

포인터

Ex)

선언 : 자료형 * 변수명;
 $(int * pInt;)$ \rightarrow $int i = 3$
 $int * p = \underline{\&i};$ i의 주소값
P: 주소값 ($0x\sim$)
*P: 내용 (3)
"포인터 변수는 메모리의 위치"

$int a = 10;$
 $int * pInt = \&a;$

%p [a의 주소값: ($0x\sim$) $\&a = pInt$, $pInt$ 의 주소값 = $*pInt$]
%d [a의 값: $a = *pInt$]
10

"포인터 연산"

$int a = 10;$
 $int * pInt = \&a;$
 $\underline{*pInt} = a + 2; \rightarrow a = 12.$

포인더 Swap

```
void Swap(int a, int b)
```

```
{  
    temp=0;  
    temp = a;  
    a = b;  
    b = temp;
```

→ "a가 b가.
 안바뀜"

```
void Swap(int *a, int *b);
```

```
int main(void)  
{
```

①

② Swap(x, y)

③ Swap(&x, &y)

```
void Swap(int *a, int *b);
```

```
{
```

```
    int = temp;  
    temp = a; *a
```

② *a = b;
 *b = temp;

① 함수 포인터로 전송
→ 멤버 이용.

a와 b의 값이

안바뀜.

Why?

堆疊영역에서 차례로 순서
조작하기 때문?

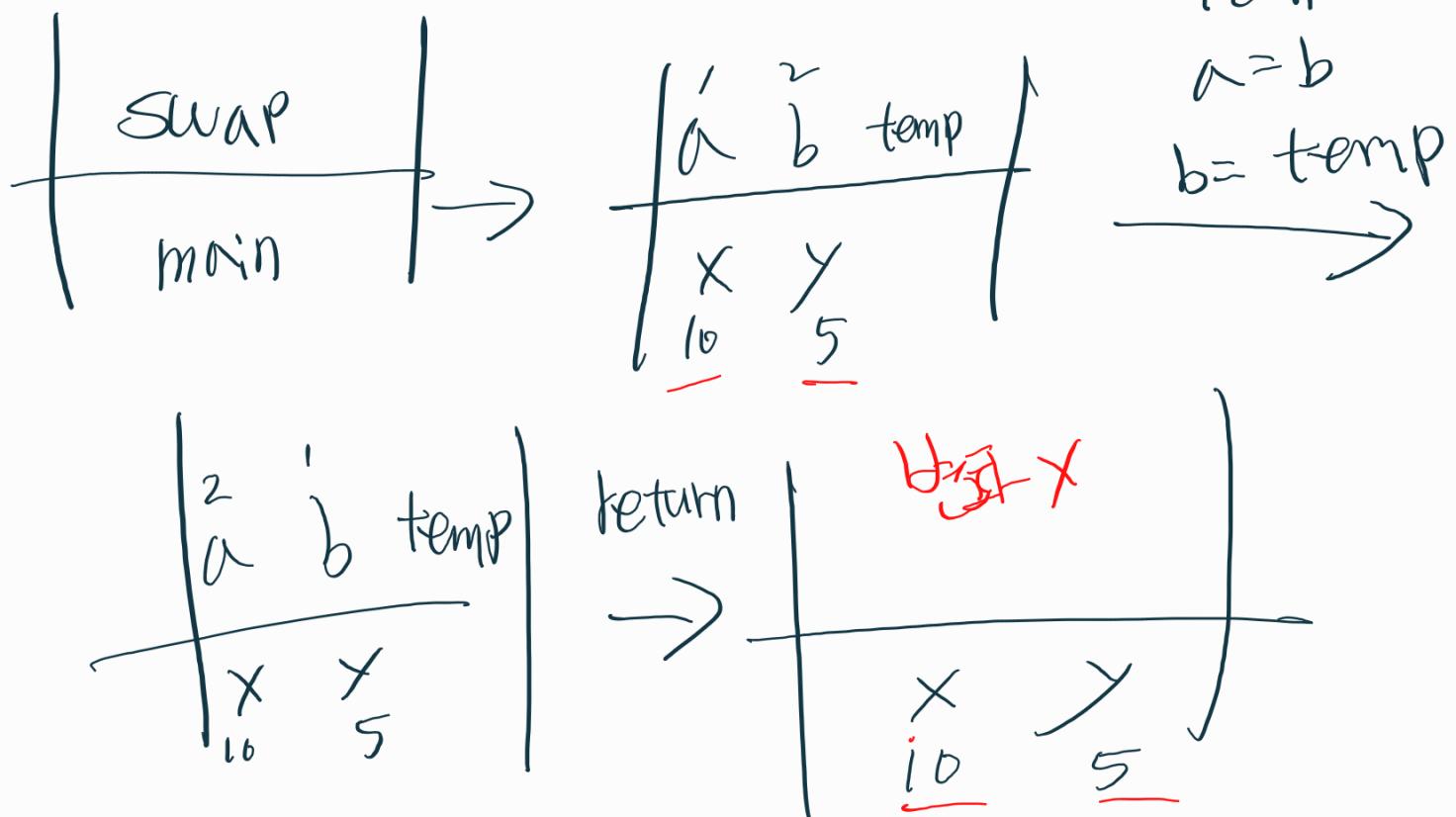
"stack" 구조

가장 최근에 들어온 것을
먼저 빼는 구조

② 포인터로 바꿨으니 *a, *b로
바꿔야 함
→ 멤버 이용.

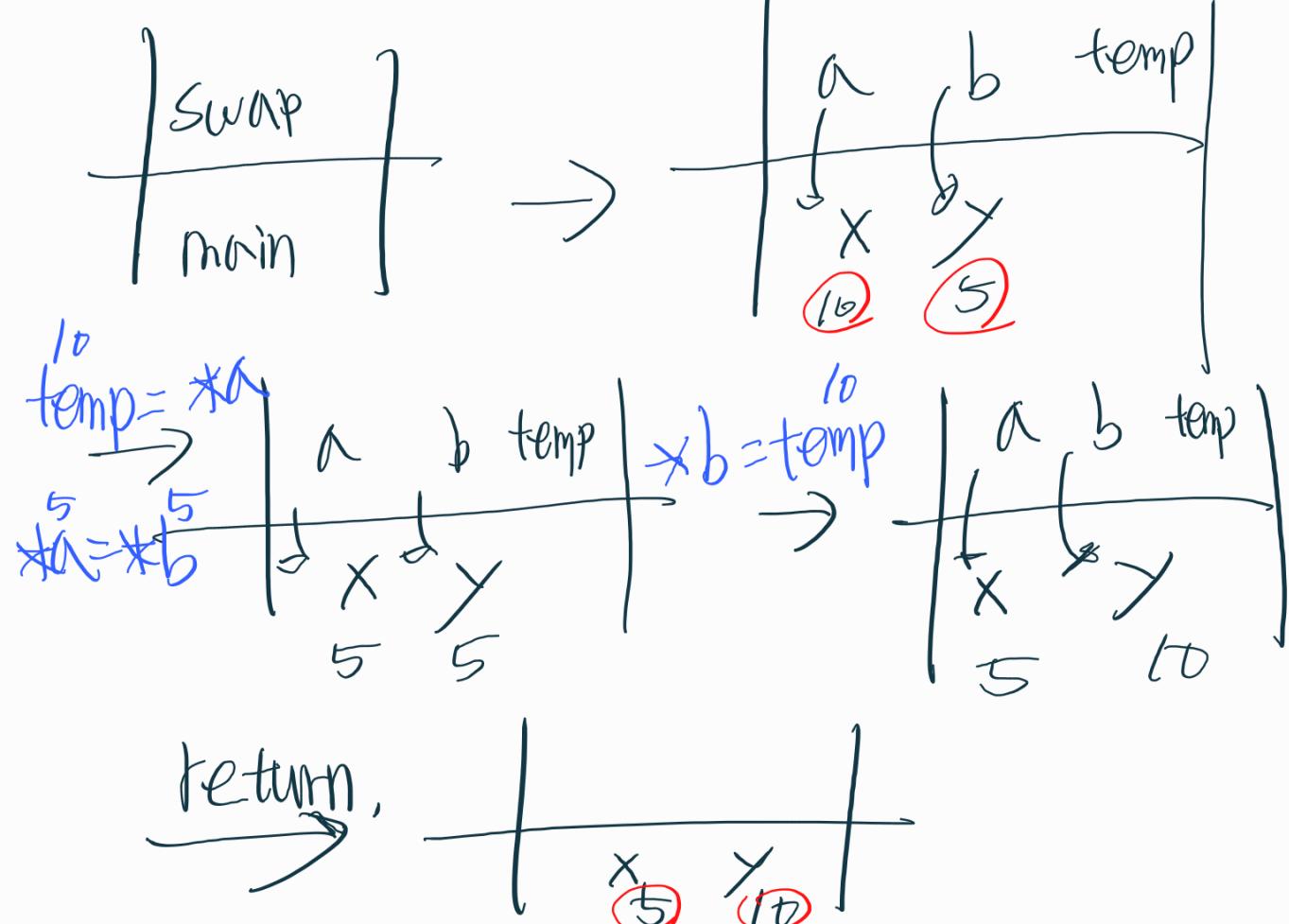
③ 주소값을 받아와야 하니
&x, &y로 반환
→ 사용

function 할당은 값과 주소



→ 표현식 사용 시 ("call by reference")

25번



포인터와 배열

■ 배열의 주소를 표현하는 방법

표 10-1 배열 a의 주소를 표현하는 방법

구분	첫 번째 주소	두 번째 주소	세 번째 주소	네 번째 주소
방법 1	$\&a[0]$	$\&a[1]$	$\&a[2]$	$\&a[3]$
방법 2	a	a+1	a+2	a+3
방법 3	p	p+1	p+2	p+3

■ 배열의 값을 표현하는 방법

표 10-2 배열 a의 값을 표현하는 방법

구분	첫 번째 값	두 번째 값	세 번째 값	네 번째 값
방법 1	$a[0]$	$a[1]$	$a[2]$	$a[3]$
방법 2	$*a$	$*(a+1)$	$*(a+2)$	$*(a+3)$
방법 3	$*p$	$*(p+1)$	$*(p+2)$	$*(p+3)$
방법 4	$p[0]$	$p[1]$	$p[2]$	$p[3]$

주소: $\&a[0] = a = p$

값: $a[0] = *a = *p = p[0]$

“배열 블리오기”

배열 전체 VS 포인터 블리오기

함수 선언: int Array(int Array[], int size) | int Array(int *pA, int size)

함수 정의: X = Array(a, 배열의 개수)

행수 정의: Array[] | pA[]

