맞은 배열의 인덱스를 참조하여 **push(), pop()** 을 할 수 있다

큐 (Queue)

front = -1, rear = -1 로 시작
front 는 (제일 앞 인덱스 - 1), rear 는 (제일 뒤 인덱스) 를 뜻함





pop() 했을때 front 가 뒤로 이동하면서
사용할 수 있는 배열의 용량이 점점 줄어듬

문제

큐를 구현한다

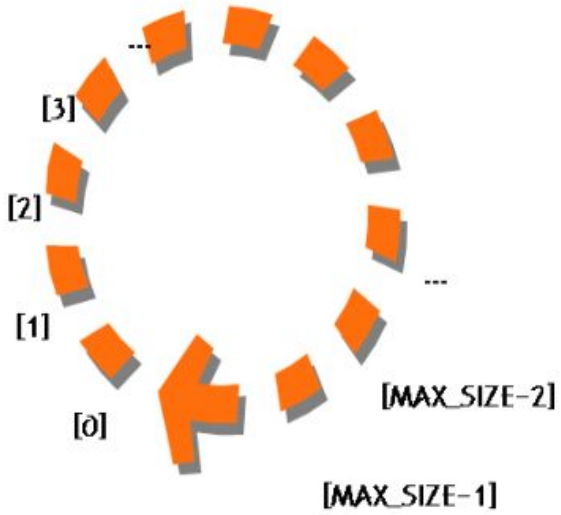
<조건>

1. push 와 pop 함수를 구현
2. push 중 큐가 꽉찼을 경우 발생하는 문구가 출력되도록 구현
3. pop 중 큐가 비었을 경우 발생하는 문구가 출력되도록 구현

<팁>

1. rear 가 마지막 인덱스를 참조하고있을때 큐가 꽉차있다고 간주
2. front 와 rear 값이 같을때 큐가 비어있다고 간주

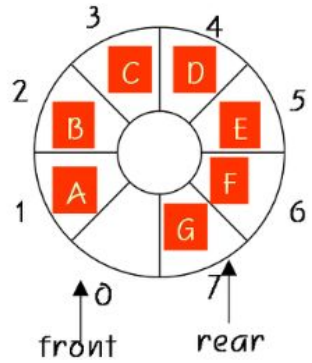
원형 큐



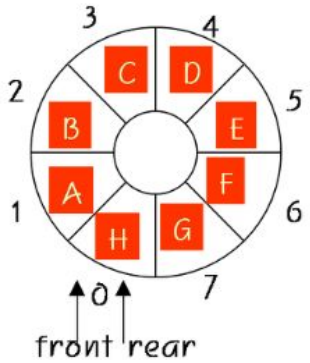
원형큐

배열을 원형으로 사용해 큐를 구현

* 항상 front는 비어있어야 함



(b) 포화상태



(c) 오류상태

공백상태와 포화상태

공백상태 : front == rear

포화상태 : front % M == (rear+1)%M

공백상태와 포화상태를 구별하기 위해 하나의 공간은 항상 비워둠

문제

원형 큐를 구현한다

<조건>

1. 구조체를 통해 Queue 구현 (int arr[QUEUE_SIZE]; int front = 0; int rear = 0;)
2. push 함수와 pop 함수 구현
3. main 함수에 push 와 pop 함수를 활용하는 코드를 구성

<팁>

어떤 값에 %100 연산한 값은 항상 0 ~ 99 사이

응용하여 %QUEUE_SIZE 를 한다면 값은 0 ~ (QUEUE_SIZE - 1) 사이