고급C프로그래밍

01 C프로그래밍 Review

01
메모리와 주소

중간고사: 1/2반 공통 실시하며 10/28(토) 13:00 시행합니다. 기말고사: 1/2반 공통 실시하며 12/16(토) 13:00 시행합니다.

메모리와 주소

- **메모리는 바이트(Byte) 단위로 나뉘며, 각 바이트에는 주소가 지정**
- **정수형 변수의 메모리 할당**
  - ✓ 정수형 변수의 크기가 4바이트
  - ➔ 메모리에 정수형 변수 a를 선언하면 임의의 위치에 4바이트가 할당
- **변수가 위치하는 곳 : 변수가 할당 받은 메모리의 시작 주소**
  - ✓ 변수의 주소를 알려면 변수 앞에 '&'를 붙임
  - ✓ a의 주소(&a) = 1036번지, b의 주소(&b) = 1048번지

```
int a = 100;
int b = 200;
```

그림 9-6 메모리에 할당된 정수형 변수의 위치 예

02
배열

중간고사: 1/2반 공통 실시하며 10/28(토) 13:00 시행합니다. 기말고사: 1/2반 공통 실시하며 12/16(토) 13:00 시행합니다.

배열 (1/3)

- **배열의 개념 : 여러 개의 변수를 나란히 연결하는 개념**
  - ✓ 박스(변수)를 한 줄로 붙이고, 박스의 이름(aa)을 지정
  - ✓ 각각의 박스는 aa[0], aa[1], … 과 같이 첨자를 붙임

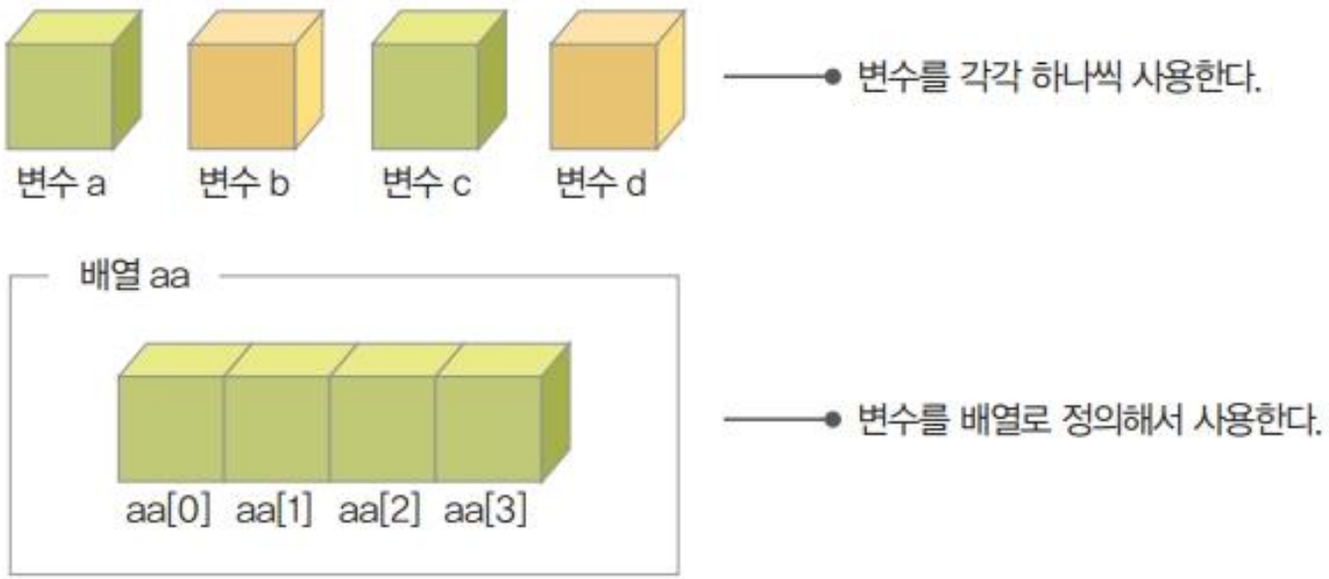

그림 8-1 배열의 개념

중간고사: 1/2반 공통 실시하며 10/28(토) 13:00 시행합니다. 기말고사: 1/2반 공통 실시하며 12/16(토) 13:00 시행합니다.

배열 (2/3)

- **2차원 배열 : 1차원 배열들의 배열**
  - ✓ 1차원 배열의 각 항목에 또 다른 1차원 배열을 넣는 구조

int aa[행][열]

그림 8-16 2차원 배열의 개념

중간고사: 1/2반 공통 실시하며 10/28(토) 13:00 시행합니다. 기말고사: 1/2반 공통 실시하며 12/16(토) 13:00 시행합니다.

배열 (3/3)

- **n차원 배열 : (n-1)차원 배열들의 배열**
- **3차원 배열**
  - ✓ 1차원 배열의 각 항목에 2차원 배열을 저장하는 방식

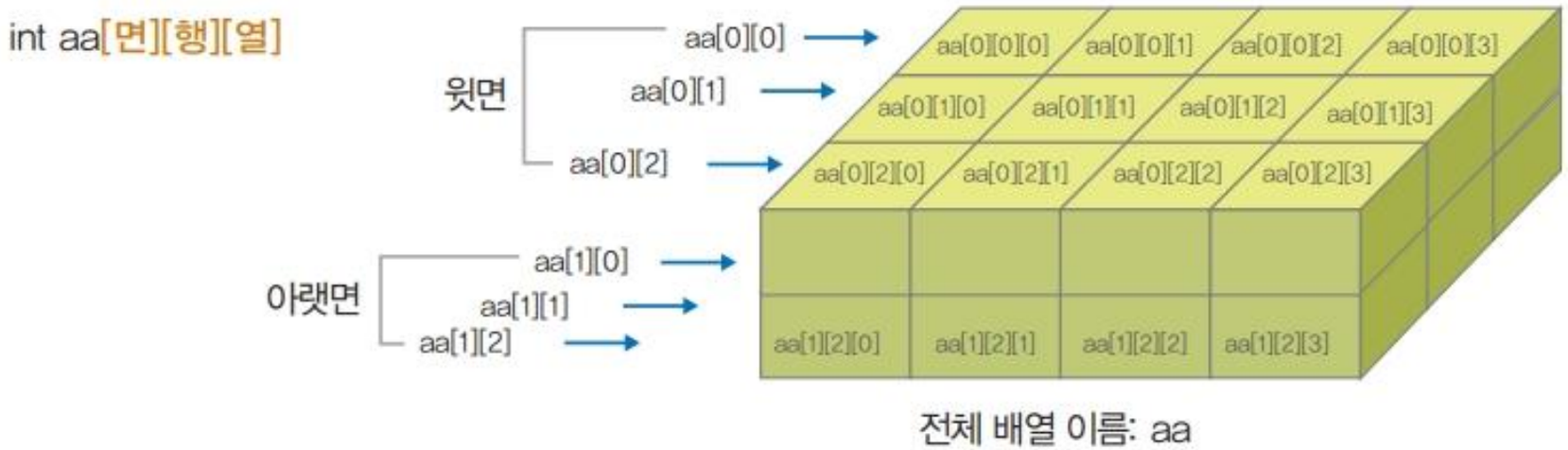

그림 8-18 3차원 배열의 개념

03
포인터

중간고사: 1/2반 공통 실시하며 10/28(토) 13:00 시행합니다. 기말고사: 1/2반 공통 실시하며 12/16(토) 13:00 시행합니다.

포인터 (1/4)

- **포인터란 주소를 담는 그릇(변수)**

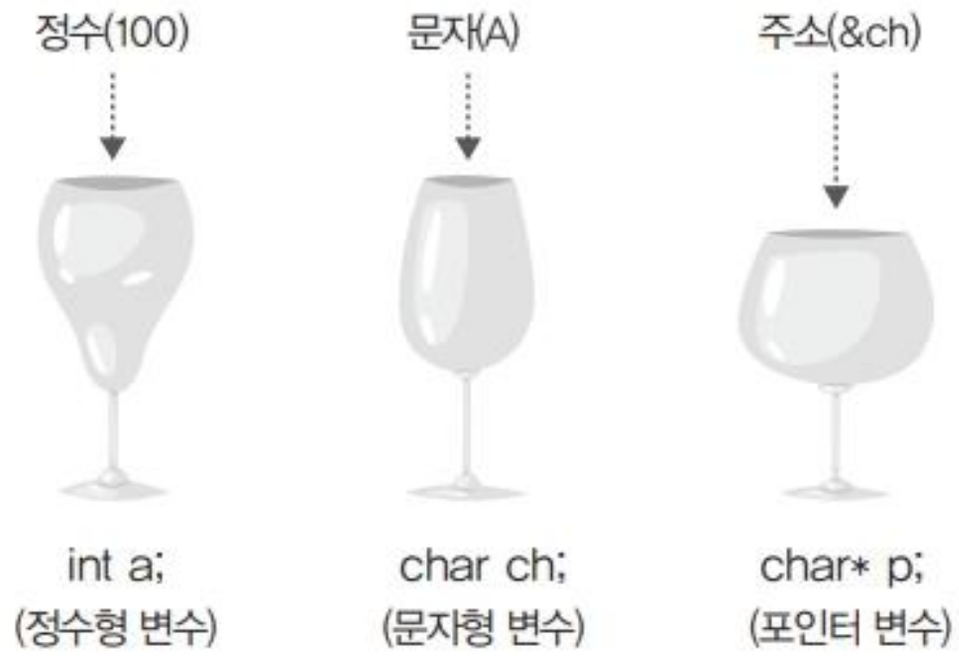

그림 9-9 변수의 종류

포인터 (2/4)

- **포인터 사용 예제**

```
❶ char ch;
❷ char* p;
❸ ch = 'A'
❹ p = &ch;
```

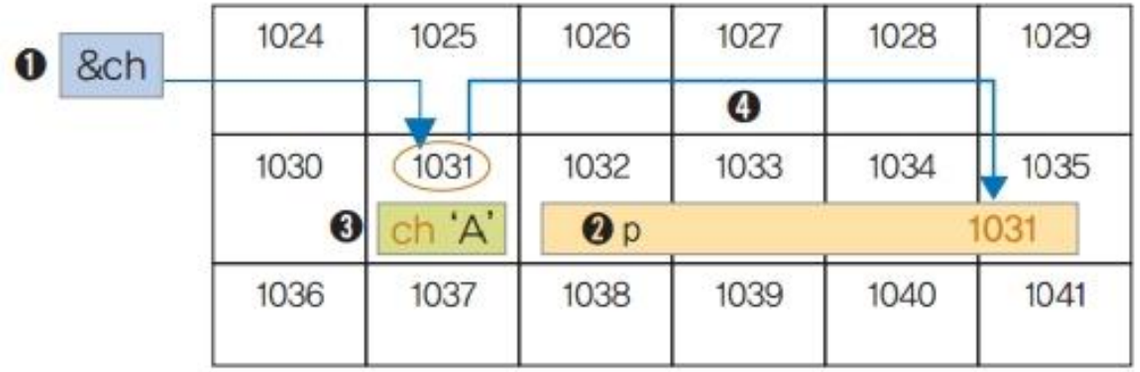

그림 9-10 일반 변수와 포인터 변수의 관계 예

❶의 문자형 변수 ch는 1바이트를 차지하므로 주소 1031번지에 1바이트가 자리 잡음 (&ch 는 1031을 뜻하는 주소의 값)

❷의 포인터 변수(char*) p는 1032~1035번지에 4바이트가 자리 잡음 (포인터 변수는 크기가 4바이트)

❸의 변수 ch에 'A 값'을 넣고 ❹의 포인터 변수 p에 변수 ch의 주솟값인 &ch를 넣음 ➔ &ch는 1031번지를 의미하므로 포인터 변수 p에는 1031이 들어감

중간고사: 1/2반 공통 실시하며 10/28(토) 13:00 시행합니다. 기말고사: 1/2반 공통 실시하며 12/16(토) 13:00 시행합니다.

포인터 (3/4)

- **주소와 포인터 정리**
  - ✓ int a; //정수형 값을 저장하는 변수 a
  - ✓ int *p; //정수형 값을 저장하는 메모리 공간의 주소를 저장하는 변수 p
  - ✓ 정수형 값을 저장하는 메모리 공간의 주소 대입 : p = &a;
  - ✓ 특정 주소에 저장되어 있는 값 : *(&a) == *p == a

중간고사: 1/2반 공통 실시하며 10/28(토) 13:00 시행합니다. 기말고사: 1/2반 공통 실시하며 12/16(토) 13:00 시행합니다.

포인터 (4/4)

- **주소와 포인터, 배열 정리**
  - ✓ char s[10]; //문자의 집합인 문자열을 저장하는 배열 s
  - ✓ char *p; //문자 값을 저장하는 메모리 공간의 주소를 저장하는 변수 p
  - ✓ 문자형 값을 저장하는 메모리 공간의 주소 대입 : p = s; or p = &s[0];
  - ✓ 배열의 첫번째 칸에 저장되어 있는 값 : *(&s[0]) == *p == *(s+0) == s[0]
  - ✓ 배열의 두번째 칸에 저장되어 있는 값 : *(&s[1]) == *(p+1) == s[1]

04
함수

중간고사: 1/2반 공통 실시하며 10/28(토) 13:00 시행합니다. 기말고사: 1/2반 공통 실시하며 12/16(토) 13:00 시행합니다.

함수 (1/5)

- **함수의 기본 형태**

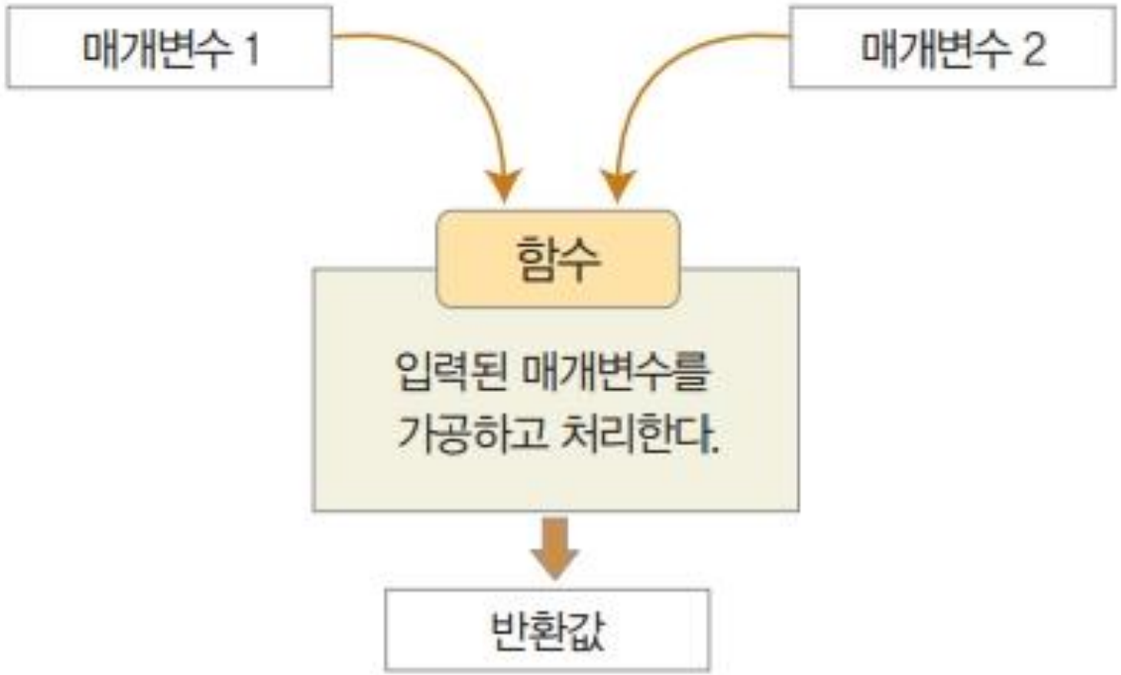

그림 10-3 함수의 형태

중간고사: 1/2반 공통 실시하며 10/28(토) 13:00 시행합니다. 기말고사: 1/2반 공통 실시하며 12/16(토) 13:00 시행합니다.

함수 (2/5)

- **지역변수와 전역변수**
  - ✓ 지역변수 : 한정된 지역(local)에서만 사용되는 변수
  - ✓ 전역변수 : 프로그램 전체(global)에서 사용되는 변수

그림 10-6 지역변수와 전역변수의 생존 범위

중간고사: 1/2반 공통 실시하며 10/28(토) 13:00 시행합니다. 기말고사: 1/2반 공통 실시하며 12/16(토) 13:00 시행합니다.

함수 (3/5)

- **반환값이 있는 함수**
  - ✓ 함수를 실행한 후에 나온 결과값은 함수의 데이터형을 따름

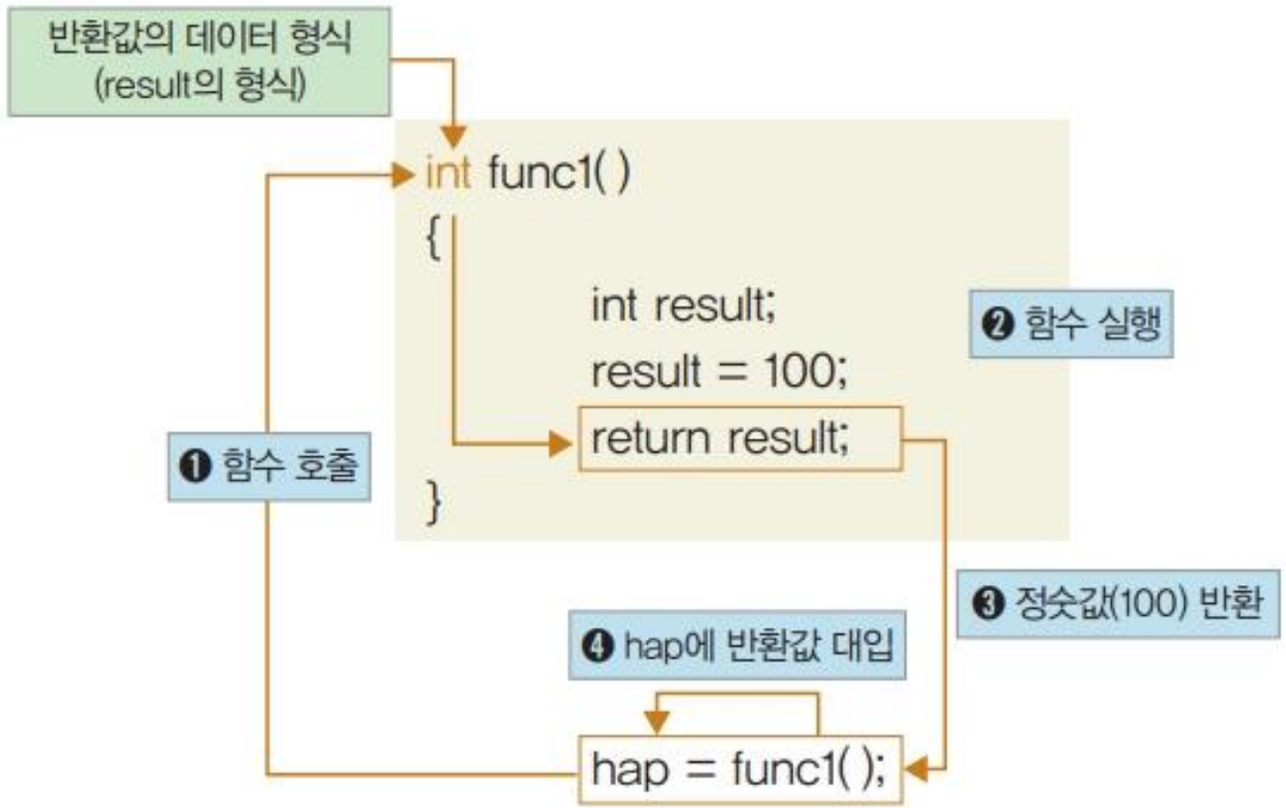

그림 10-8 int 형 값의 반환

중간고사: 1/2반 공통 실시하며 10/28(토) 13:00 시행합니다. 기말고사: 1/2반 공통 실시하며 12/16(토) 13:00 시행합니다.

함수 (4/5)

- **반환값이 없는 함수**
  - ✓ 함수를 실행한 결과, 돌려줄 것이 없는 경우
  - ✓ void로 함수 표시 : void 형 함수를 호출할 때는 함수 이름만 표시

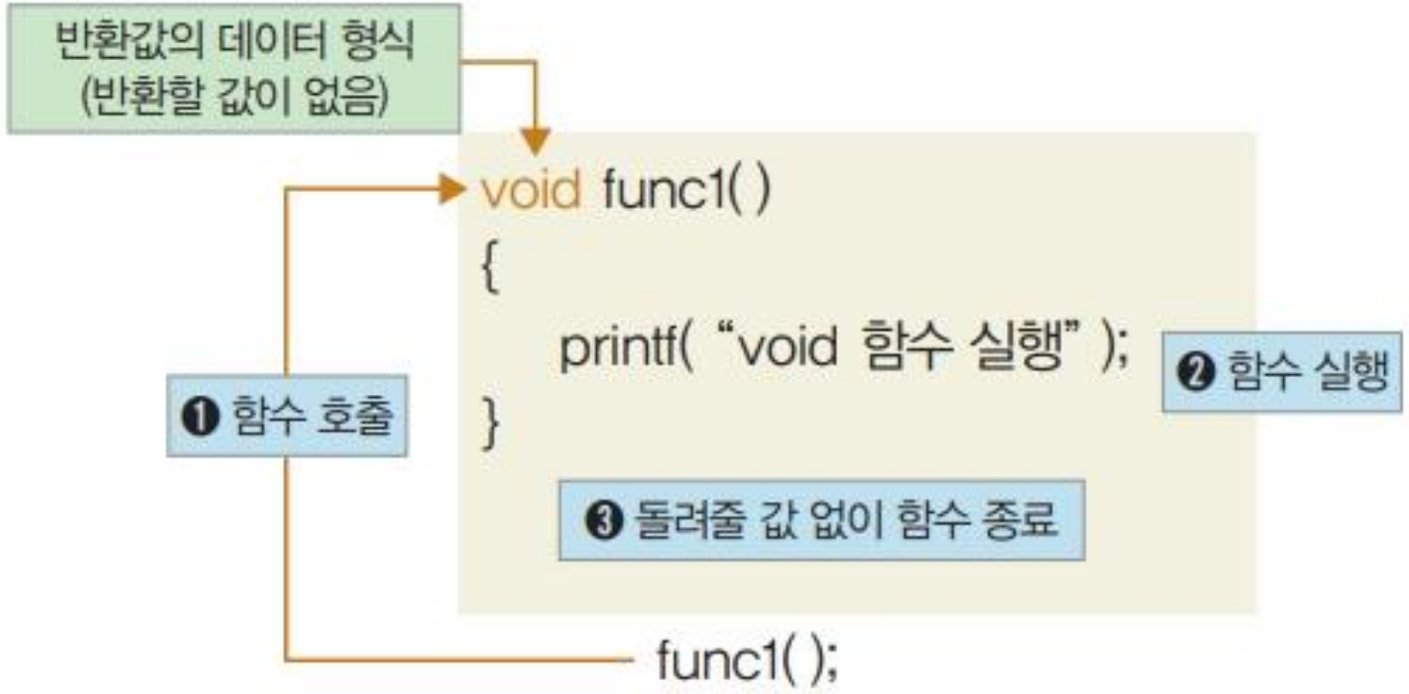

그림 10-9 void 형 함수의 작동

함수 (5/5)

- **매개변수 전달방법 : 값으로 전달(call by value)**
  - ✓ 메모리에 저장된 값을 복사하여 전달 ➔ 원본은 보존

```
void func1(int a){
    a = a +1; }

void main{
    int a = 10;
    func1(a);
    printf("%d", a); }
```

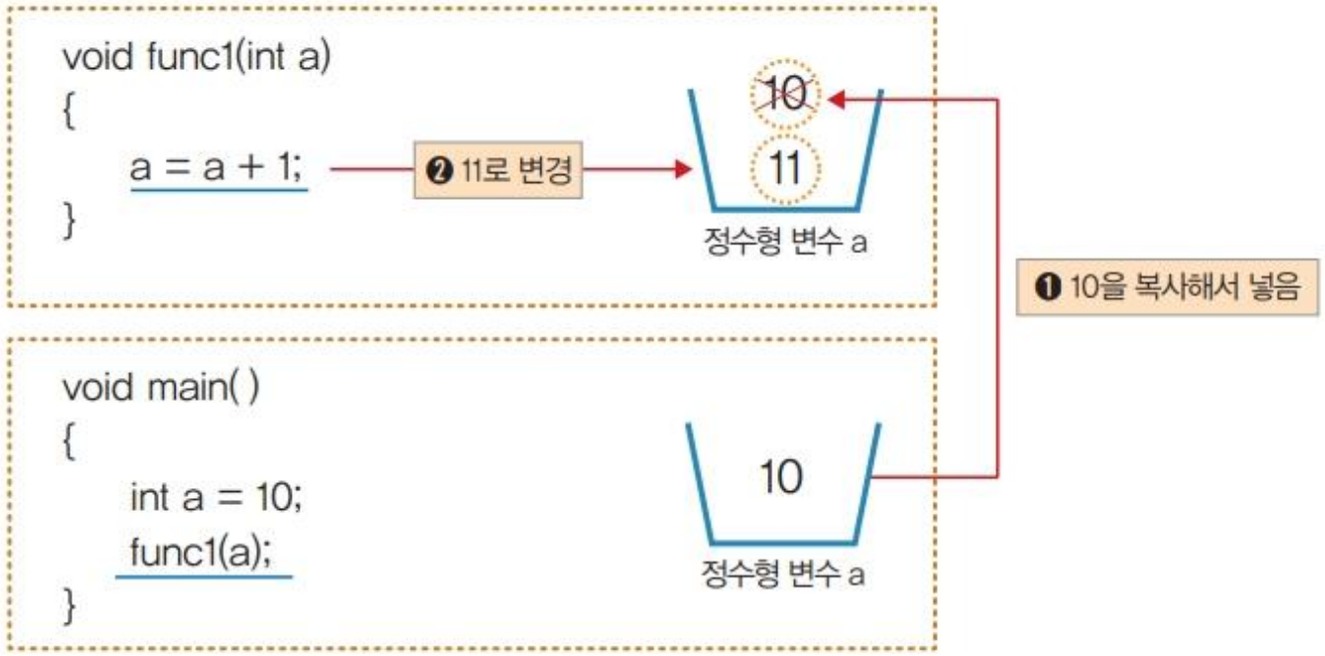

그림 10-10 매개변수 전달: 값으로 전달

  - ✓ 출력 결과는? 10

05
함수와 포인터

중간고사: 1/2반 공통 실시하며 10/28(토) 13:00 시행합니다. 기말고사: 1/2반 공통 실시하며 12/16(토) 13:00 시행합니다.

함수와 포인터 (1/4)

- **매개변수 전달방법 : 주소로 전달**
  - ✓ 값이 저장된 메모리의 주소를 전달 ➔ 원본의 변경이 발생

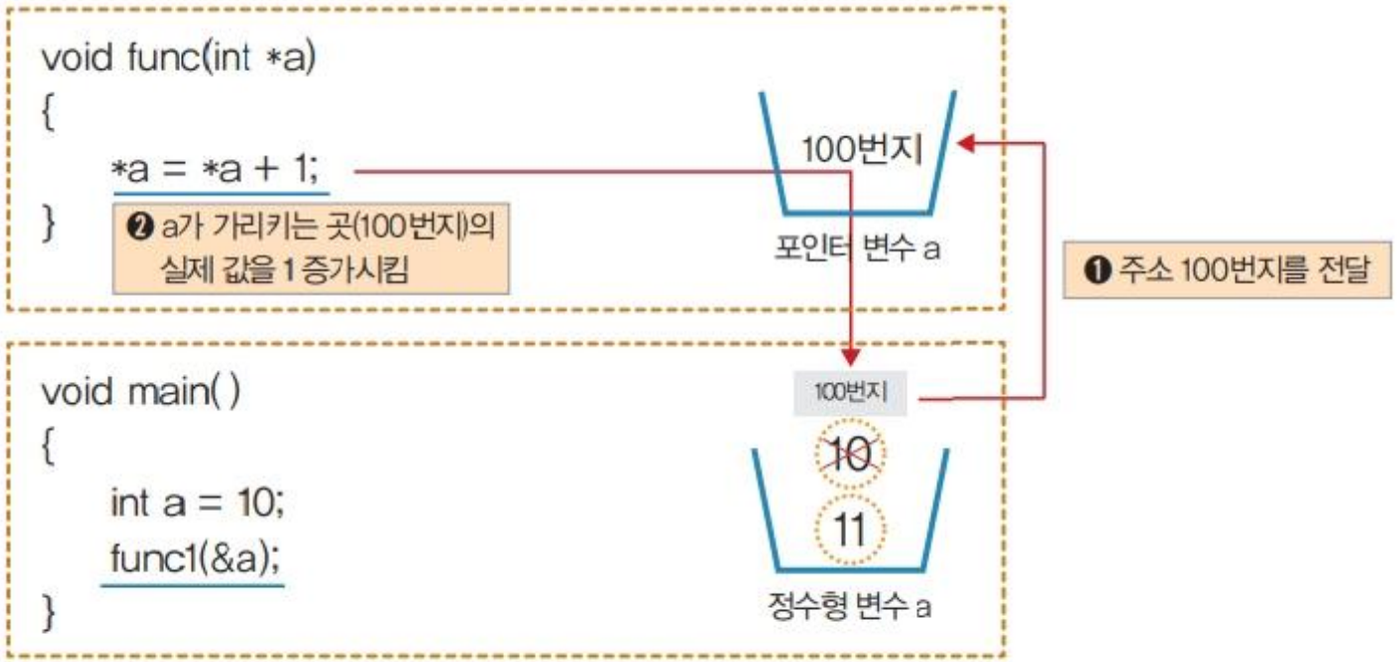

그림 10-11 매개변수 전달: 주소로 전달

함수와 포인터 (2/4)

- **값으로 전달 예제**

응용 10-10 매개변수 전달 방법 비교

값으로 전달하는 경우

```
01 #include <stdio.h>
02
03 void func1(char a, char b)          ----- 매개변수가 값인 함수이다.
04 {
05    int imsi;
06
07    imsi = a;
08    a = b;
09    b = imsi;
10 }
11
12 void func2(char *a, char *b)
13 {
14    int imsi;
15
16    imsi = *a;
17    *a = *b;
18    *b = imsi;
19 }
```

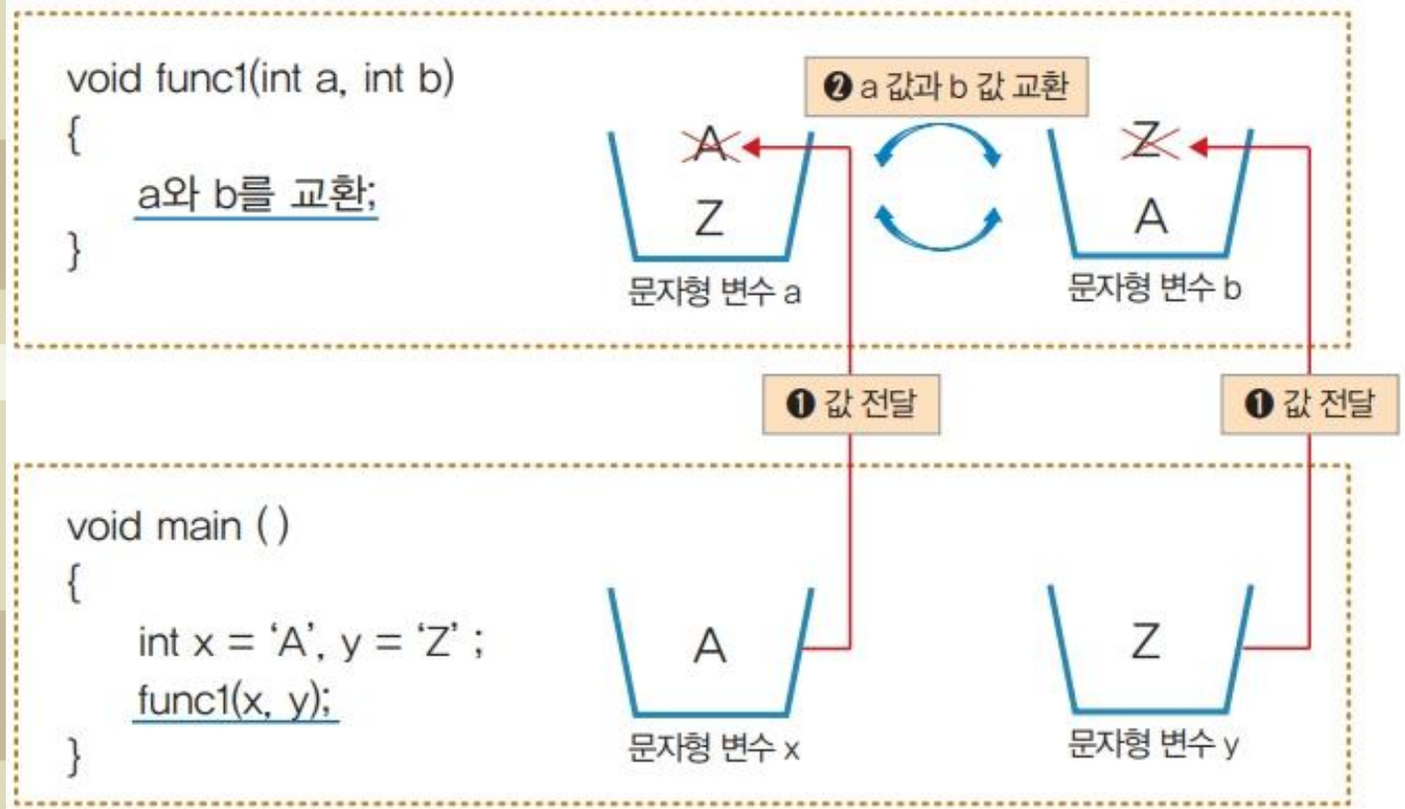

그림 10-12 값으로 전달을 통한 교환

함수와 포인터 (3/4)

- **주소로 전달 예제**

응용 10-10 매개변수 전달 방법 비교

```
01 #include <stdio.h>
02
03 void func1(char a, char b)
04 {
05    int imsi;
06
07    imsi = a;
08    a = b;
09    b = imsi;
10 }
11
12 void func2(char *a, char *b)
13 {
14    int imsi;
15
16    imsi = *a;
17    *a = *b;
18    *b = imsi;
19 }
```

주소로 전달하는 경우

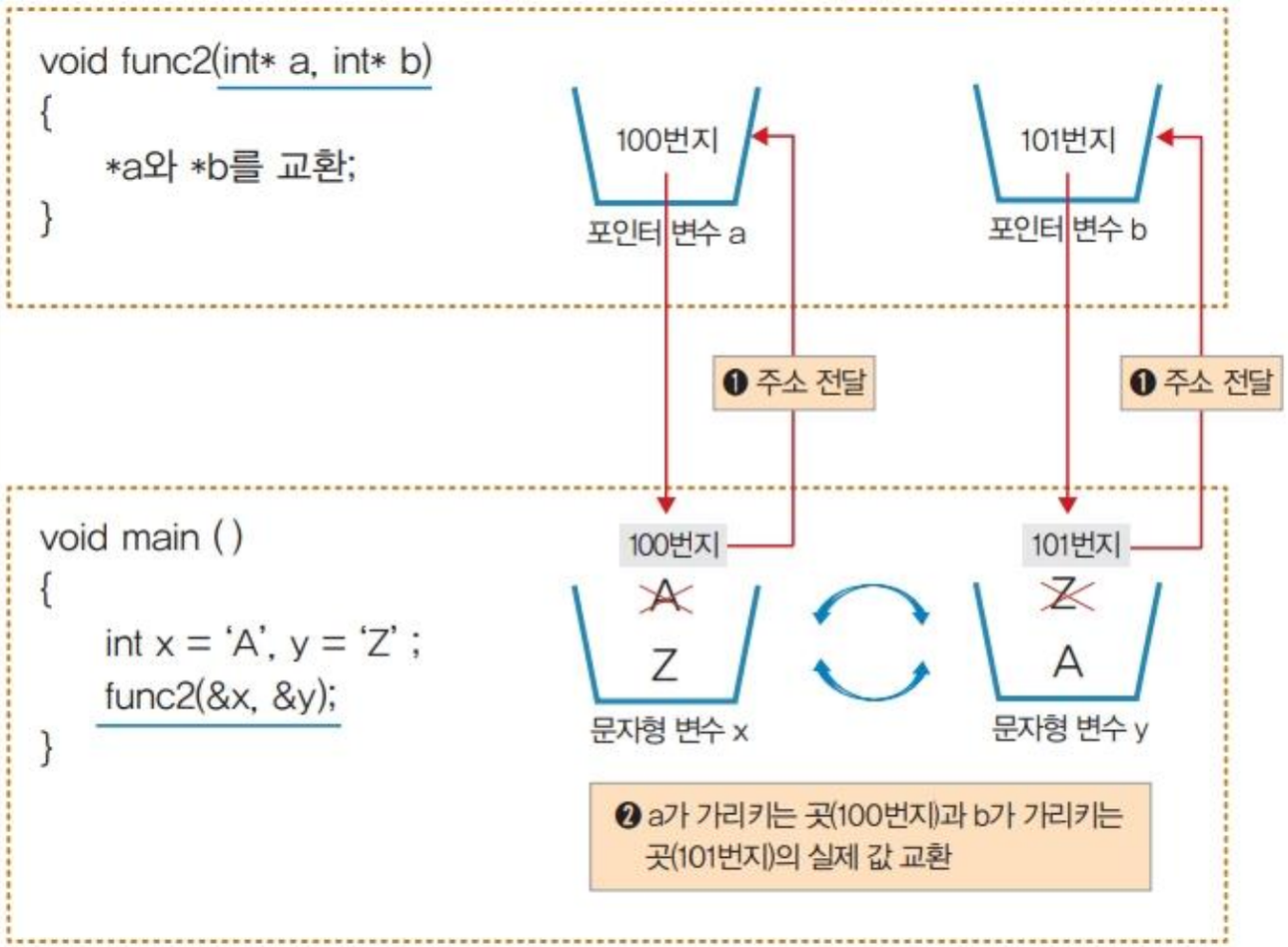

그림 10-13 주소로 전달을 통한 교환

함수와 포인터 (4/4)

- **실행 결과**

```
20
21 void main( )
22 {
23    char x = 'A', y = 'Z';
24
25    printf("원래 값        : x=%c, y=%c\n", x, y);        ---- 원래 문자를 출력한다.
26
27    func1(x, y);                                          ---- 값을 전달해서 func1( ) 함수를
28    printf("값을 전달한 후  : x=%c, y=%c\n", x, y);              호출한다.
29
30    func2(&x, &y);                                        ---- 주소를 전달해서 func2( ) 함수를
                                                                 호출한다.
31    printf("주소를 전달한 후: x=%c, y=%c\n", x, y);
32 }
```

정답_ 1 a = b; 2 *a = *b; 3 x, y 4 &x, &y

실행 결과

```
원래 값         : x=A, y=Z
값을 전달한 후  : x=A, y=Z
주소를 전달한 후: x=Z, y=A
```

Q & A