

C언어 스터디

4주차 — 4/13

오늘의 할 일

- 포인터
- 포인터
- 포인터
- 포인터
- 포인터
- 포인터

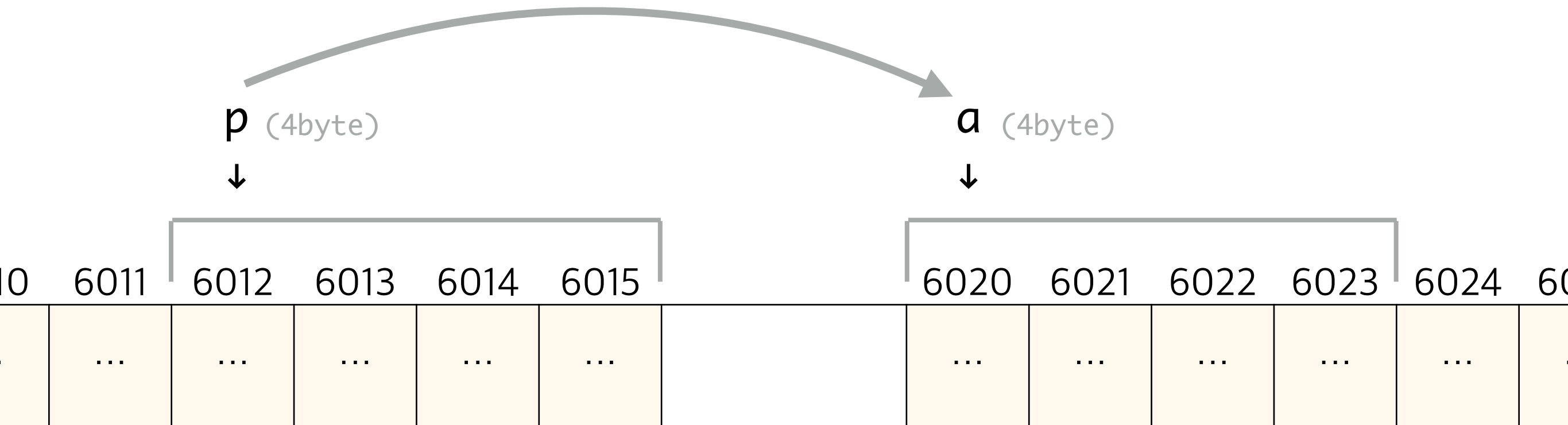
I. 포인터

포인터

- 변수의 주소를 갖고 있는 변수

```
int a = 10, b = 20;  
int *p;      // 포인터 p 선언  
  
p = &a;      // p는 a를 가리킴
```

메모리와 주소



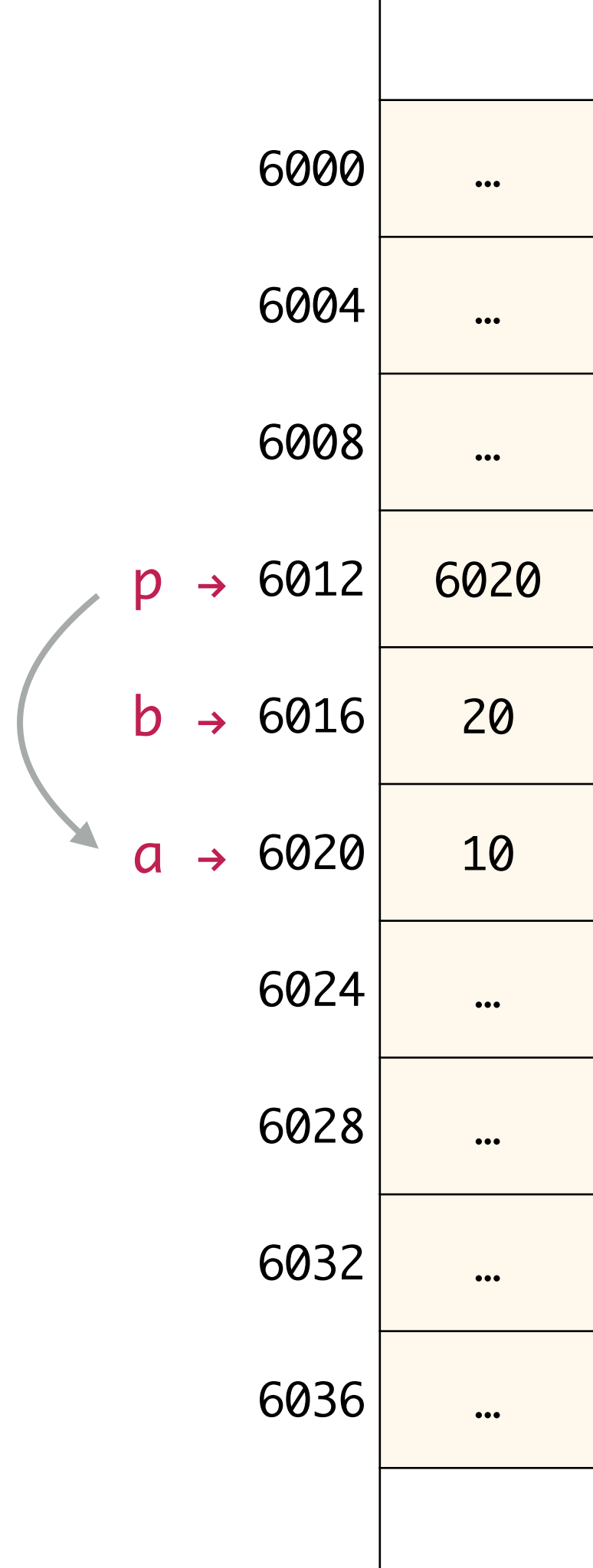
연산자 &

대상의 주소를 구함.

```
int a = 10, b = 20;  
int *p;      // 포인터 p 선언  
  
p = &a;      // p는 a를 가리킴
```

```
printf("%d", p);
```

6020



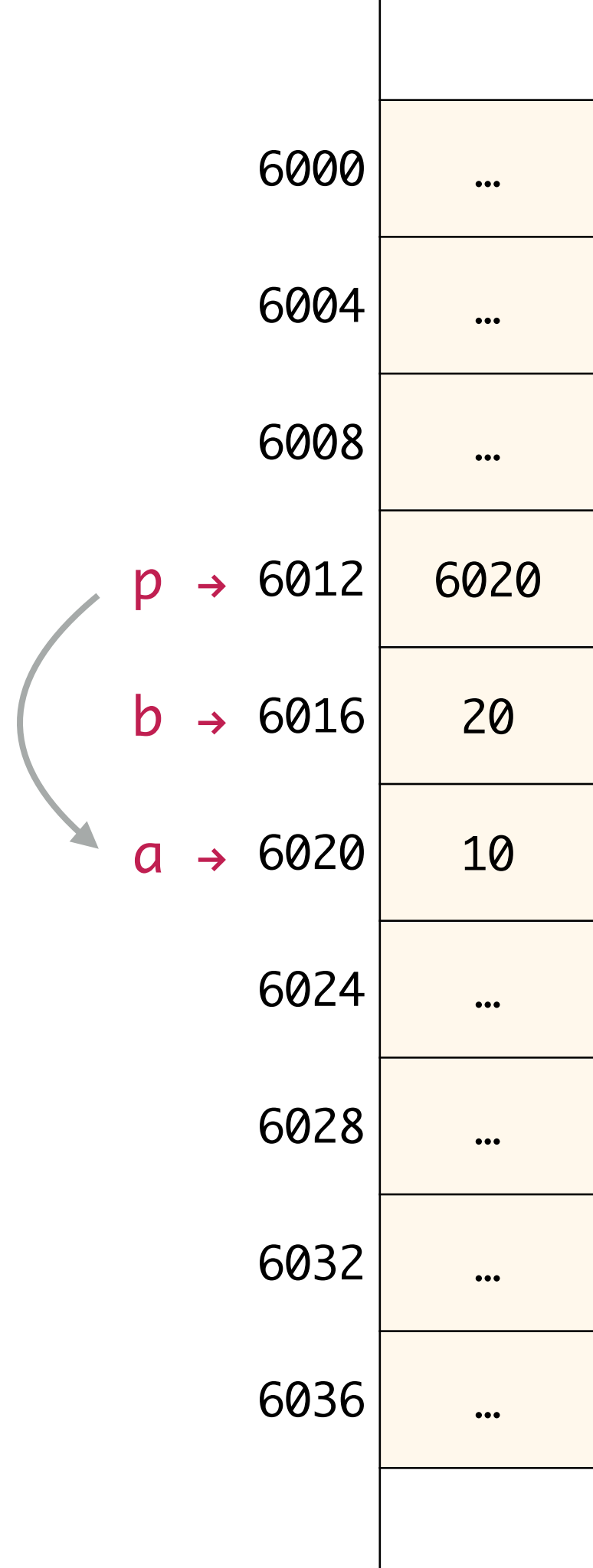
연산자 *

포인터가 가리키는 대상에 접근.

```
int a = 10, b = 20;  
int *p;      // 포인터 p 선언  
  
p = &a;      // p는 a를 가리킴
```

```
printf("%d", *p);
```

10



Q. 다음 프로그램의 실행 결과는?

```
int a = 10, b = 20;  
int *p;  
  
p = &a;  
*p += 5;  
printf("%d\n", a);
```


포인터 선언

```
int *p;
```

p의 타입

(int *)

*p의 타입

int

— "p가 가리키는 변수의 타입이 int"

```
int *p;
```

```
int* p;
```

=> 둘 다 가능하다.

포인터 선언

```
int *a, b;
```

a의 타입

(int *)

b의 타입

int

*a의 타입

int

*b의 타입

```
int *a, *b;
```

a의 타입

(int *)

b의 타입

(int *)

*a의 타입

int

*b의 타입

int

swap 함수 만들기

```
void swap(int *v1, int *v2) {  
    int temp = *v1;  
    *v1 = *v2;  
    *v2 = temp;  
}
```

```
int main() {  
    int a = 10, b = 20;    int *p = &a;  
  
    swap(_____, _____);  
    printf("%d %d\n", a, b);  
    return 0;  
}
```

20 10

swap 함수 만들기

```
void swap(int *v1, int *v2) {  
    int temp = *v1;  
    *v1 = *v2;  
    *v2 = temp;  
}  
  
int main() {  
    int a = 10, b = 20;    int *p = &a;  
  
    swap(&a, &b);          // swap(p, &b); 도 가능  
    printf("%d %d\n", a, b);  
    return 0;  
}
```

20 10

swap 함수 만들기

```
void swap(int *v1, int *v2) {  
    int temp = *v1;  
    *v1 = *v2;  
    *v2 = temp;  
}
```

```
int main() {  
    int a = 10, b = 20;  
    int *p = &a;  
  
    swap(&a, &b);  
    printf("%d %d\n", a, b);  
    return 0;  
}
```

	3538	
	3542	
	3546	
temp →	3550	
v2 →	3554	
v1 →	3558	
p →	7580	
b →	7584	
a →	7588	

II. 포인터와 배열

포인터와 배열

주소 확인해보기

```
int arr[5] = {0,};  
int i, *p = arr;  
  
printf("arr: %d\n", arr);  
for (i = 0; i < 5; i++)  
    printf("arr[%d]: %d\n", i, &arr[i]);
```

- 배열 변수의 값은 0번째 원소의 주소

3132	...	
3136	...	
p → 3140		
i → 3144		
3148	...	
arr[0]→ 3152		
arr[1]→ 3156		
arr[2]→ 3160		
arr[3]→ 3164		
arr[4]→ 3168		

포인터와 배열

3132

...

3136

...

$p \rightarrow 3140$

$i \rightarrow 3144$

3148

...

$arr[0] \rightarrow 3152$

$\leftarrow p$

$arr[1] \rightarrow 3156$

$\leftarrow p + 1$

$arr[2] \rightarrow 3160$

$\leftarrow p + 2$

$arr[3] \rightarrow 3164$

$\leftarrow p + 3$

$arr[4] \rightarrow 3168$

$\leftarrow p + 4$

포인터와 배열

```
int arr[5] = {2, 3, 5, 7, 9};  
int i;  
  
for (i = 0; i < 5; i++)  
    printf("%d %d\n", arr[i], *(arr + i));
```

`*(arr + i)`

`arr[i]`

=> 같다.

strlen 함수 만들기

```
int strlen(char *s) {  
    int n;  
    for (n = 0; *s != '\0'; s++)  
        n++;  
    return n;  
}
```

```
int main() {  
    char str[128]; int res;  
    scanf("%s", str);  
  
    res = strlen(str);  
    printf("length: %d\n", res);  
    return 0;  
}
```

포인터와 2차원 배열


```
int arr[5][5];  
int i, j;  
for (i = 0; i < 5; i++)  
    for (j = 0; j < 5; j++)  
        arr[i][j] = i * 5 + j;
```

arr[0]	0	1	2	3	4
arr[1]	5	6	7	8	9
arr[2]	10	11	12	13	14
arr[3]	15	16	17	18	19
arr[4]	20	21	22	23	24

포인터와 2차원 배열

```
for (i = 0; i < 5; i++) {  
    for (j = 0; j < 5; j++)  
        printf("%2d ", *(*(arr + i) + j));  
    printf("\n");  
}
```

(int *) 타입



arr[0]	0	1	2	3	4
arr[1]	5	6	7	8	9
arr[2]	10	11	12	13	14
arr[3]	15	16	17	18	19
arr[4]	20	21	22	23	24

- `int *pi`로 받을 수 있다.
- +1 하면 `sizeof(int)` 만큼 증가한다.

포인터와 2차원 배열

```
for (i = 0; i < 5; i++) {  
    int *pb = arr[i];  
    for (j = 0; j < 5; j++) {  
        printf("%2d ", *pb);  
        pb++;  
    }  
    printf("\n");  
}
```

포인터와 2차원 배열

```
for (i = 0; i < 5; i++) {  
    for (j = 0; j < 5; j++)  
        printf("%2d ", *(*(arr + i) + j));  
    printf("\n");  
}
```

타입

arr[0]	0	1	2	3	4
arr[1]	5	6	7	8	9
arr[2]	10	11	12	13	14
arr[3]	15	16	17	18	19
arr[4]	20	21	22	23	24

- _____로 받을 수 있다.
- +1 하면 sizeof(int)*5 만큼 증가한다.

포인터와 2차원 배열

```
for (i = 0; i < 5; i++) {  
    for (j = 0; j < 5; j++)  
        printf("%2d ", *(*(arr + i) + j));  
    printf("\n");  
}
```

`int (*)[5]` 타입

arr[0]	0	1	2	3	4
arr[1]	5	6	7	8	9
arr[2]	10	11	12	13	14
arr[3]	15	16	17	18	19
arr[4]	20	21	22	23	24

- `int (*p)[5]`로 받을 수 있다.
- +1 하면 `sizeof(int)*5` 만큼 증가한다.

배열 포인터

```
int (*pa)[5] = arr;
for (i = 0; i < 5; i++) {
    int *pb = *pa;
    for (j = 0; j < 5; j++) {
        printf("%2d ", *pb);
        pb++;
    }
    printf("\n");
    pa++;
}
```

→ 이런 친구들을 '배열 포인터'라고 합니다.

배열을 가리키는 포인터라고 생각하면 될 듯

포인터 배열

```
int a = 20, b = 10, c = 30, i;  
int *pArr[3] = {&a, &b, &c};  
for (i = 0; i < 3; i++)  
    printf("%d ", *pArr[i]);
```

→ '포인터 배열'

배열의 원소가 포인터

끝.

시험 잘보세요!