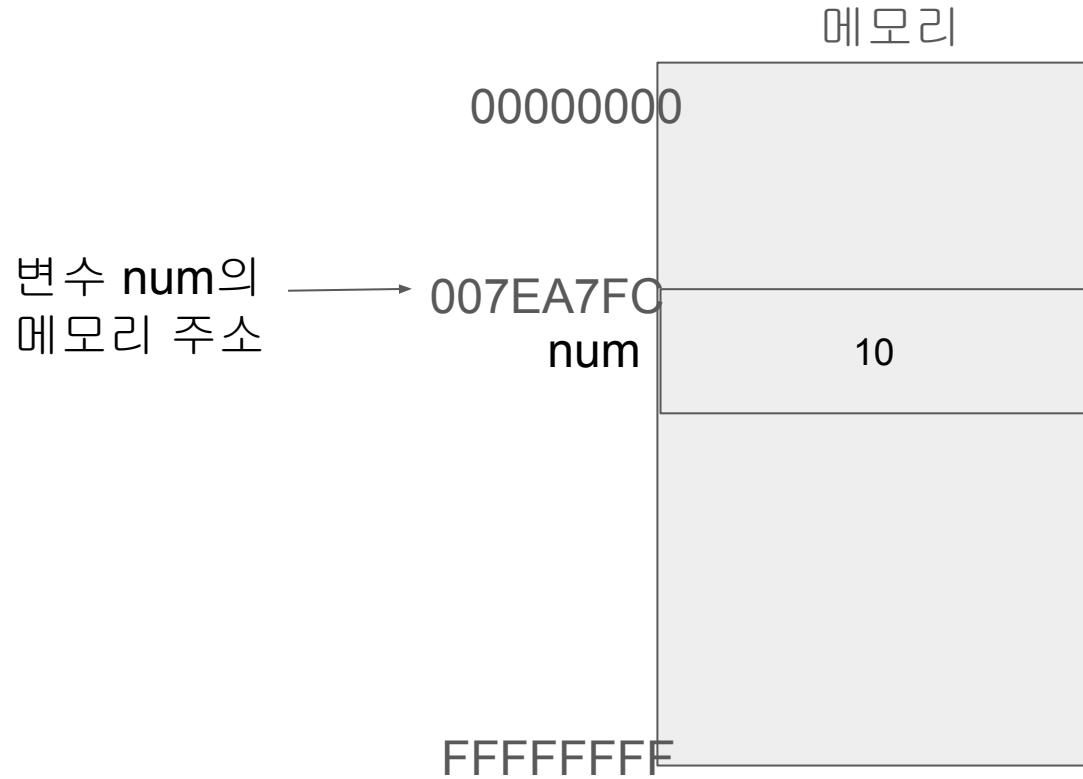
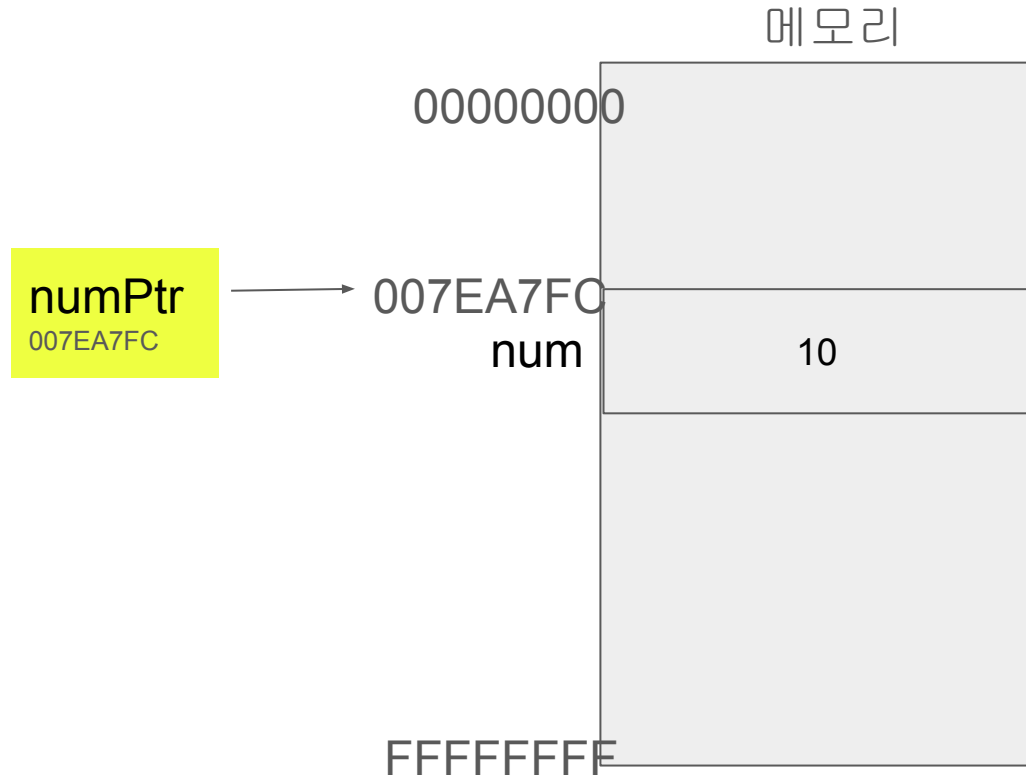


C

```
int num = 10;
```

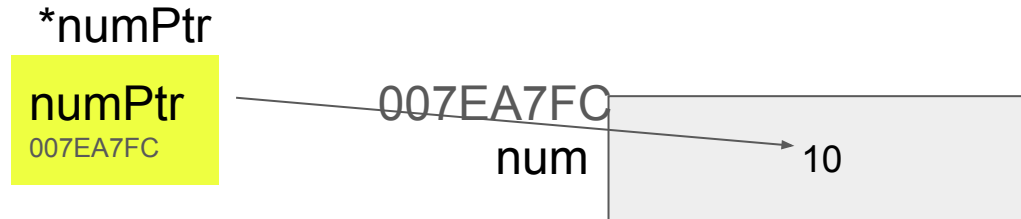
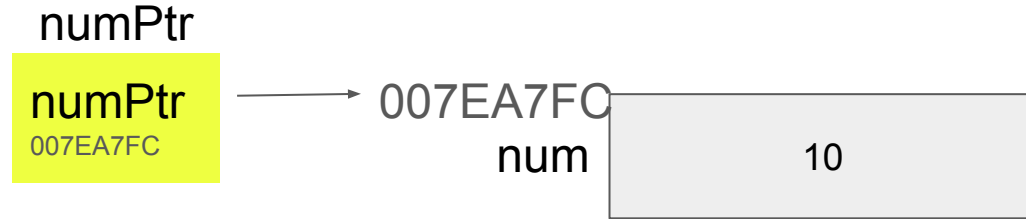


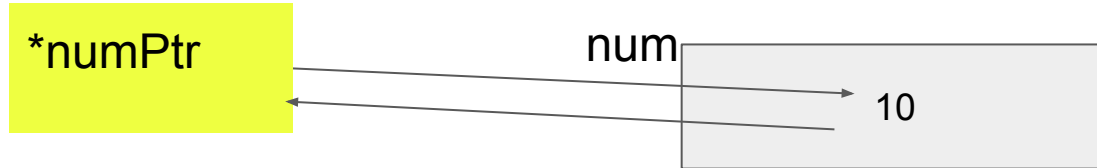
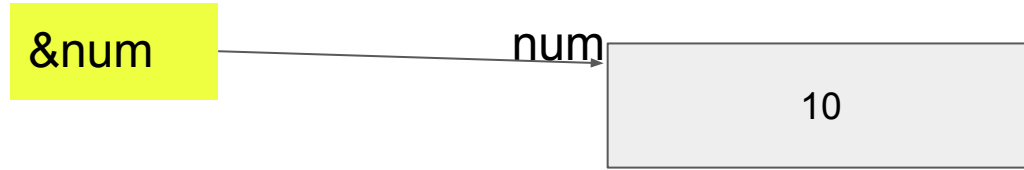
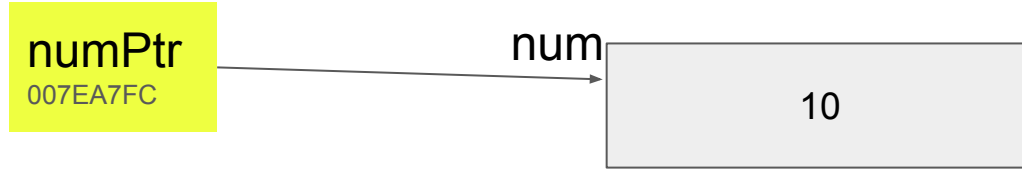
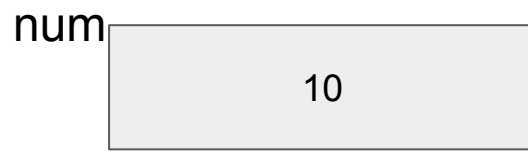
```
int num = 10;
```



```
int num = 10;
```

```
int* numPtr;
```





****numPtr2**
numPtr1의 메모리 주소

***numPtr1**
num의 메모리 주소

num

10



```
int a = 5;  
int *pA;  
pA = &a;
```

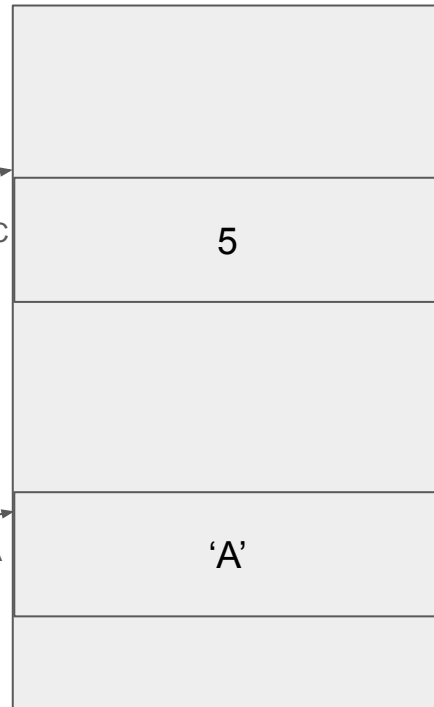
pA
007EA7FC

a
007EA7FC

```
char c = 'A';  
char *pC;  
pC = &c;
```

pC
007EA7AA

c
007EA7AA



```
int arr[5];
```

1번째 원소

2번째 원소

3번째 원소

4번째 원소

5번째 원소

arr

int	int	int	int	int
-----	-----	-----	-----	-----

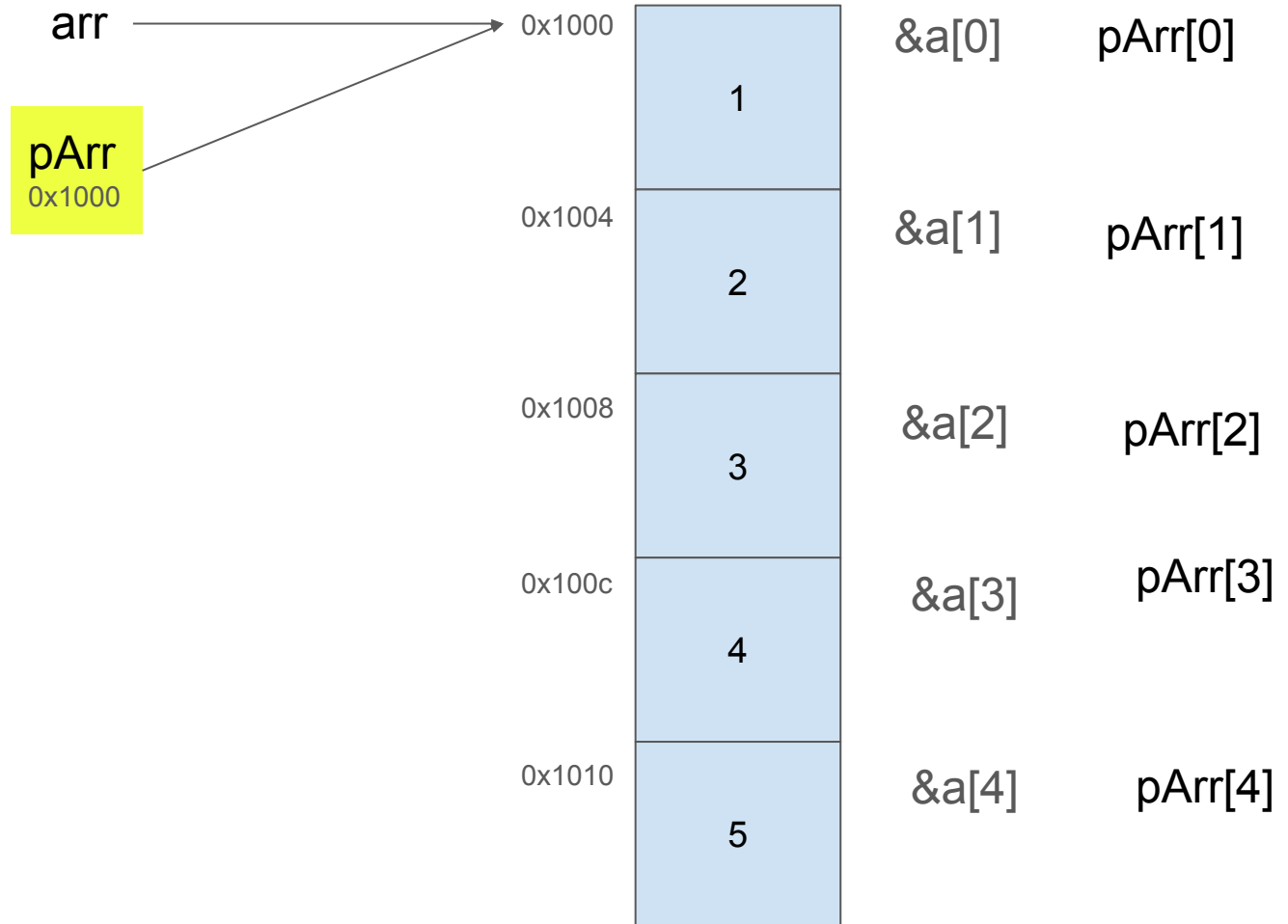
array[0]

array[1]

array[2]

array[3]

array[4]



`arr`

`0x1000`

`&a[0]`

`pArr[0]`

1

`0x1004`

`&a[1]`

`pArr[1]`

2

`0x1008`

`&a[2]`

`pArr[2]`

3

`0x100c`

`&a[3]`

`pArr[3]`

4

`0x1010`

`&a[4]`

`pArr[4]`

5

`pArr`

`0x1000`

`int arr[5] = {1,2,3,4,5}`

`int *pArr = arr ;`

```
int a[5] = {1,2,3,4,5}
```

```
int *p = a ;
```

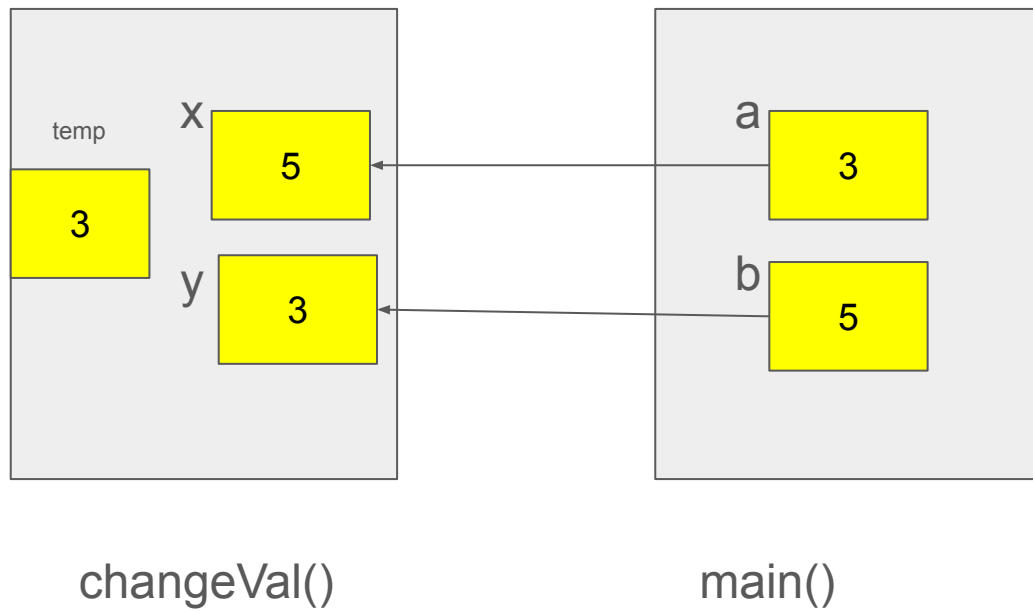
요소값 표현

p[0]	*p	*a
p[1]	*(p+1)	*(a+1)
p[2]	*(p+2)	*(a+2)
p[3]	*(p+3)	*(a+3)
p[4]	*(p+4)	*(a+4)

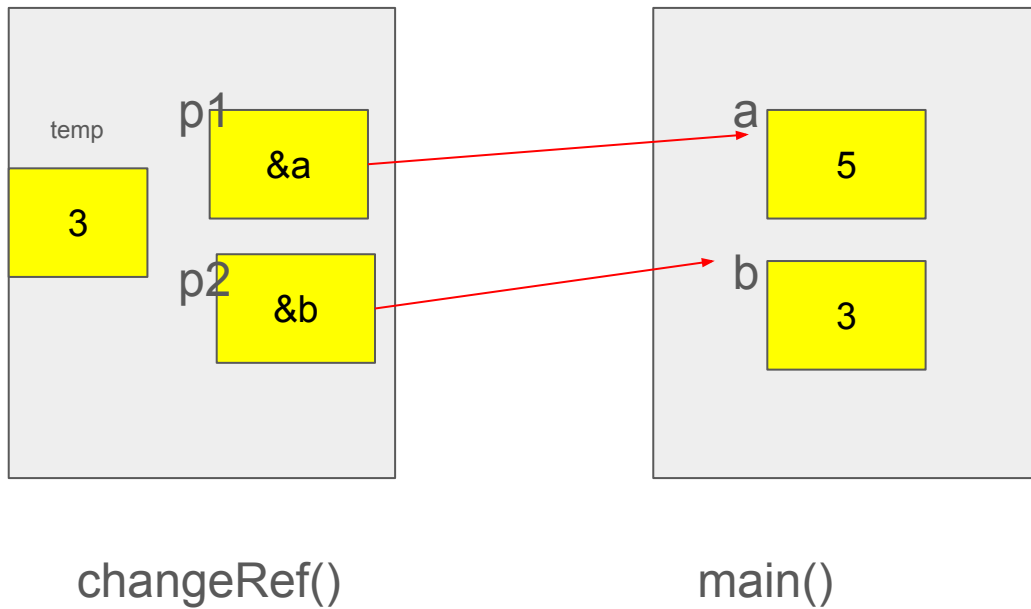
주소값 표현

&a[0]	a	p
&a[1]	a+1	p+1
&a[2]	a+2	p+2
&a[3]	a+3	p+3
&a[4]	a+4	p+4

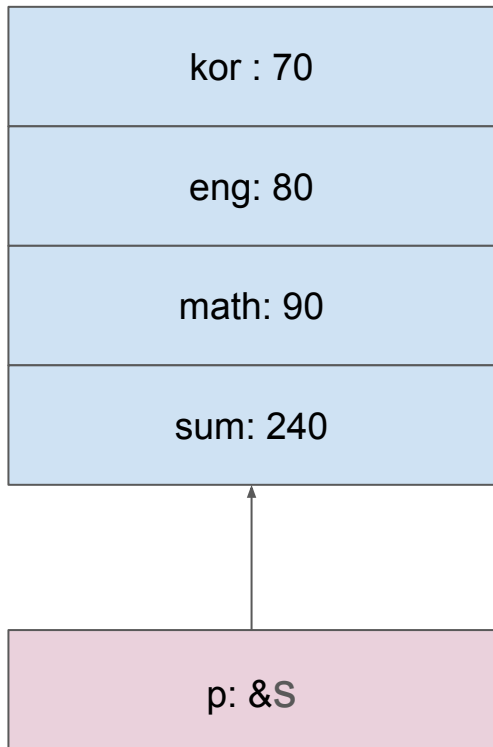
Call by Value (값 전달)



Call by Address (주소 전달)



구조체 변수 s



구조체 멤버 접근 방법

- 1) `s.sum` (직접 접근)
- 2) `p->sum` (화살표 연산자)
- 3) `(*p).sum` (역참조 후 점 연산자)

C++

&value: 0x1000

value
값 : 3 -> 10 -> 20

ptr
값 : &value

ref
(참조)

