-Part3-

제1장 구조체와 공용체란 무엇인가

학습목차

1.1 구조체란

1.2 중첩 구조체

1.3 구조체와 배열

1.4 구조체와 포인터

1.5 구조체와 함수

1.6 공용체와 열거형

1.1 구조체란

1.1 구조체란

▶배울 내용

① 구조체 정의

② 구조체 변수

③ 구조체 변수로 멤버 변수에 접근하기

④ 구조체 변수의 초기화

⑤ 구조체 변수의 복사

1.1 구조체란 (1/18)

▶ 구조체

✓ 하나 이상의 변수를 묶어 **그룹화**하는 **사용자 정의 자료형**

그룹화(같은 자료형)

```
int a;
int b;
int c;
```

그룹화(다른 자료형)

```
int      a;
float    b;
double   c;
```

1.1 구조체란 (2/18)

▶ 구조체의 정의

① 구조체의 시작을 알리는 키워드

② 구조체 이름

```
struct   point
{
    int x;   ③ 멤버 변수
    int y;   ③ 멤버 변수
};
```

① **구조체 키워드:** 구조체의 시작을 알리는 struct 키워드 지정

② **구조체 이름:** 구조체를 구분하는 이름

③ **멤버변수:** 구조체를 구성하는 구조체 멤버 변수의 이름

1.1 구조체란

▶배울 내용

① 구조체 정의

② 구조체 변수

③ 구조체 변수로 멤버 변수에 접근하기

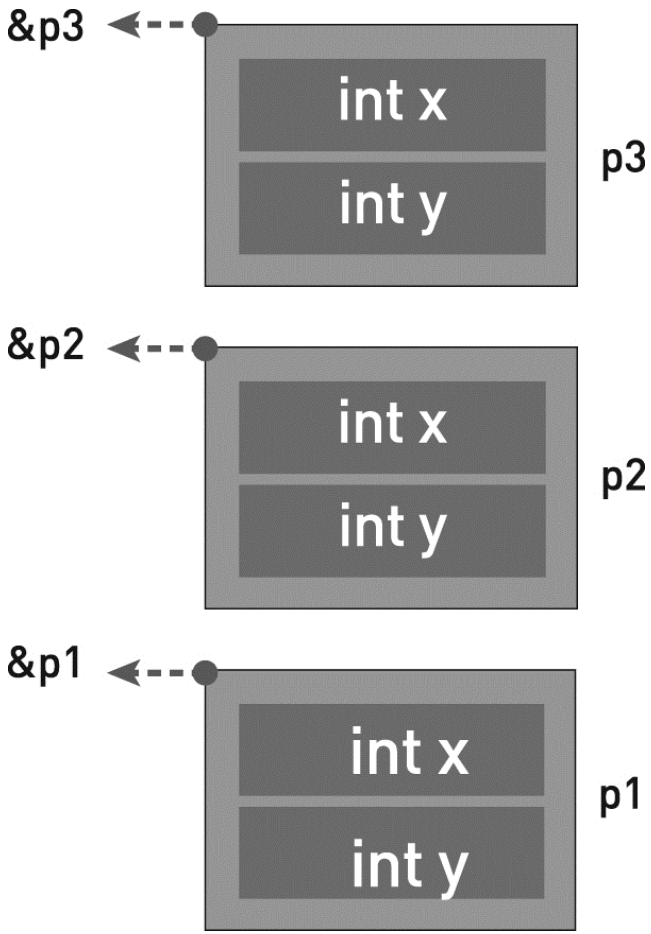
④ 구조체 변수의 초기화

⑤ 구조체 변수의 복사

1.1 구조체란 (3/18)

▶ 구조체 정의, 구조체 변수 선언 – **동시에**

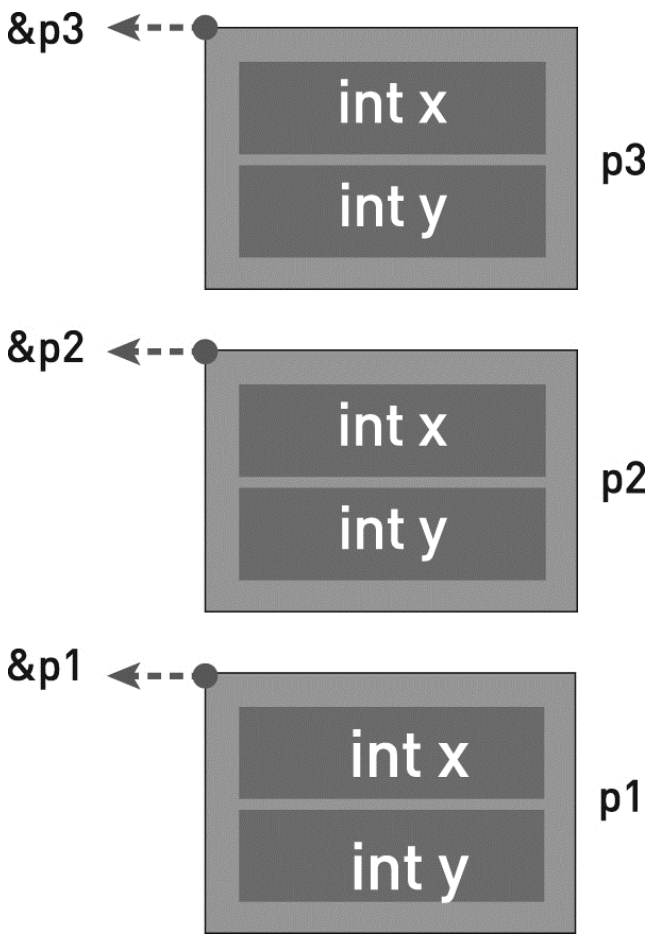
```
#include <stdio.h>

struct point
{
    int x;
    int y;
} p1, p2, p3;

int main(void)
{
    ...
    return 0;
}
```



1.1 구조체란 (4/18)

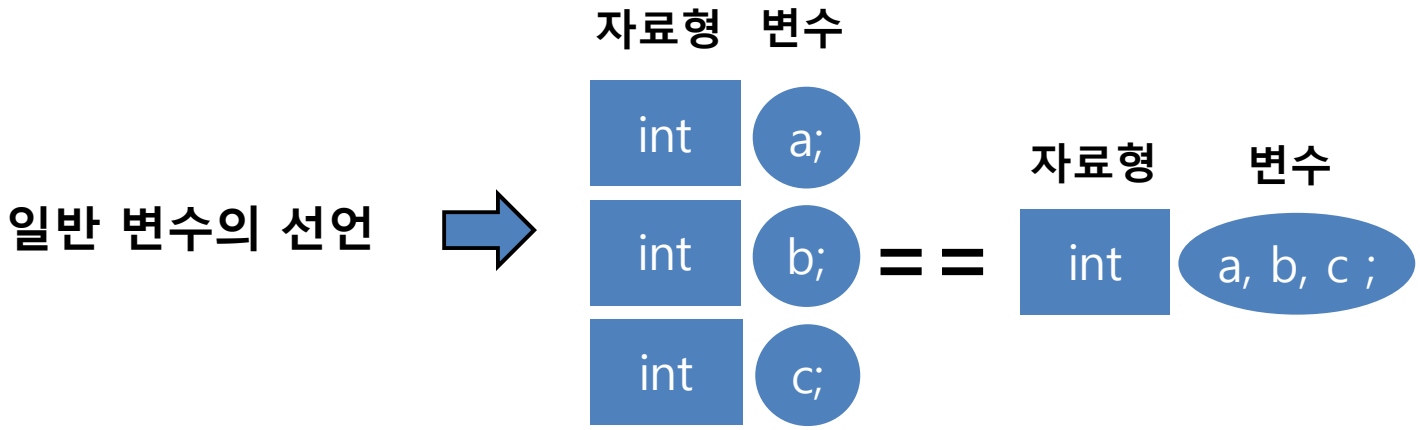
▶ 구조체 정의, 구조체 변수 선언 – 따로

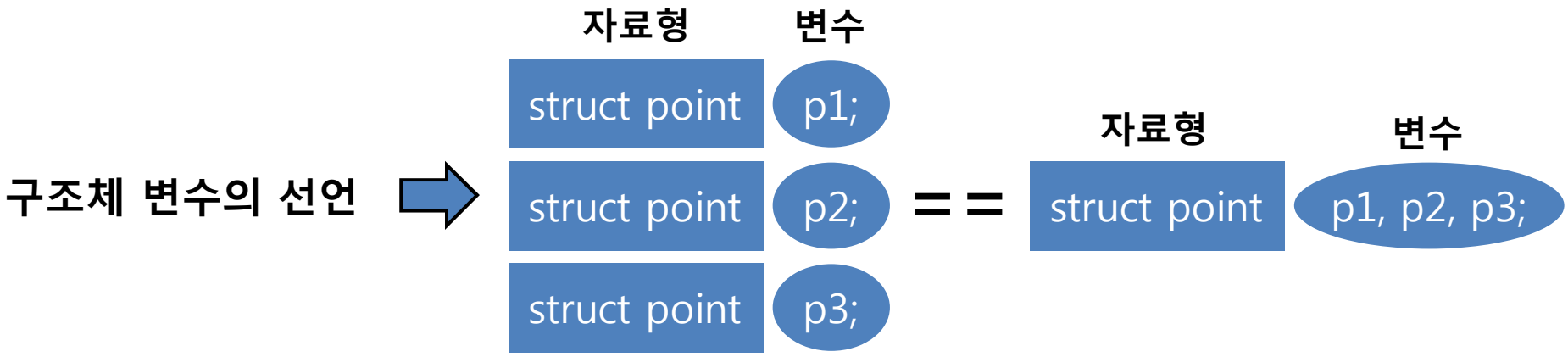
```
#include <stdio.h>

struct point
{
    int x;
    int y;
};

int main(void)
{
    struct point p1, p2, p3;
    ...
    return 0;
}
```



1.1 구조체란 (5/18)

▶ 일반 변수 vs. 구조체 변수





1.1 구조체란

▶배울 내용

① 구조체 정의

② 구조체 변수

③ 구조체 변수로 멤버 변수에 접근하기

④ 구조체 변수의 초기화

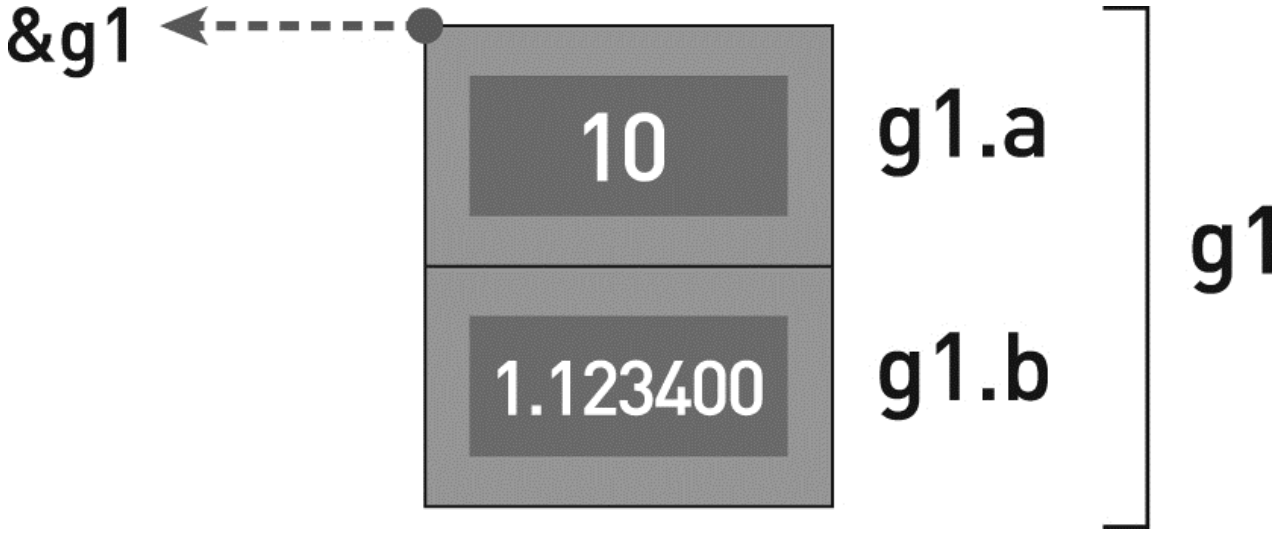
⑤ 구조체 변수의 복사

1.1 구조체란 (6/18)---[1-1.c 실습]

```
#include <stdio.h>
struct group                    // 구조체 정의
{
    int a;
    double b;
};
int main(void)
{
    struct group g1;            // 구조체 변수 g1 선언

    g1.a=10;                    // 구조체 변수로 멤버 변수 접근
    g1.b=1.1234;                // 구조체 변수로 멤버 변수 접근

    printf("g1.a의 값: %d ₩n", g1.a);
    printf("g1.b의 값: %lf ₩n", g1.b);
    return 0;
}
```

1.1 구조체란 (7/18)---[1-1.c 분석]

▶ 구조체 변수 g1의 메모리 구조

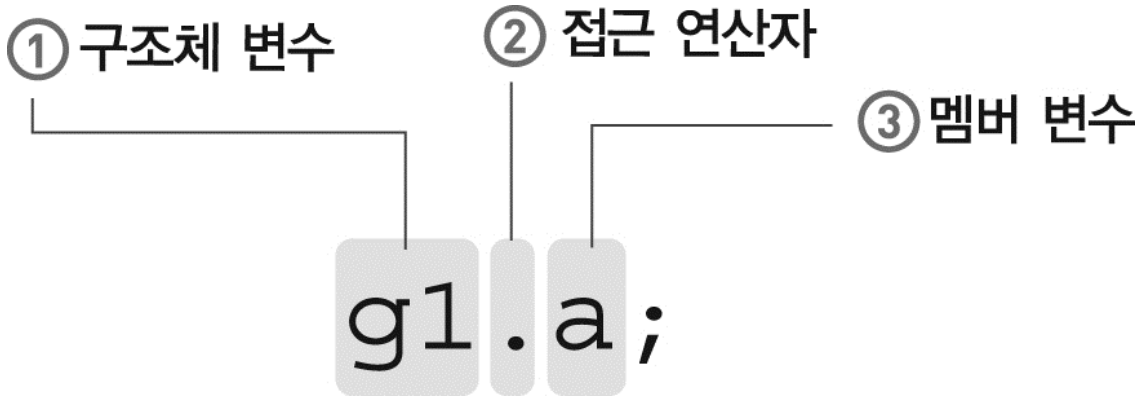


1.1 구조체란 (8/18)---[1-2.c 실습]

```c
#include <stdio.h>

struct group                    // 구조체 정의
{
    int a;
    double b;
};

int main(void)
{
    struct group g1;                    // 구조체 변수 g1 선언

    scanf("%d %lf", &g1.a, &g1.b);      // 데이터 입력

    printf("g1.a의 값: %d ₩n", g1.a);
    printf("g1.b의 값: %lf ₩n", g1.b);

    return 0;
}
```

1.1 구조체란 (9/18)

▶ 구조체 변수를 사용하는 법



- ✓ **구조체 변수:** 멤버 변수에 접근하게 해주는 구조체 변수의 이름을 지정
- ✓ **접근 연산자:** 구조체 변수로 멤버 변수에 접근하는 연산자 지정
- ✓ **멤버 변수:** 접근하려는 멤버 변수의 이름을 지정

1.1 구조체란

▶배울 내용

① 구조체 정의

② 구조체 변수

③ 구조체 변수로 멤버 변수에 접근하기

④ 구조체 변수의 초기화

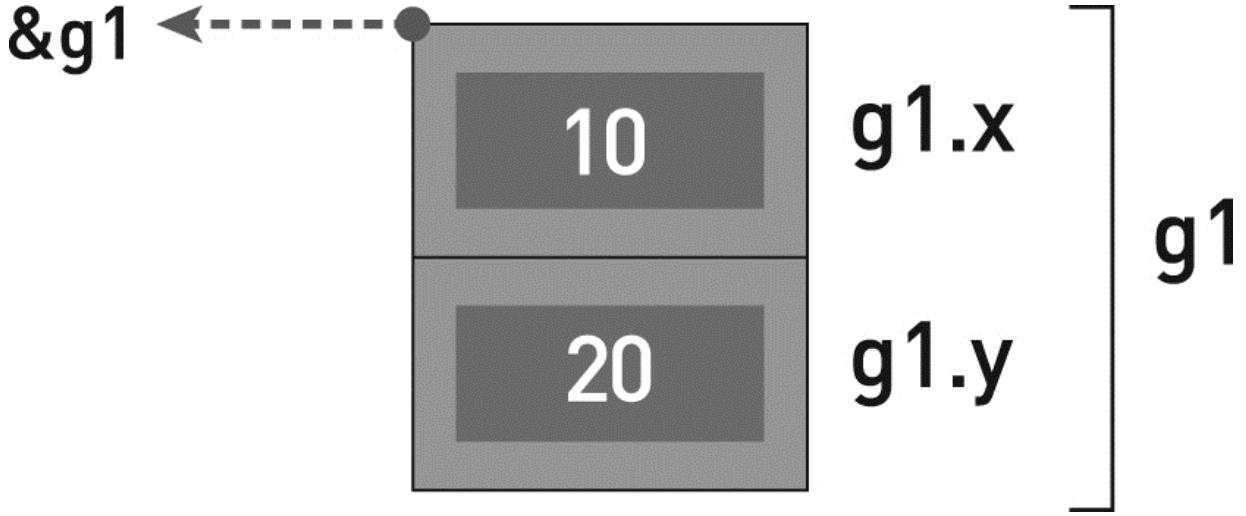
⑤ 구조체 변수의 복사

1.1 구조체란 (10/18)---[1-3.c 실습]

```
#include <stdio.h>

struct point
{
    int x;
    int y;
};

int main(void)
{
    struct point p1={10, 20};    // 구조체 변수의 초기화

    printf("%d %d ₩n", p1.x, p1.y);
    return 0;
}
```

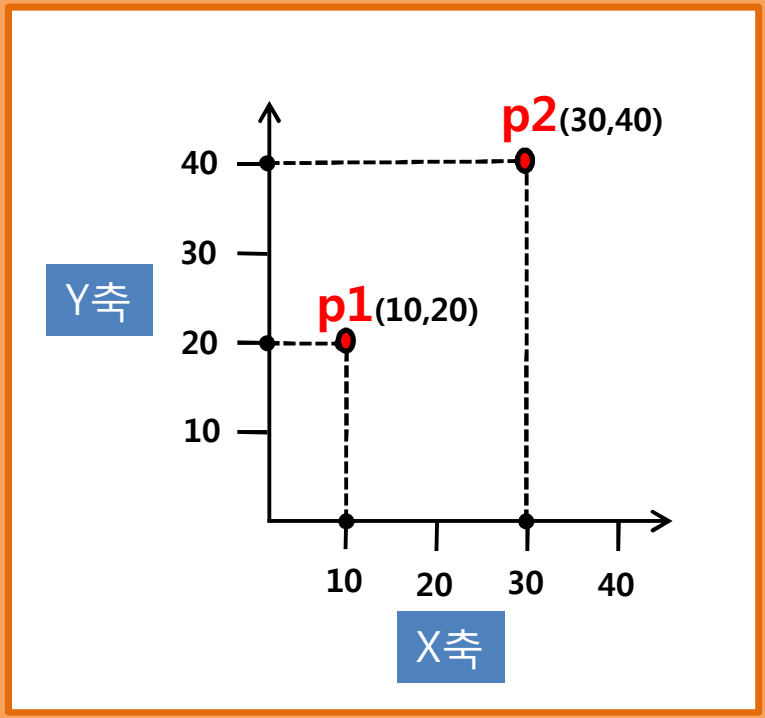
1.1 구조체란 (11/18)---[1-3.c 분석]

▶ 구조체 변수 p1의 메모리 구조



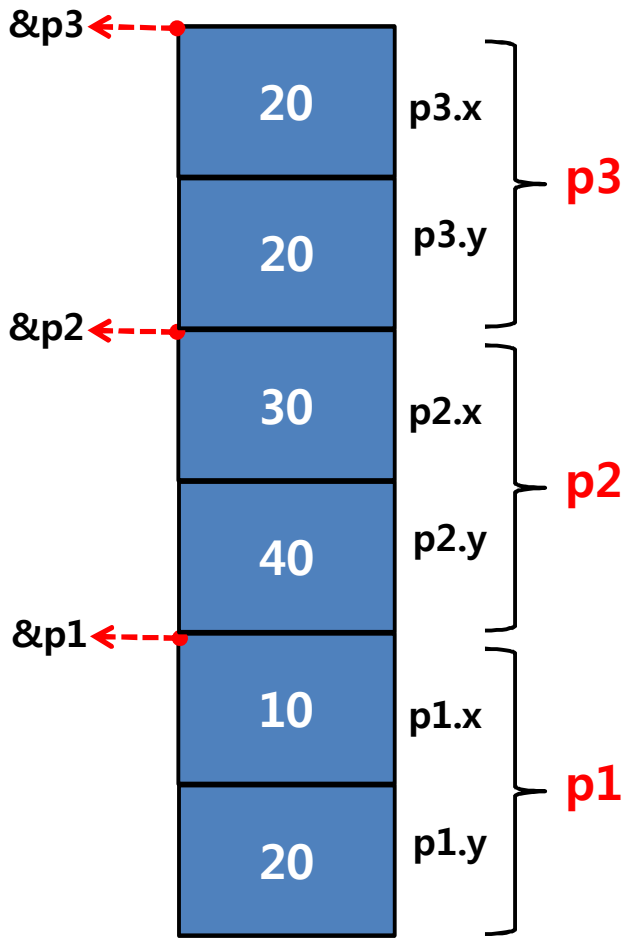
1.1 구조체란 (12/18)---[1-4.c 실습]

```
#include <stdio.h>
struct point
{
    int x;
    int y;
};
int main(void)
{
    struct point p1={10, 20};
    struct point p2={30, 40};
    struct point p3={0, 0};

    p3.x = p2.x - p1.x;
    p3.y = p2.y - p1.y;

    printf("%d %d ₩n", p3.x , p3.y);

    return 0;
}
```



1.1 구조체란 (13/18)---[1-4.c 분석]

▶ 구조체 변수 p1, p2, p3의 메모리 구조



1.1 구조체란 (14/18)

▶ 구조체 변수의 초기화

✓ 중괄호를 이용한 '구조체 변수'의 초기화 시 주의사항

- 구조체 변수의 **선언**과 구조체 변수의 **초기화**를 **따로 하면 에러**가 발생

```
struct point p1;
p1={10, 20};      // 에러
```

```
struct point p1 = {10, 20};  // 정상
```

```
struct point p1;
p1.x=10;          // 정상
p1.y=20;          // 정상
```

1.1 구조체란

▶배울 내용

① 구조체 정의

② 구조체 변수

③ 구조체 변수로 멤버 변수에 접근하기

④ 구조체 변수의 초기화

⑤ 구조체 변수의 복사

1.1 구조체란 (15/18)

▶ 구조체 변수의 복사

✓ 일반 변수의 복사와 같이 **구조체 변수 간 복사 가능**

```
int a=3;
int b=0;
b=a;          // 변수의 복사
printf("%d %d ₩n", a, b);
```

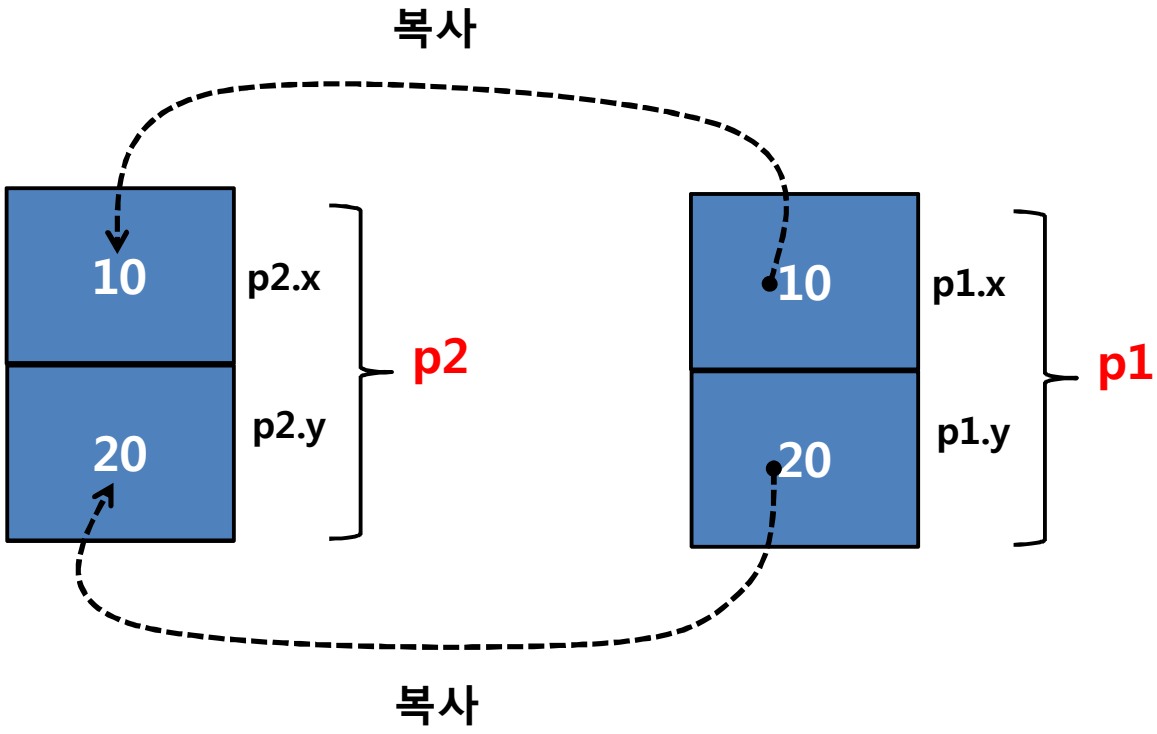
```
struct point p1={10, 20};
struct point p2={0, 0};
p2=p1       // 구조체 변수의 복사
```

1.1 구조체란 (16/18)---[1-5.c 실습]

```
#include <stdio.h>
struct point
{
    int x;
    int y;
};

int main(void)
{
    struct point p1={10, 20};
    struct point p2={0, 0};

    p2=p1;          // 구조체 변수 p2에 p1을 복사

    printf("%d %d ₩n", p1.x, p1.y);
    printf("%d %d ₩n", p2.x, p2.y);

    return 0;
}
```

1.1 구조체란 (17/18)---[1-5.c 분석]

▶ 구조체 변수 간의 복사



1.1 구조체란 (18/18)---[1-6.c 실습]

```
#include <stdio.h>
struct point
{
    int x;
    int y;
};

int main(void)
{
    struct point p1={10, 20};
    struct point p2={0, 0};

    p2+p1;    // 에러
    p2-p1;    // 에러

    return 0;
}
```

<---- 구조체 변수 간 산술연산 불가능

1.2 중첩 구조체

1.2 중첩 구조체

▶배울 내용

① 중첩 구조체

② 중첩 구조체의 초기화

③ typedef를 이용한 구조체의 재정의

1.2 중첩 구조체 (1/9)

▶ 중첩 구조체

✓ 구조체 내에 구조체가 포함

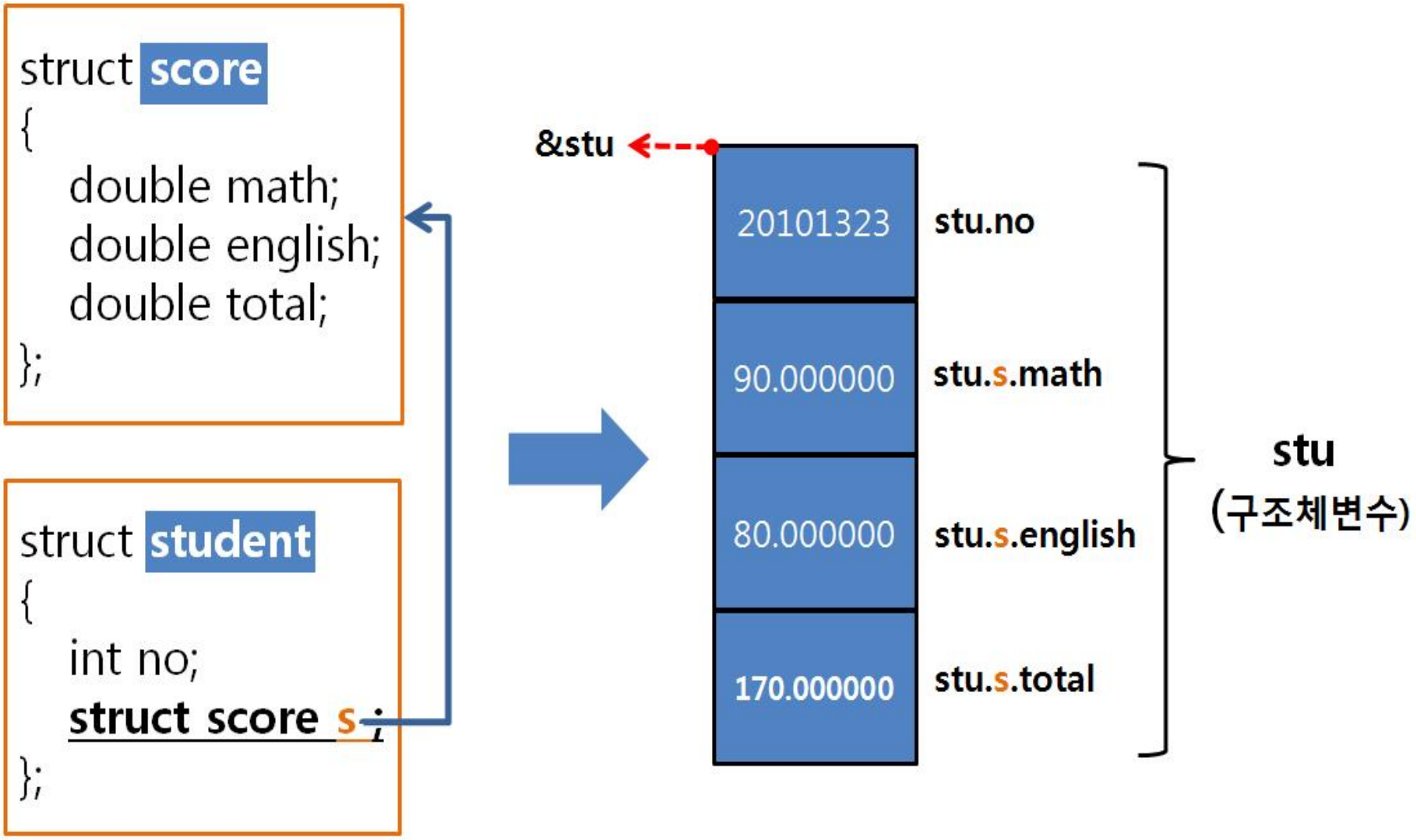
✓ '구조체 변수를 멤버변수로 사용한다.'

1.2 중첩 구조체 (2/9)---[1-7.c 실습]

```
#include <stdio.h>

struct score
{
    double math;
    double english;
    double total;
};
struct student
{
    int no;
    struct score s ;
};
```

```
int main(void)
{
    struct student stu;

    stu.no=20101323;
    stu.s.math=90;
    stu.s.english=80;
    stu.s.total=stu.s.math+stu.s.english;

    printf("학번: %d ₩n", stu.no);
    printf("총점: %lf ₩n", stu.s.total);
    return 0;
}
```

1.2 중첩 구조체 (3/9)---[1-7.c 분석]



1.2 중첩 구조체

▶배울 내용

① 중첩 구조체

② 중첩 구조체의 초기화

③ typedef를 이용한 구조체의 재정의

1.2 중첩 구조체 (4/9)---[1-8.c 실습]

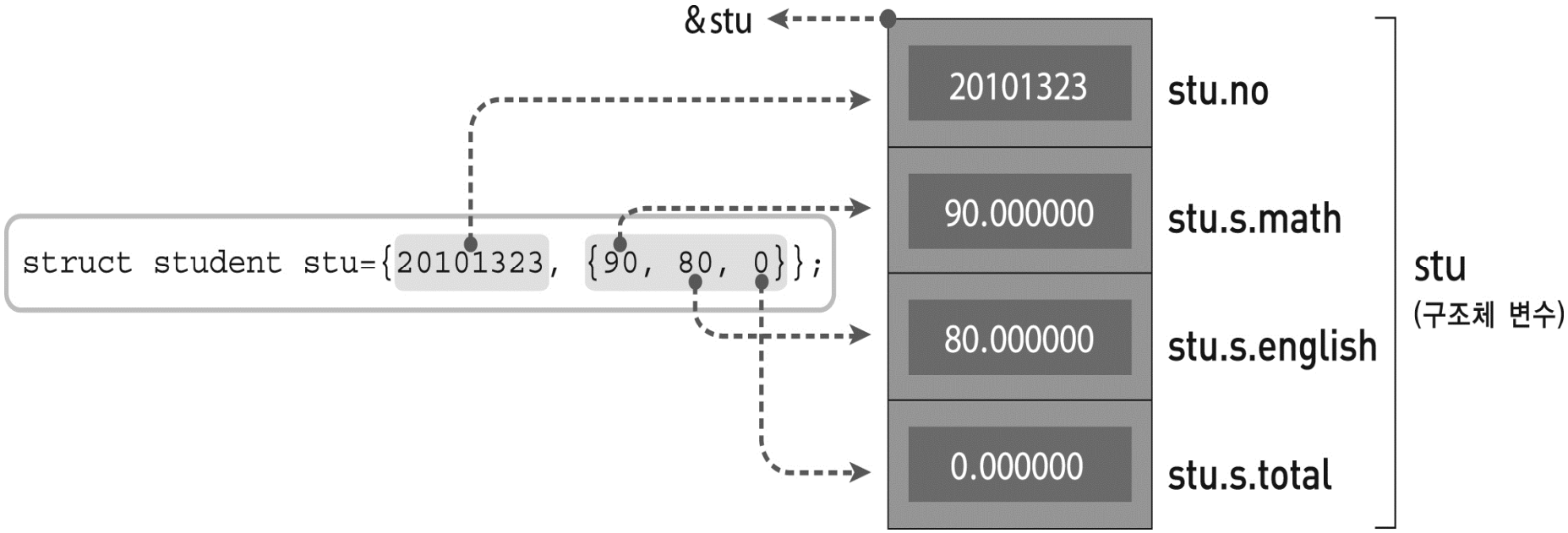
```
#include <stdio.h>
struct score
{
    double math;
    double english;
    double total;
};

struct student
{
    int no;
    struct score s;
};
```

중첩 구조체의 초기화 방법 2가지

```
int main(void)
{
    struct student stu={20101323, {90, 80, 0}};
 // struct student stu={20101323, 90, 80, 0};

    stu.s.total=stu.s.math+stu.s.english;
    printf("학번: %d ₩n", stu.no);
    printf("총점: %lf ₩n", stu.s.total);

    return 0;
}
```

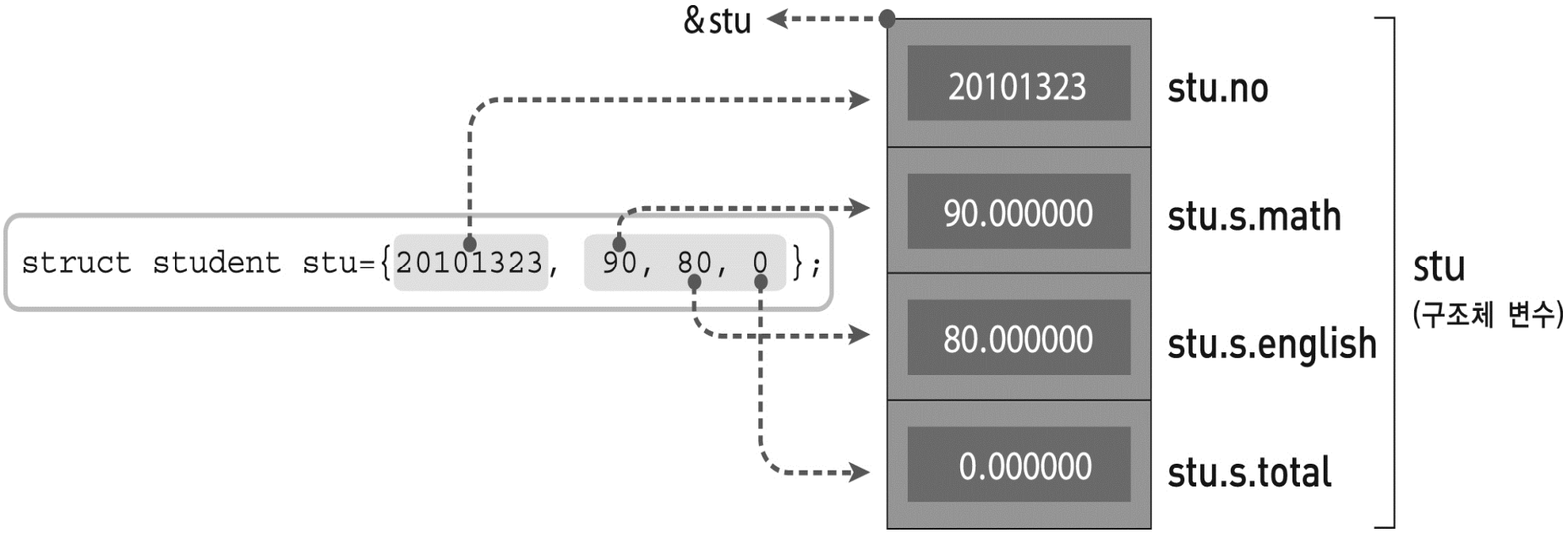
1.2 중첩 구조체 (5/9)---[1-8.c 분석]

(1) 중첩 중괄호를 사용한 초기화



1.2 중첩 구조체 (6/9)---[1-8.c 분석]

(2) 중첩 중괄호를 생략한 초기화



1.2 중첩 구조체

▶배울 내용

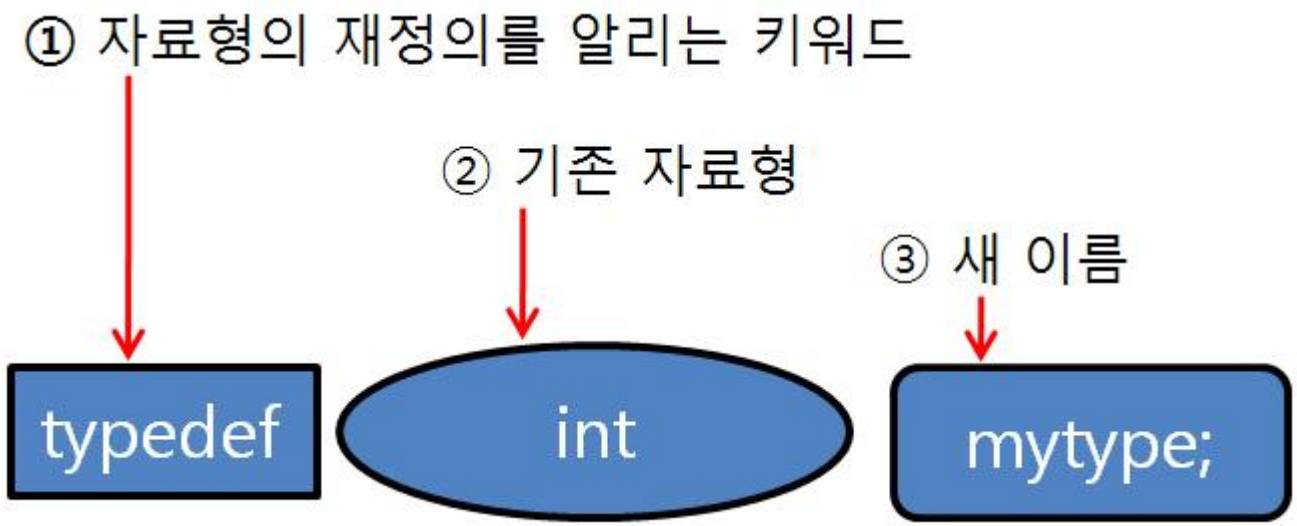
① 중첩 구조체

② 중첩 구조체의 초기화

③ typedef를 이용한 구조체의 재정의

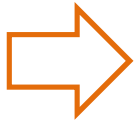
1.2 중첩 구조체 (7/9)

▶ typedef의 사용 방법



1.2 중첩 구조체 (8/9)---[1-9.c 실습(1/2)]

```
#include <stdio.h>

typedef struct score
{
    double math;
    double english;
    double average;
} SCORE;

struct student
{
    int no;
    SCORE s;  // struct socre s;
};

typedef struct student STUDENT ;
```



```
int main(void)
{
    STUDENT stu={20101323, {90, 80, 0}};

    stu.s. average=(stu.s.math+stu.s.english)/2;
    printf("학번: %d ₩n", stu.no);
    printf("평균점수: %lf ₩n", stu.s.average);
    return 0;
}
```

1.2 중첩 구조체 (9/9)

① 구조체 정의와 동시에 typedef 선언

```
typedef struct score
{
    double math;
    double english;
    double average;

} SCORE;
```

```
typedef struct student
{
    int no;
    SCORE s;

} STUDENT;
```

② 구조체 정의와 개별적으로 typedef 선언

```
struct score
{
    double math;
    double english;
    double average;

};
typedef struct score SCORE
```

```
struct student
{
    int no;
    SCORE s;
};
typedef struct score STUDENT;
```

1.3 구조체와 배열

1.3 구조체와 배열

▶배울 내용

① 멤버 변수로 배열 사용하기

② 구조체 변수로 배열 사용하기

③ 멤버 변수로 배열을 사용할 때 주의 사항

1.3 구조체와 배열 (1/7)---[1-10.c 실습(1/2)]

```
#include <stdio.h>

struct student
{
    char no[10];          // 학번 (멤버변수에 배열 사용)
    char name[20];        // 이름 (멤버변수에 배열 사용)
    double math;          // 수학 점수
    double english;       // 영어 점수
    double total;         // 총점
};

int main(void)
{
    struct student stu1={"20101323", "Park", 80, 80, 0};  // 학생 1의 정보
    struct student stu2={"20101324", "Kim", 95, 85, 0};   // 학생 2의 정보
    struct student stu3={"20101325", "Lee", 100, 90, 0};  // 학생 3의 정보
```

1.3 구조체와 배열 (2/7)---[1-10.c 실습(2/2)]

```
    stu1.total=stu1.math+stu1.english;
    printf("학번: %s, 이름: %s ₩n", stu1.no, stu1.name);
    printf("총점: %lf ₩n", stu1.total);

    printf("₩n");
    stu2.total=stu2.math+stu2.english;
    printf("학번: %s, 이름: %s ₩n", stu2.no, stu2.name);
    printf("총점: %lf ₩n", stu2.total);

    printf("₩n");
    stu3.total=stu3.math+stu3.english;
    printf("학번: %s, 이름: %s ₩n", stu3.no, stu3.name);
    printf("총점: %lf ₩n", stu3.total);

    return 0;
}
```

1.3 구조체와 배열

▶배울 내용

① 멤버 변수로 배열 사용하기

② 구조체 변수로 배열 사용하기

③ 멤버 변수로 배열을 사용할 때 주의 사항

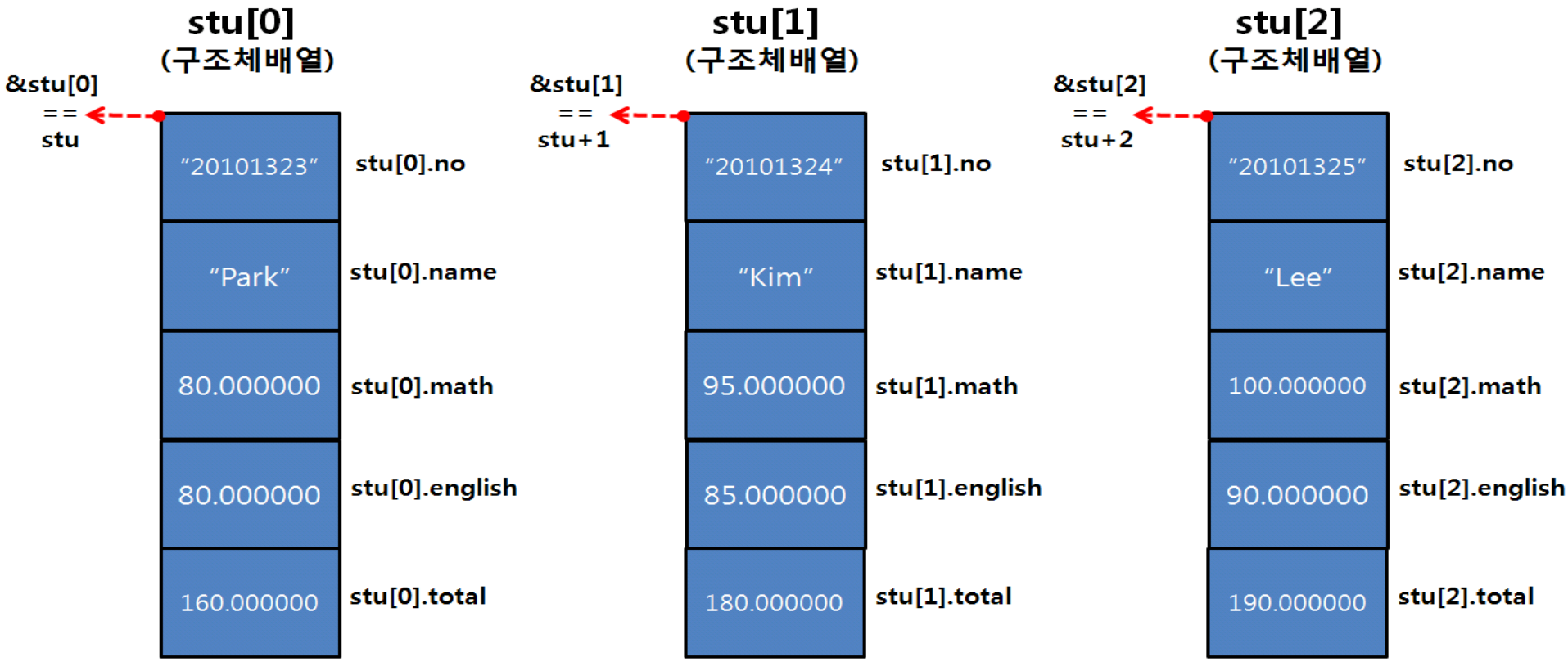
1.3 구조체와 배열 (3/7)---[1-11.c 실습(1/2)]

```
#include <stdio.h>
struct student
{
    char no[10];
    char name[20];
    double math;
    double english;
    double total;
};
```

구조체 변수로 배열 사용

```
int main(void)
{
    int i=0;
    struct student stu[3]={
        {"20101323", "Park", 80, 80, 0},
        {"20101324", "Kim", 95, 85, 0},
        {"20101325", "Lee", 100, 90, 0}
    };

    for(i=0; i<3; i++)
    {
        stu[i].total=stu[i].math+stu[i].english;
        printf("학번: %s, 이름: %s ₩n", stu[i].no, stu[i].name);
        printf("총점: %lf ₩n", stu[i].total);
        printf("₩n");
    }
    return 0;
}
```

1.3 구조체와 배열 (4/7)---[1-11.c 분석]



1.3 구조체와 배열

▶배울 내용

① 멤버 변수로 배열 사용하기

② 구조체 변수로 배열 사용하기

③ 멤버 변수로 배열을 사용할 때 주의 사항

1.3 구조체와 배열 (5/7)---[1-13.c 실습]

```
#include <stdio.h>
struct student
{
    char no[10];      // 멤버 변수로 배열 선언
    char name[20];    // 멤버 변수로 배열 선언
};
int main(void)
{
    int i=0;
    struct student stu;

    stu.no="20101323";   // 에러
    stu.name="Park";     // 에러
    printf("학번: %s, 이름: %s ₩n", stu.no, stu.name);

    return 0;
}
```

<<에러>>
배열의 시작 주소에 문자열을 입력

1.3 구조체와 배열 (6/7)---[1-14.c 실습]

```
#include <stdio.h>
#include <string.h>
struct student
{
    char no[10];        // 멤버 변수로 배열 선언
    char name[20];      // 멤버 변수로 배열 선언
};
int main(void)
{
    int i=0;
    struct student stu;

    strcpy (stu.no, "20101323");
    strcpy (stu.name, "Park");
    printf("학번: %s, 이름: %s ₩n", stu.no, stu.name);

    return 0;
}
```

<<에러 해결 1>>
strcpy()함수 사용(PART3-2장 참조)

1.3 구조체와 배열 (7/7)---[참고]

```
#include <stdio.h>
struct student
{
    char* no;       // 멤버 변수로 포인터 선언
    char* name;     // 멤버 변수로 포인터 선언
};
int main(void)
{
    int i=0;
    struct student stu;

    stu.no = "20101323";
    stu.name = "Park";
    printf("학번: %s, 이름: %s ₩n", stu.no, stu.name);

    return 0;
}
```

<<에러 해결 2>>
멤버 변수로 포인터 선언(다음 슬라이드참조)

1.4 구조체와 포인터

1.4 구조체와 포인터

▶배울 내용

① 멤버 변수로 포인터 사용하기

② 구조체 변수로 포인터 사용하기

③ 자기 참조 구조체와 외부 참조 구조체

1.4 구조체와 포인터 (1/15)

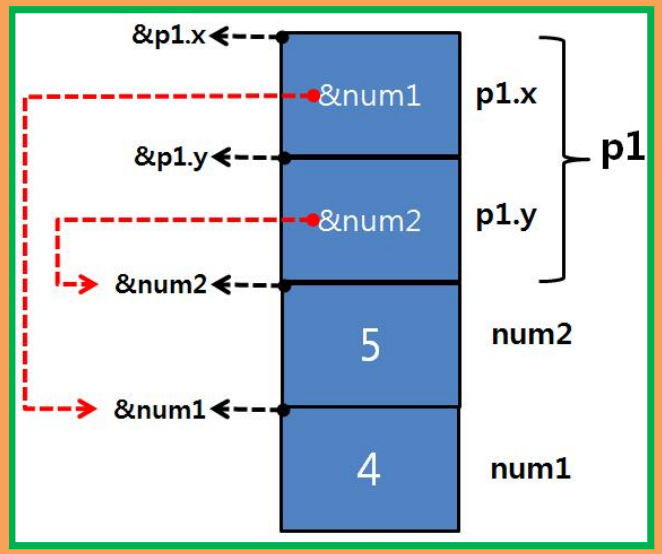
▶ 멤버 변수로 포인터 사용하기

```
struct point
{
    int* x;    // 멤버 변수로 1차원 포인터 선언
    int* y;    // 멤버 변수로 1차원 포인터 선언
};
```

```
struct point
{
    int* x;    // 멤버 변수로 1차원 포인터 선언
    int** y;   // 멤버 변수로 2차원 포인터 선언
}
```

1.4 구조체와 포인터 (3/15)---[1-15.c 실습]

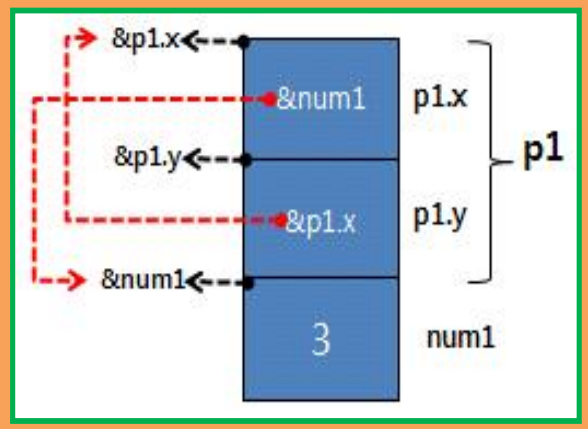
```
#include <stdio.h>
struct point
{
    int* x;   // 1차원 포인터 멤버변수
    int* y;   // 1차원 포인터 멤버변수
};
int main(void)
{
    int num1=4;
    int num2=5;
    struct point p1;

    p1.x=&num1;
    p1.y=&num2;

    printf("%d %d ₩n", num1, num2);
    printf("%d %d ₩n", *p1.x, *p1.y);
    return 0;
}
```



'. 연산자가 *연산자보다 우선순위가 높다.'

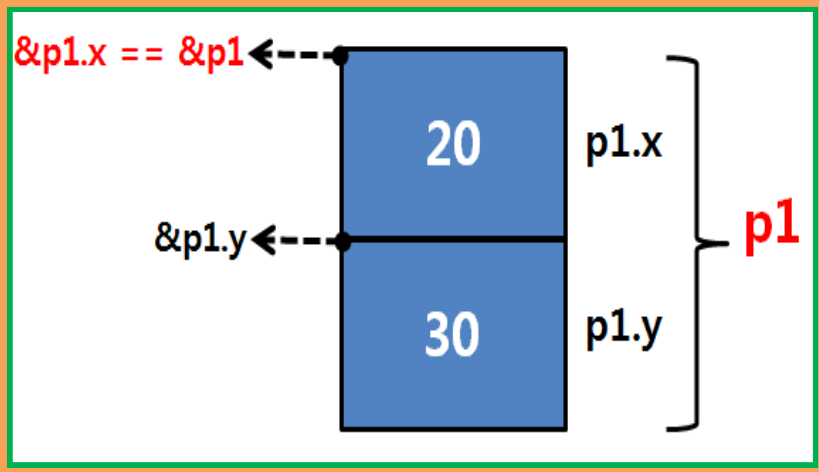
1.4 구조체와 포인터 (4/15)---[1-16.c 실습]

```
#include<stdio.h>
struct point
{
    int*  x;  // 1차원 포인터 멤버변수
    int** y;  // 2차원 포인터 멤버변수

};
int main(void)
{
    int num1 = 3;
    struct point p1;

    p1.x = &num1;
    p1.y = &p1.x;

    printf("%d %d %d ₩n", num1, *p1.x, **p1.y);

    return 0;
}
```



1.4 구조체와 포인터 (5/15)---[1-17.c 실습]

```
#include<stdio.h>
struct point
{
    int  x;
    int  y;
};
int main(void)
{
    struct point p1={20,30};
    printf("구조체 변수 p1의 주소: %x ₩n", &p1);
    printf("멤버 변수 p1.x의 주소: %x ₩n", &p1.x);

    return 0;
}
```



1.4 구조체와 포인터

▶배울 내용

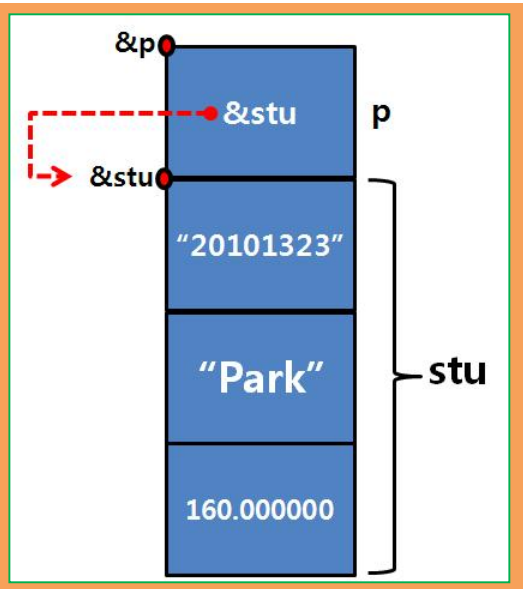
① 멤버 변수로 포인터 사용하기

② 구조체 변수로 포인터 사용하기

③ 자기 참조 구조체와 외부 참조 구조체

1.4 구조체와 포인터 (6/15)---[1-18.c 실습]

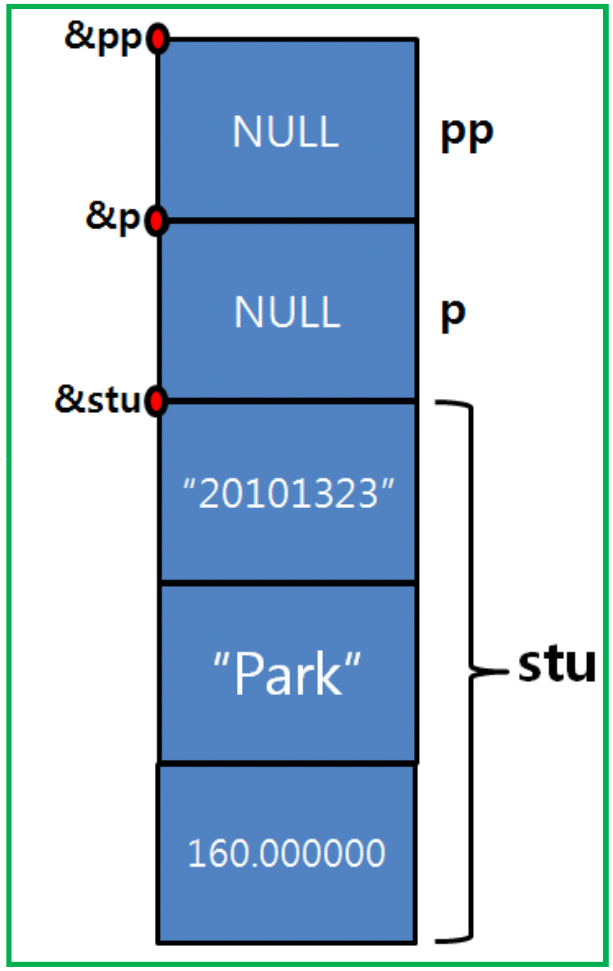
```
#include<stdio.h>
struct student
{
    char no[10];        // 학번
    char name[20];      // 이름
    double total;       // 총점
};
int main(void)
{
    struct student stu = {"20101323", "Park", 160};
    struct student* p=NULL;  // 1차원 구조체 포인터 변수 선언

    p = &stu;
    printf("%s %s %lf ₩n",stu.no, stu.name, stu.total);
    printf("%s %s %lf ₩n",(*p).no, (*p).name, (*p).total);
    printf("%s %s %lf ₩n",p->no, p->name, p->total);

    return 0;
}
```




```
(*p).no == p->no
```

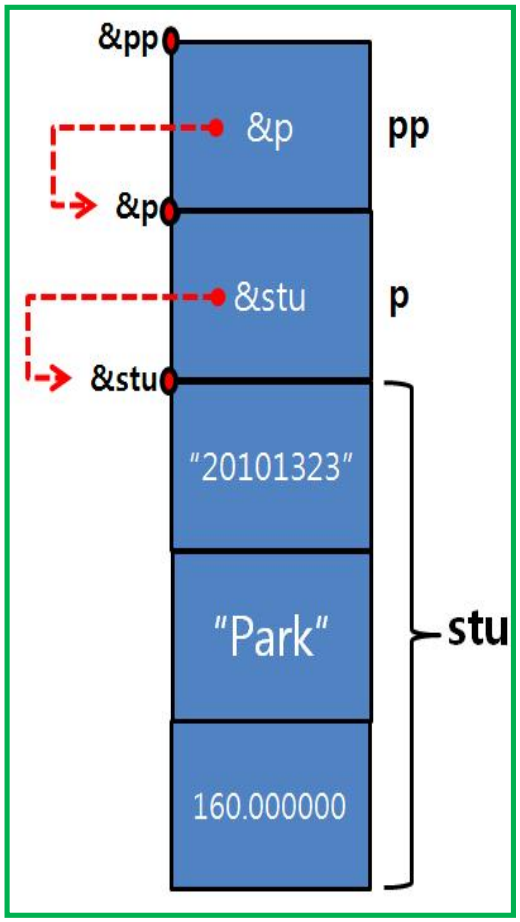
1.4 구조체와 포인터 (7/15)---[1-19.c 실습(1/2)]

```
#include<stdio.h>
struct student
{
  char no[10];        // 학번
  char name[20];      // 이름
  double total;       // 총점
};
int main(void)
{
  struct student stu = {"20101323", "Park", 160};
  struct student*   p=NULL;
  struct student** pp=NULL;
```



1.4 구조체와 포인터 (8/15)---[1-19.c 실습(2/2)]

```
 p = &stu;
 pp = &p;

 printf("%s %s %lf ₩n",stu.no, stu.name, stu.total);

 printf("%s %s %lf ₩n",(*p).no, (*p).name, (*p).total);
 printf("%s %s %lf ₩n",p->no, p->name, p->total);

 printf("%s %s %lf ₩n",(**pp).no, (**pp).name, (**pp).total);
 printf("%s %s %lf ₩n",(*pp)->no, (*pp)->name, (*pp)->total);
 return 0;
}
```



1.4 구조체와 포인터

▶배울 내용

① 멤버 변수로 포인터 사용하기

② 구조체 변수로 포인터 사용하기

③ 자기 참조 구조체와 외부 참조 구조체

1.4 구조체와 포인터 (9/15)

자기 참조 구조체

```
struct student
{
    char name[20];
    int age;
    struct student* p;
};
```

구조체 내에서 자기 구조체 참조

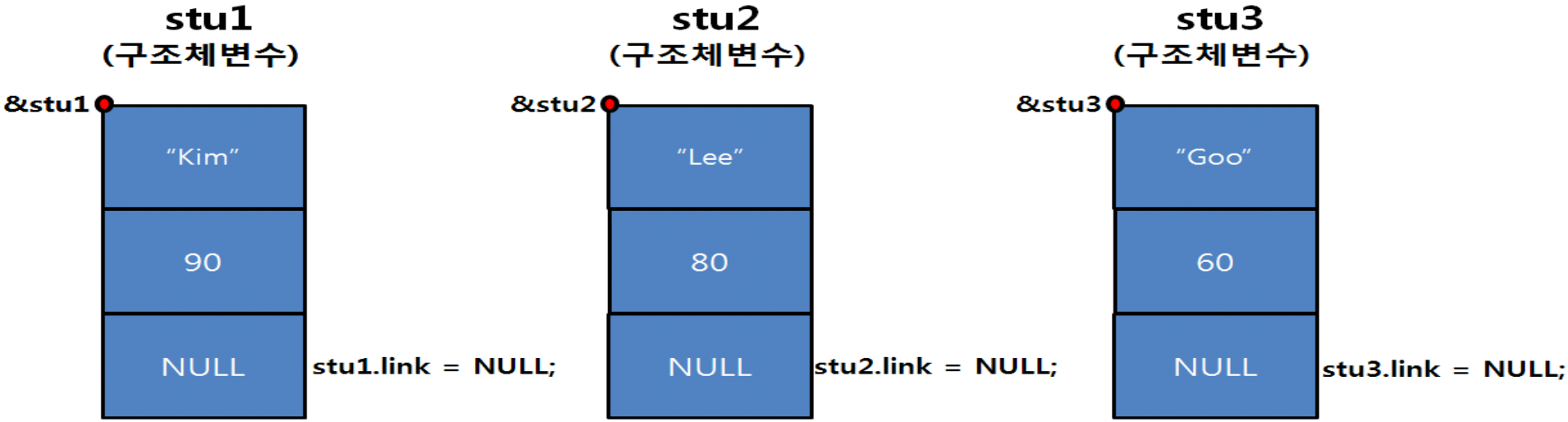
외부 참조 구조체

```
struct student
{
    char name[20];
    int age;
    struct score* p;
};
```

구조체 내에서 외부 구조체 참조

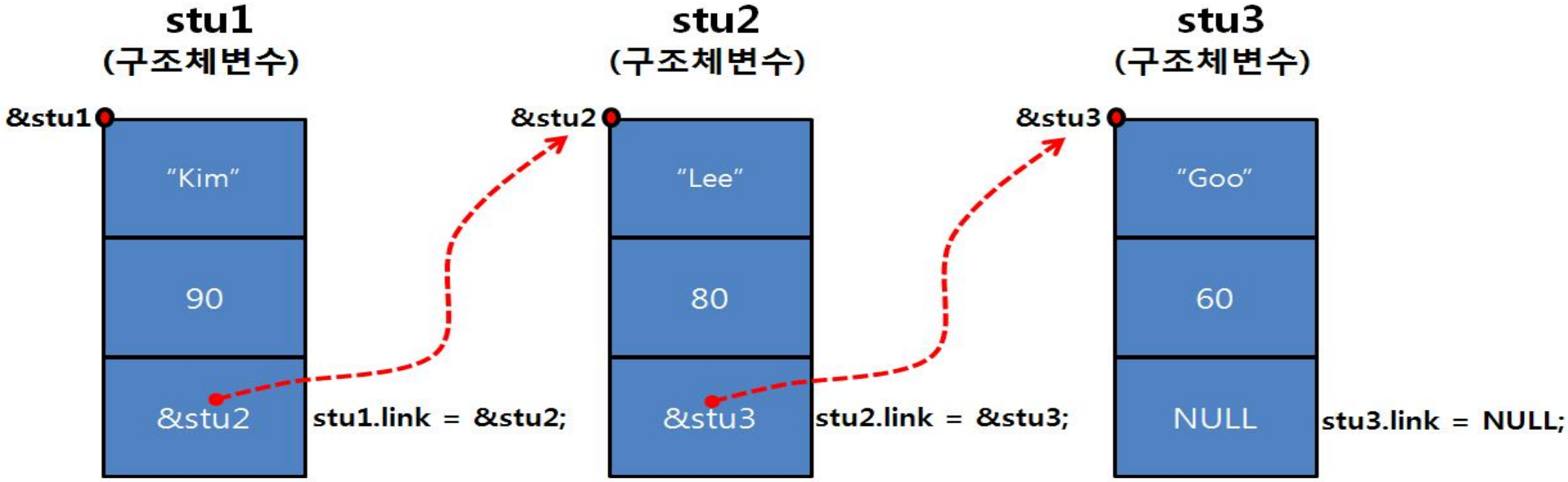
1.4 구조체와 포인터 (10/15)---[1-20.c 실습(1/2)]

```c
#include<stdio.h>
struct student
{
    char name[20];        // 이름
    int money;            // 나이
    struct student* link; // 자기 참조 구조체 포인터 변수
};
int main(void)
{
    struct student stu1 = {"Kim", 90, NULL};
    struct student stu2 = {"Lee", 80, NULL};
    struct student stu3 = {"Goo", 60, NULL};
```



1.4 구조체와 포인터 (11/15)---[1-20.c 실습(2/2)]

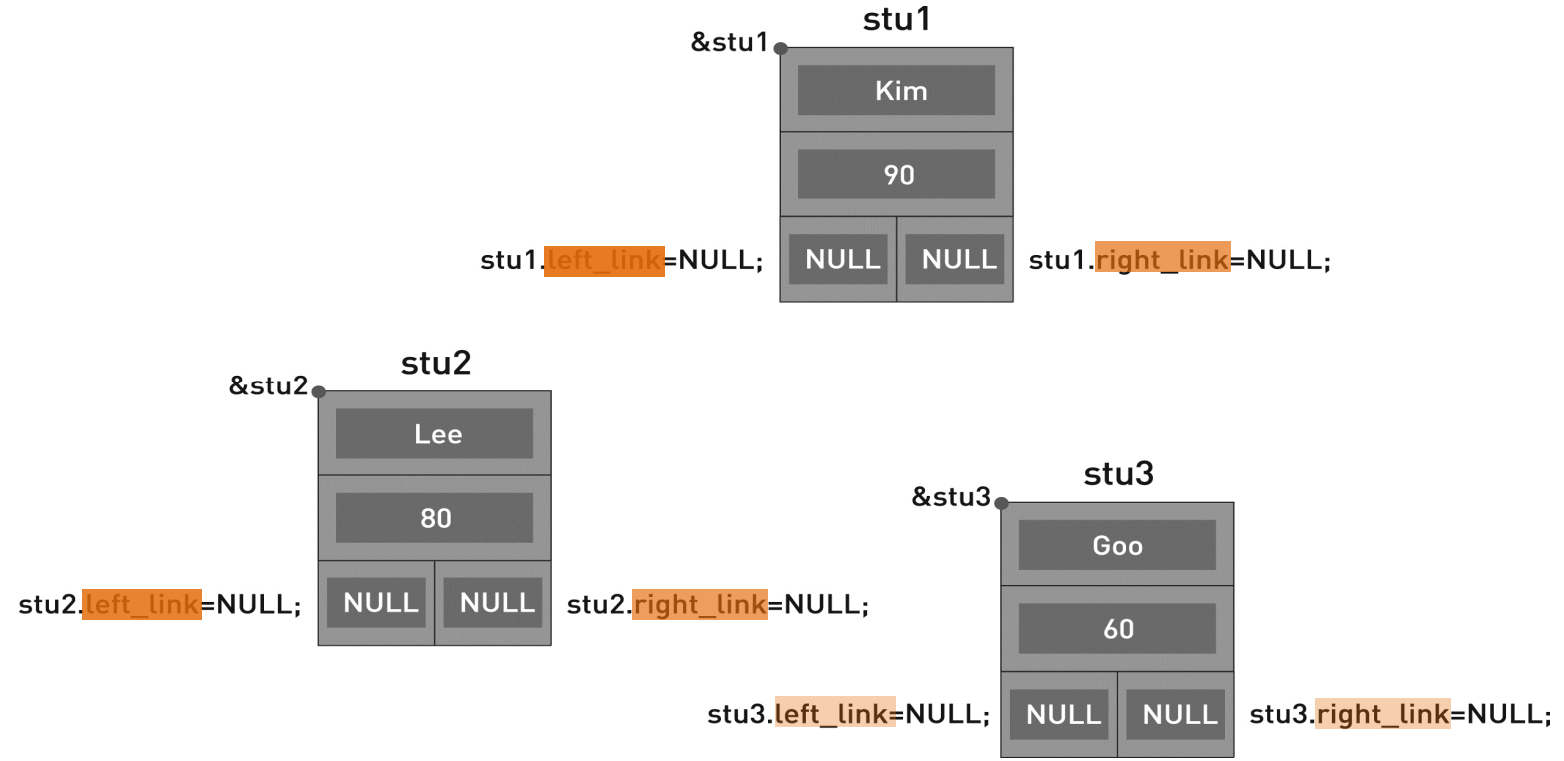
```
    stu1.link = &stu2;
    stu2.link = &stu3;

    printf("%s %d ₩n", stu1.name, stu1.money);
    printf("%s %d ₩n", stu1.link->name, stu1.link->money);
    printf("%s %d ₩n", stu1.link->link->name, stu1.link->link->money);
     return 0;
}
```



1.4 구조체와 포인터 (12/15)---[1-21.c 실습(1/2)]

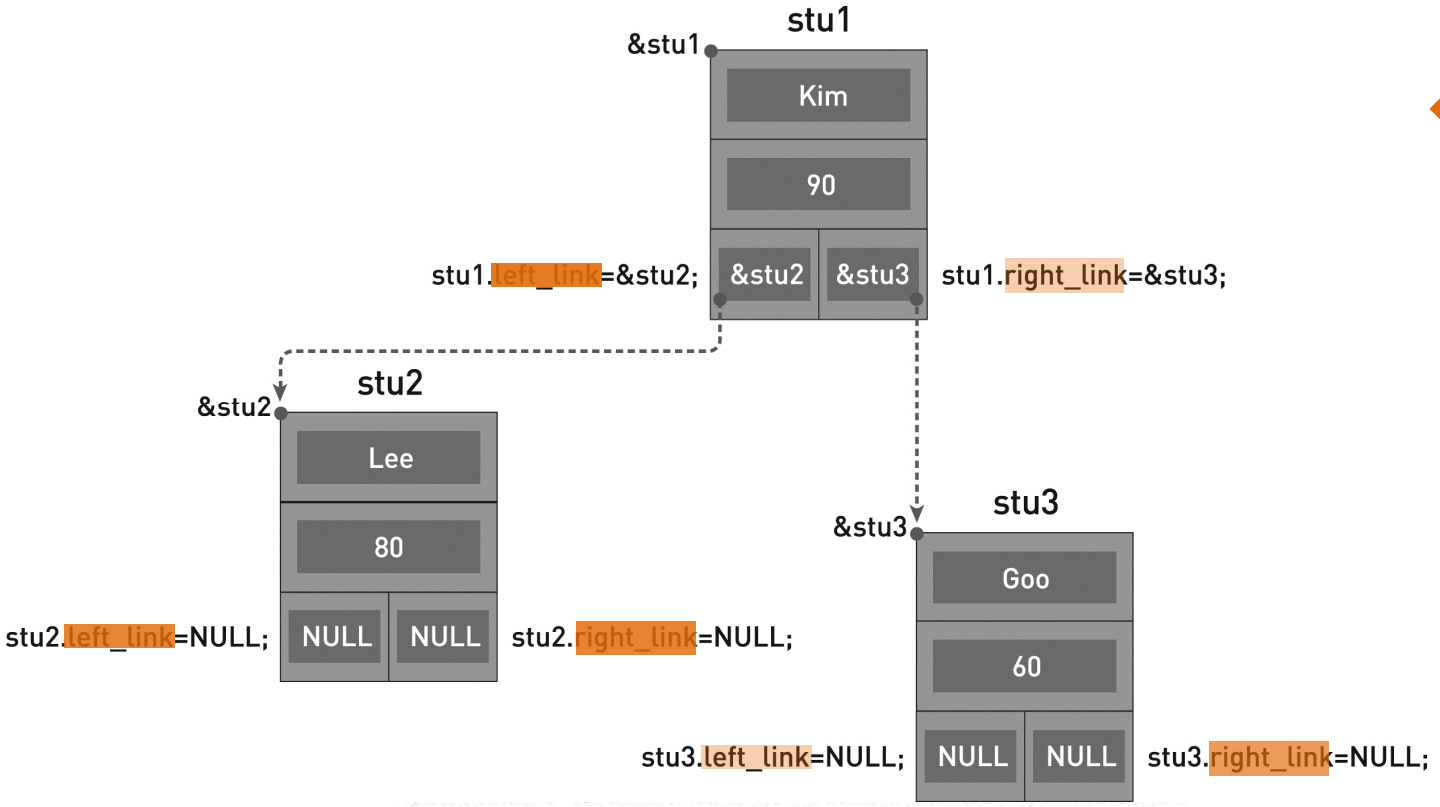
```
struct student
{
    char name[20];
    int money;
    struct student* left_link;
    struct student* right_link;
};
```

```
struct student stu1 = {"Kim", 90, NULL, NULL};
struct student stu2 = {"Lee", 80, NULL, NULL};
struct student stu3 = {"Goo", 60, NULL, NULL};
```



1.4 구조체와 포인터 (13/15)---[1-21.c 실습(2/2)]

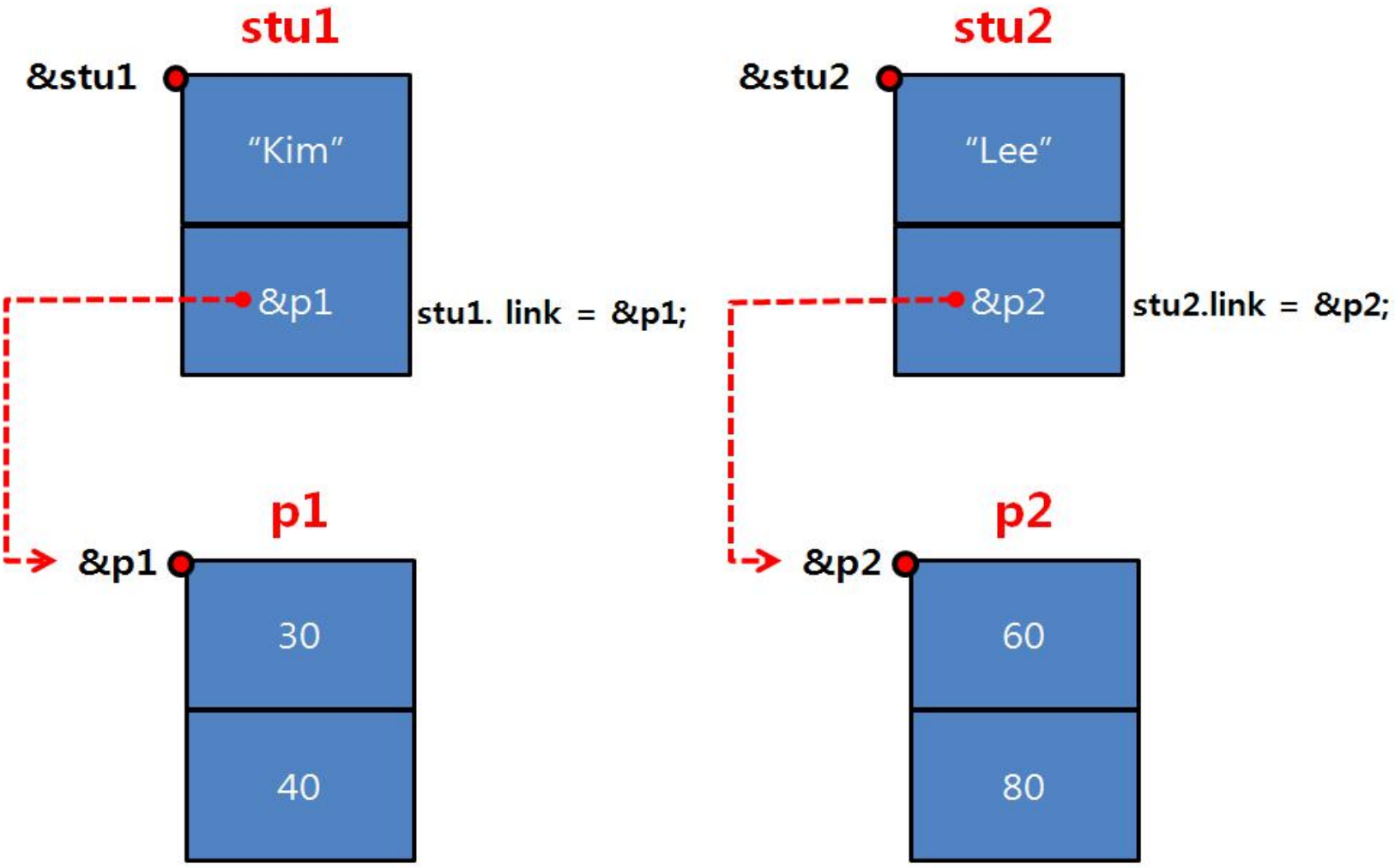
```
stu1.left_link = &stu2;
stu1.right_link= &stu3;

printf("%s %d ₩n", stu1.name, stu1.money);
printf("%s %d ₩n", stu1.left_link->name, stu1.left_link->money);
printf("%s %d ₩n", stu1.right_link->name, stu1.right_link->money);
```



1.4 구조체와 포인터 (14/15)---[1-22.c 실습(1/2)]

```
#include<stdio.h>
struct point {
    int x;      // x좌표
    int y;      // y좌표
};
struct student {
    char name[20];
    struct point* link;
};
int main(void)
{
    struct student stu1 = {"Kim", NULL};
    struct student stu2 = {"Lee", NULL};
    struct point p1 = {30, 40};
    struct point p2 = {60, 80};

    stu1.link = &p1;
    stu2.link = &p2;

    printf("%s %d %d ₩n", stu1.name, stu1.link->x, stu1.link->y);
    printf("%s %d %d ₩n", stu2.name, stu2.link->x, stu2.link->y);
    return 0;
}
```

1.4 구조체와 포인터 (15/15)---[1-22.c 분석]



1.5 구조체와 함수

1.4 구조체와 포인터

▶배울 내용

① 구조체를 함수의 인자로 전달하기
- 값에 의한 호출과 주소에 의한 호출

② 구조체를 함수의 반환형으로 전달하기
- 값 반환과 주소 반환

1.5 구조체와 함수 (1/10)

▶ 값에 의한 호출(Call by value)

함수의 인자 : 정수형 변수

```
void function(int a)
{
  ...
}
```

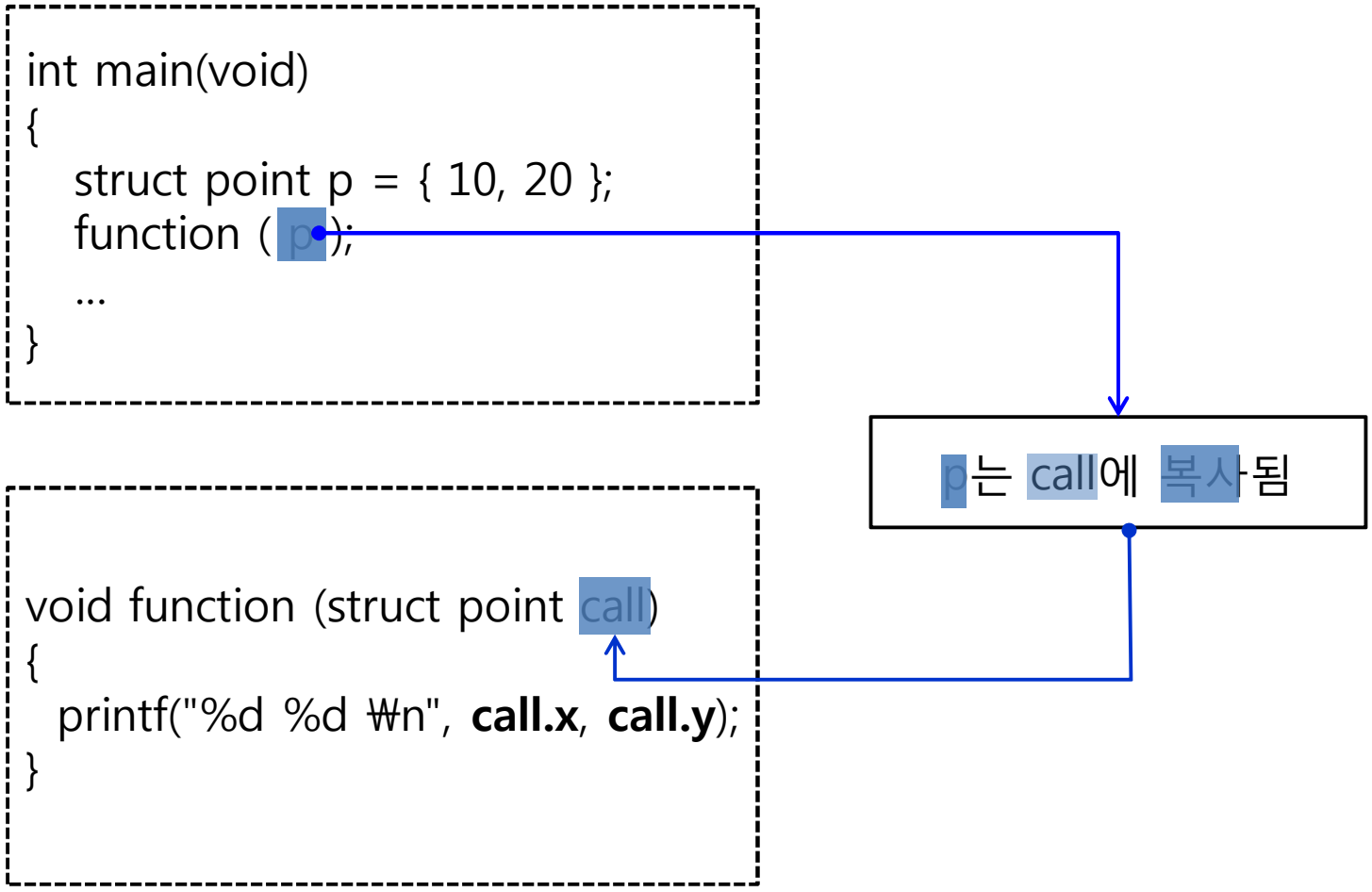
함수의 인자 : 구조체형 변수

```
void function(struct point a)
{
  ...
}
```

1.5 구조체와 함수 (2/10)---[1-23.c 실습]

```
#include<stdio.h>
struct point                                    // 구조체 정의
{
    int x;
    int y;
};

void function (struct point call);      // 함수의 선언

int main(void)
{
    struct point p = {10, 20};
    function(p);                                // 값에 의한 호출(call by value)
    return 0;
}

void function (struct point call)       // 함수의 정의
{
    printf("%d %d ₩n", call.x, call.y);  // 10, 20 출력
}
```

1.5 구조체와 함수 (3/10)---[1-23.c 분석]



1.5 구조체와 함수 (4/10)

▶ 주소에 의한 호출 (call by reference)

함수의 인자 : 정수형 포인터 변수

```
void function(int* a)
{
  ...
}
```

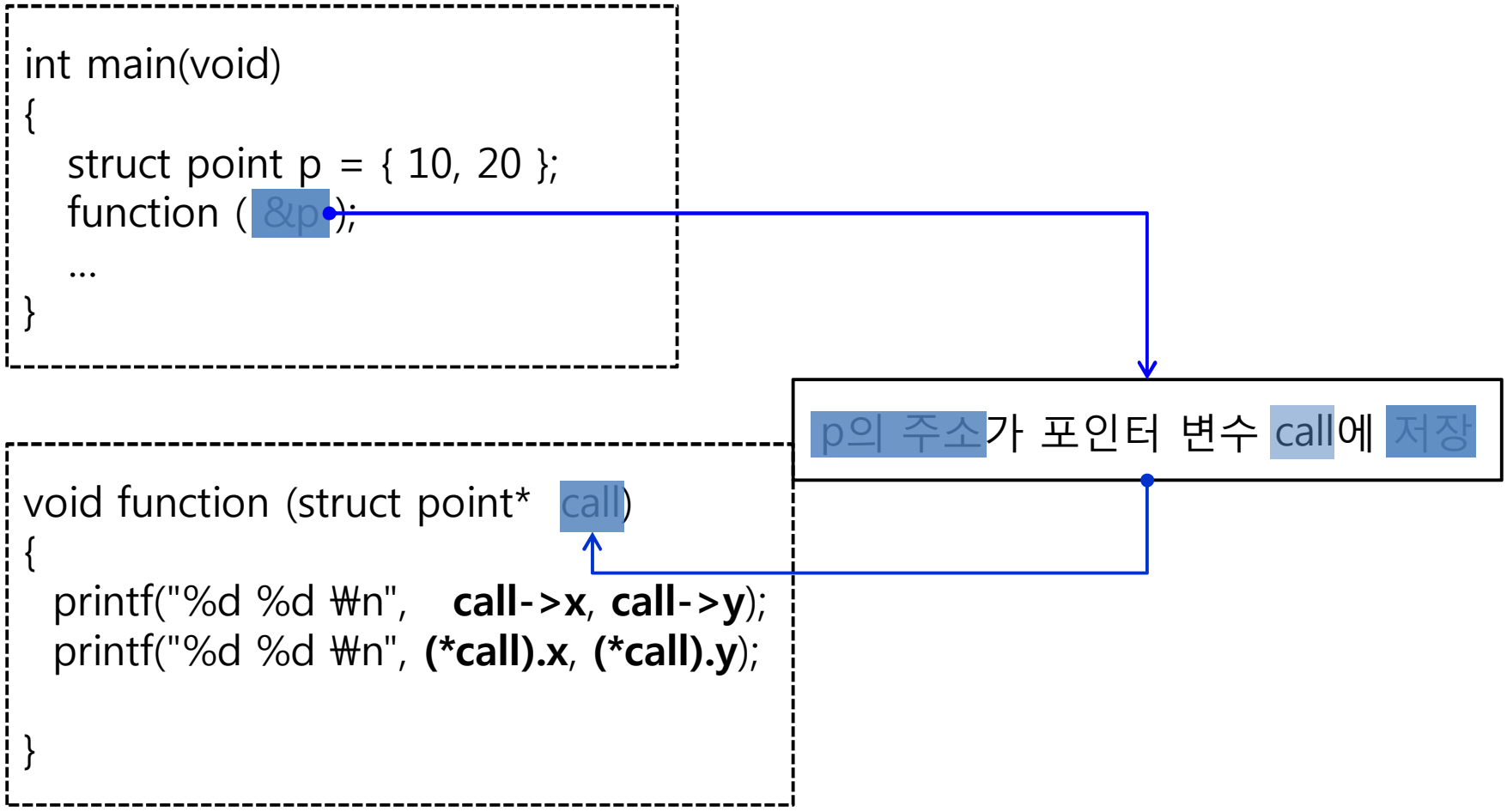
함수의 인자 : 구조체 포인터 변수

```
void function(struct point* a)
{
  ...
}
```

1.5 구조체와 함수 (5/10)---[1-24.c 실습]

```
#include<stdio.h>
struct point
{
    int x;
    int y;
};
void function (struct point* call);  // 함수의 선언
int main(void)
{
    struct point p = {10, 20};
    function(&p);                    // 주소에 의한 호출(call by reference)

    return 0;
}

void function (struct point* call)  // 함수의 정의
{
    printf("%d %d ₩n",    call->x, call->y);
    printf("%d %d ₩n", (*call).x, (*call).y);
}
```

1.5 구조체와 함수 (6/10)---[1-24.c 분석]



1.4 구조체와 포인터

▶배울 내용

① 구조체를 함수의 인자로 전달하기
- 값에 의한 호출과 주소에 의한 호출

② 구조체를 함수의 반환형으로 전달하기
- 값 반환과 주소 반환

1.5 구조체와 함수 (7/10)

▶ 구조체의 값(value)을 반환(return)하는 함수

함수의 반환형: 구조체 값 반환

① 반환 형태

```
struct point function ( )
{
    struct point p={10, 20};
    return p;
}
```

② 구조체 변수 이름

1.5 구조체와 함수 (8/10)---[1-25.c 실습]

```
#include<stdio.h>
struct point
{
    int x;
    int y;
};
struct point function(void);        // 함수의 선언
int main(void)
{
    struct point p;
--> p = function();                 // 함수 호출
    printf("%d %d ₩n", p.x, p.y);
    return 0;
}
struct point function(void)         // 함수의 정의
{
    struct point call = {10, 20};
    return call;                    // 구조체 변수 call 반환
```

1.5 구조체와 함수 (9/10)

▶ 구조체 주소(reference)를 반환(return)하는 함수

함수의 반환형: 구조체 주소반환

① 반환 형태

```
struct point* function ( )
{
    static struct point p={10, 20};
    return &p;
}
```

② 구조체 변수의 주소

1.5 구조체와 함수 (10/10)---[1-26.c 실습]

```
#include<stdio.h>
struct point
{
    int x;
    int y;
};
struct point* function(void);        // 함수의 선언
int main(void)
{
    struct point* p;
    p = function();                  // 함수 호출
    printf("%d %d ₩n", p->x, p->y);
    printf("%d %d ₩n", (*p).x,(*p).y);
    return 0;
}
struct point* function(void)         // 함수의 정의
{
    static struct point* call = {10, 20};
    return &call;                    // 구조체 변수 call의 주소 반환
```

1.6 공용체와 열거형

1.6 공용체와 열거형

▶배울 내용

① 공용체

② 열거형

1.6 공용체와 열거형 (1/7)

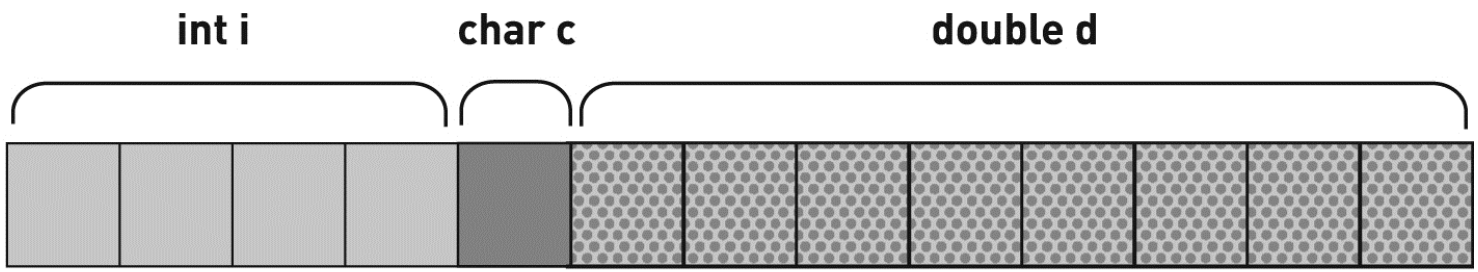
▶ 공용체

- ✓ 멤버 변수들 중 가장 큰 **메모리 공간을 '공유'**해서 사용
- ✓ **'union'** 키워드 사용
- ✓ **공용체 멤버 변수의 선언:** 구조체와 동일
- ✓ **공용체 변수의 선언:** 구조체와 동일
- ✓ **멤버 변수 접근:** 구조체와 동일

1.6 공용체와 열거형 (2/7)

▶ 구조체

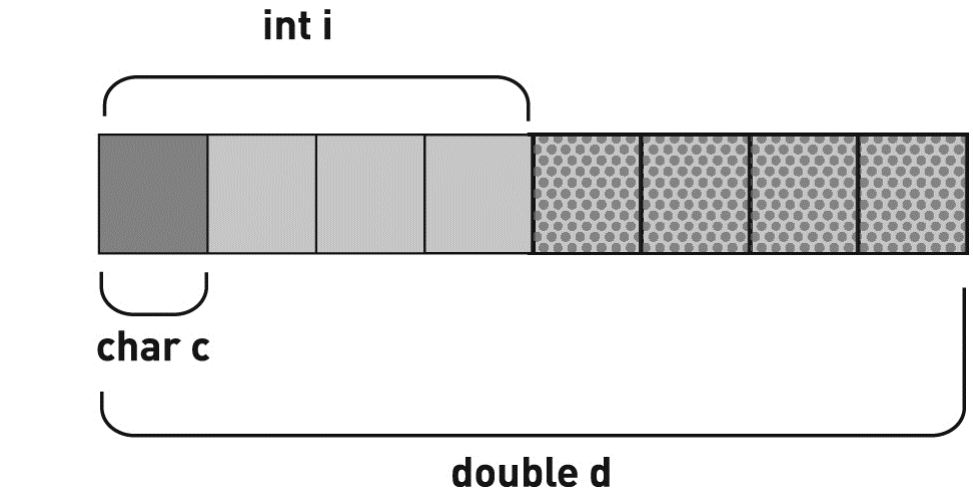
```
struct date
{
        int i;
        char c;
        double d;
};
```



1.6 공용체와 열거형 (3/7)

▶ 공용체

```
union date
{
        int i;
        char c;
        double d;
};
```



1.6 공용체와 열거형 (4/7)---[1-27.c 실습]

```
#include<stdio.h>
union point                 // 공용체 정의
{
    int x;
    int y;
};
struct student              // 구조체 정의
{
    int a;
    int b;
};

int main(void)
{
    printf("%d %d ₩n",sizeof(union point), sizeof(struct student) );
    return 0;
}
```

1.6 공용체와 열거형 (5/7)---[1-28.c 실습]

```
#include<stdio.h>

union point                 // 공용체정의
{
    int x;
    int y;
};

int main(void)
{
    union point p;          // 공용체변수선언
    p.x = 10;

    printf("%d %d ₩n", p.x, p.y);
    return 0;
}
```

1.4 구조체와 포인터

▶배울 내용

① 공용체

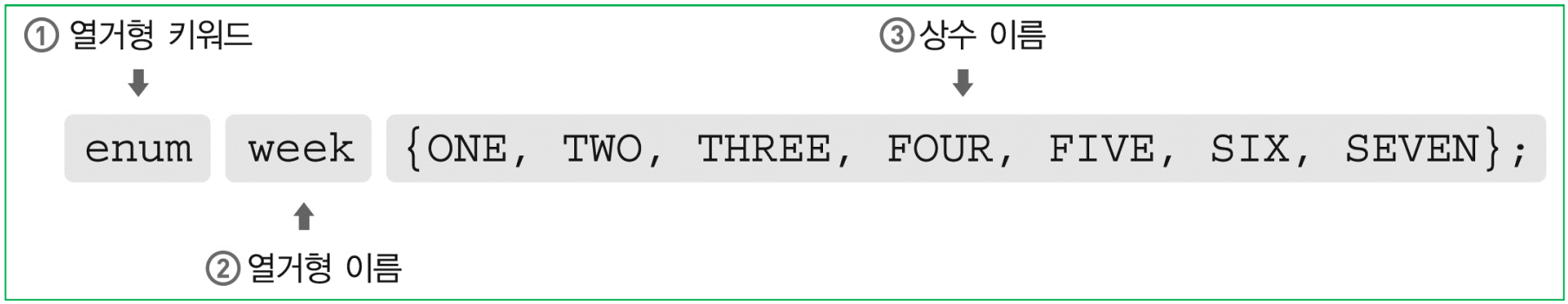
② 열거형

1.6 공용체와 열거형 (6/7)

▶ 열거형

- ✓ 변수가 갖는 값에 의미를 부여
- ✓ 프로그램의 가독성이 높아짐
- ✓ 컴파일러는 실제로 열거형 멤버들을 정수형 상수로 인식

▶ 정의 방법



- ✓ **열거형 키워드:** enum 키워드를 지정
- ✓ **열거형 이름:** 열거형을 대표하는 열거형 이름 지정
- ✓ **상수 이름:** 열거형 데이터로 사용할 상수 이름을 지정

1.6 공용체와 열거형 (7/7)---[1-28.c 실습(1/2)]

```
#include<stdio.h>
enum week {ONE, TWO, THREE, FOUR, FIVE, SIX, SEVEN};   // 열거형 정의
enum season {SPRING, SUMMER=2, FALL, WINTER};          // 열거형 정의

int main(void)
{
    enum week   p1, p2, p3;       // 열거형 변수 p1, p2, p3 선언
    enum season s1, s2, s3, s4;   // 열거형 변수 s1, s2, s3, s4 선언

    p1 = ONE;
    p2 = TWO;
    p3 = THREE;

    printf("%d %d %d ₩n", ONE, TWO, THREE );
    printf("%d %d %d ₩n", p1, p2, p3 );

    s1 = SPRING;
    s2 = SUMMER;
    s3 = FALL;
    s4 = WINTER;

    printf("%d %d %d %d ₩n", SPRING, SUMMER, FALL, WINTER );
    printf("%d %d %d %d ₩n", s1, s2, s3, s4 );
   return 0;
}
```

공부한 내용 떠올리기

- ▶ **구조체**의 정의, **구조체 변수**, 구조체 변수로 **멤버 변수의 접근**
- ▶ 구조체 **변수의 초기화**, 구조체 **변수의 복사**
- ▶ **중첩 구조체**, 중첩 구조체의 초기화
- ▶ typedef를 이용하여 사용자 **정의 자료형의 재정의 방법**
- ▶ **구조체 배열**과 **구조체 포인터**
- ▶ **구조체와 함수**, **공용체**와 **열거형**

곰이 가르쳐준 '동반자'의 의리(출처: 사랑과 지혜의 탈무드)