

기초프로그래밍

제15장 구조체와 공용체

Sangsoo Lim

CSAI

Dongguk University

차례

- 구조체란
- 구조체와 배열
- 구조체와 포인터
- 공용체와 열거형

구조체란

- 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;
```

구조체란

- 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

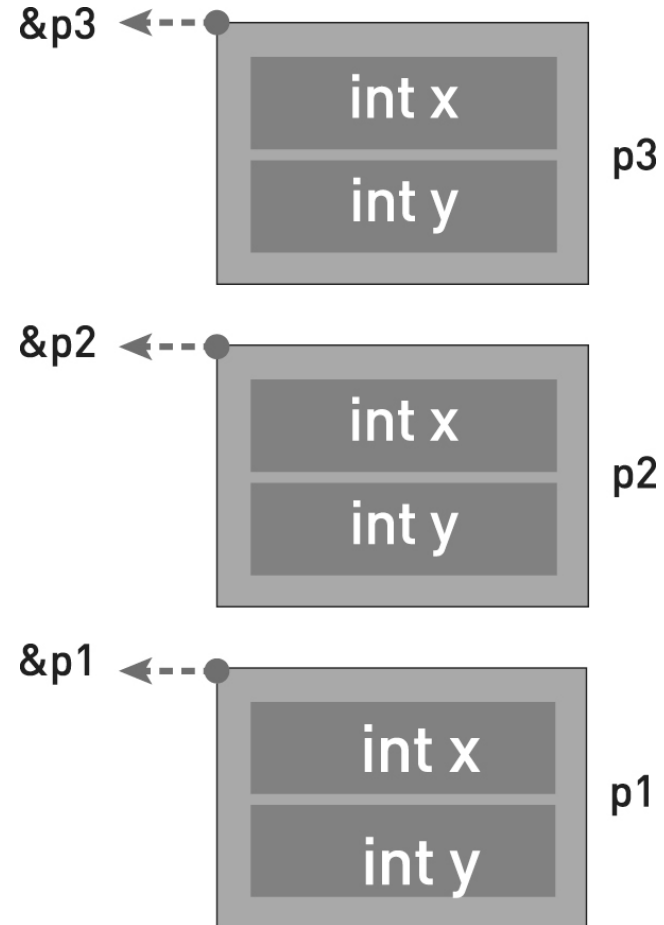
구조체란

- 구조체 정의, 구조체 변수 선언 – 동시에

```
#include <stdio.h>

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

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



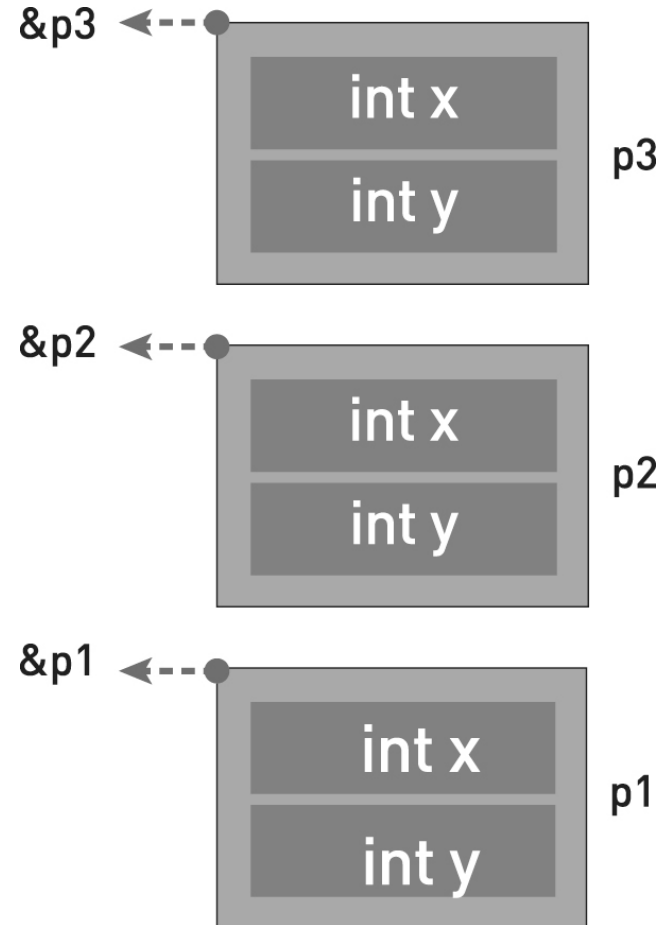
구조체란

- 구조체 정의, 구조체 변수 선언 – **따로**

```
#include <stdio.h>

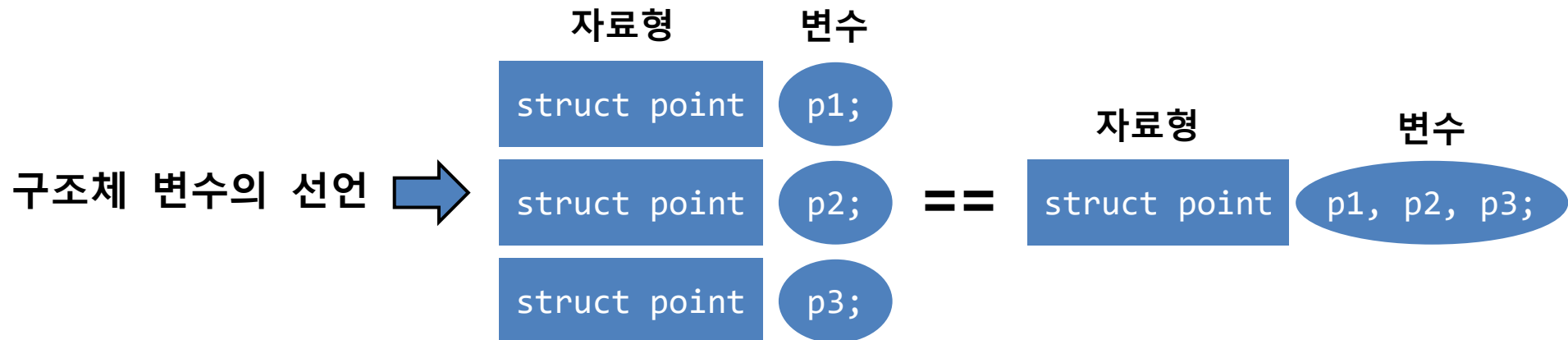
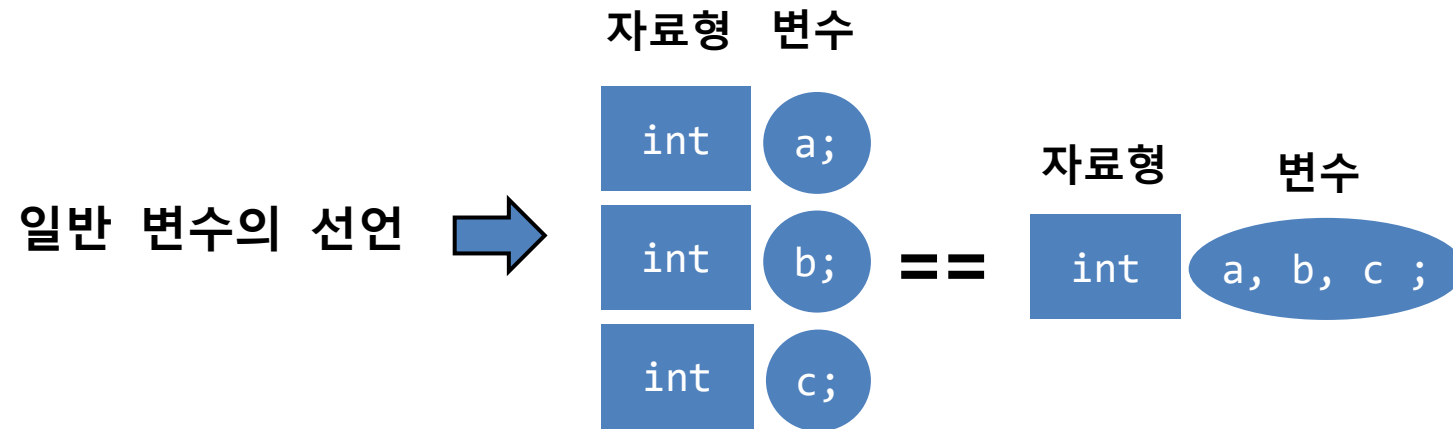
struct point
{
    int x;
    int y;
};

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



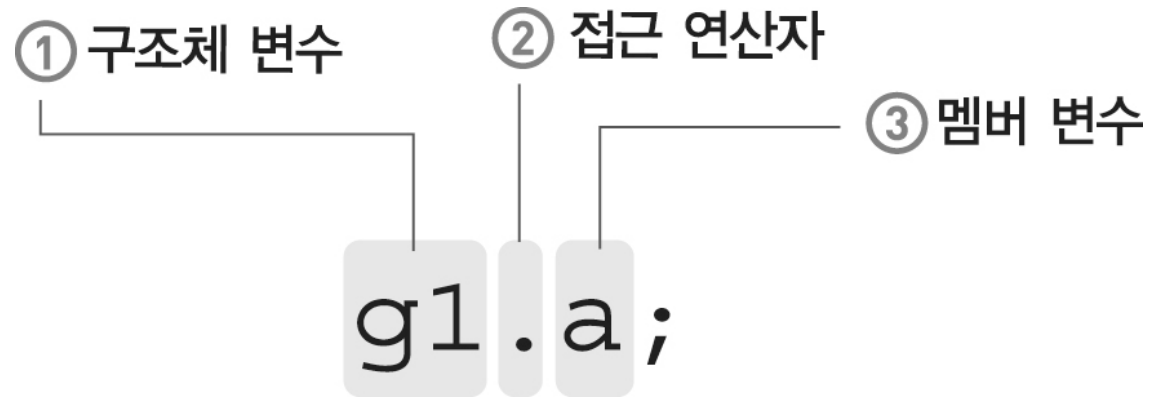
구조체란

- 일반 변수 vs. 구조체 변수



구조체란

- 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

구조체란

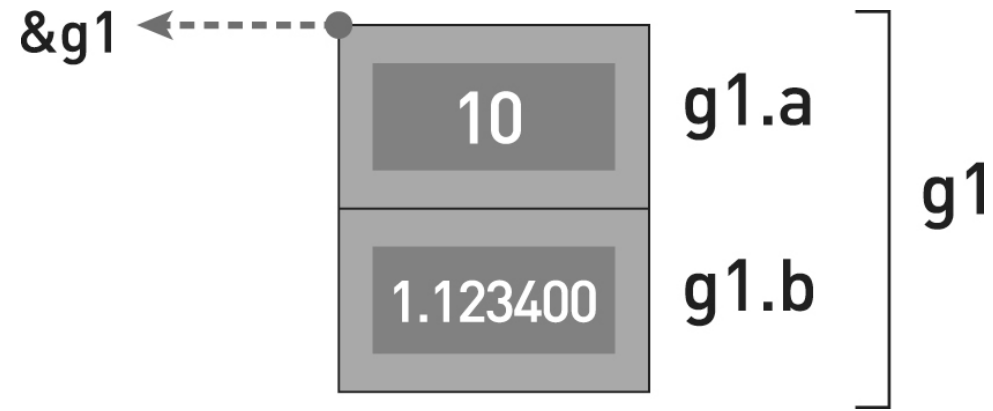
```
/* 15-1.c */
#include <stdio.h>
struct group                // 구조체 정의
{
    int a;
    double b;
};
int main(void)
{
    struct group g1;        // 구조체 변수 g1 선언

    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: %1f \n", g1.b);
    return 0;
}
```

구조체란

- 구조체 변수 **g1**의 메모리 구조



구조체란

```
/* 15-2.c */
#include <stdio.h>

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

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

    printf("정수와 실수를 입력하세요 (예: 3 4.5): ");
    if (scanf("%d %lf", &g1.a, &g1.b) != 2) { // 입력 오류 처리
        printf("입력이 잘못되었습니다. 프로그램을 종료합니다.\n");
        return 1; // 비정상 종료
    }

    printf("g1.a의 값: %d\n", g1.a);
    printf("g1.b의 값: %.2lf\n", g1.b); // 출력 형식 정리 (소수점 2자리)

    return 0; // 정상 종료
}
```

구조체란

```
/* 15-3.c */
#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;
}
```

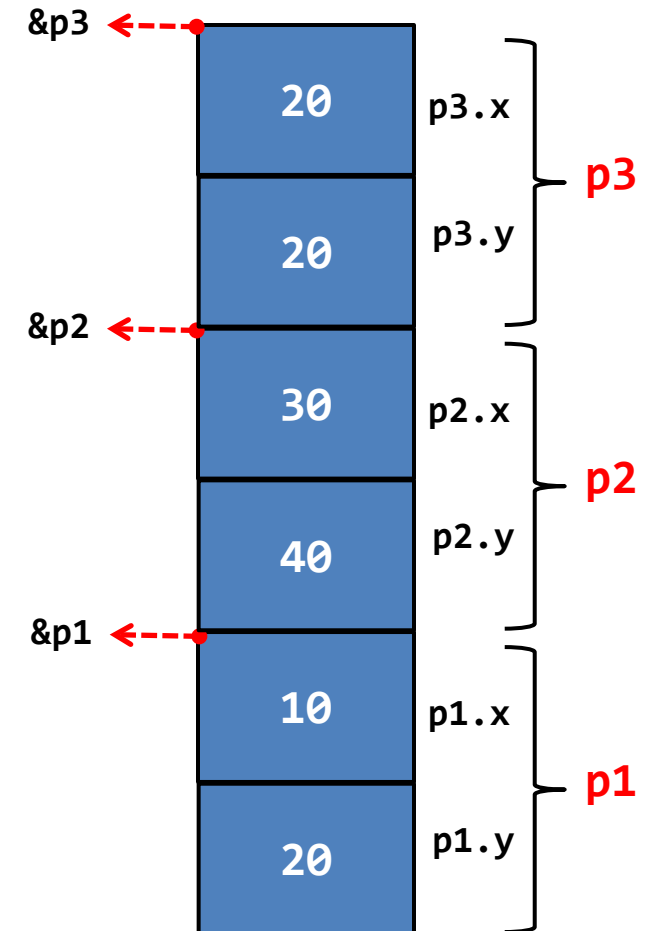
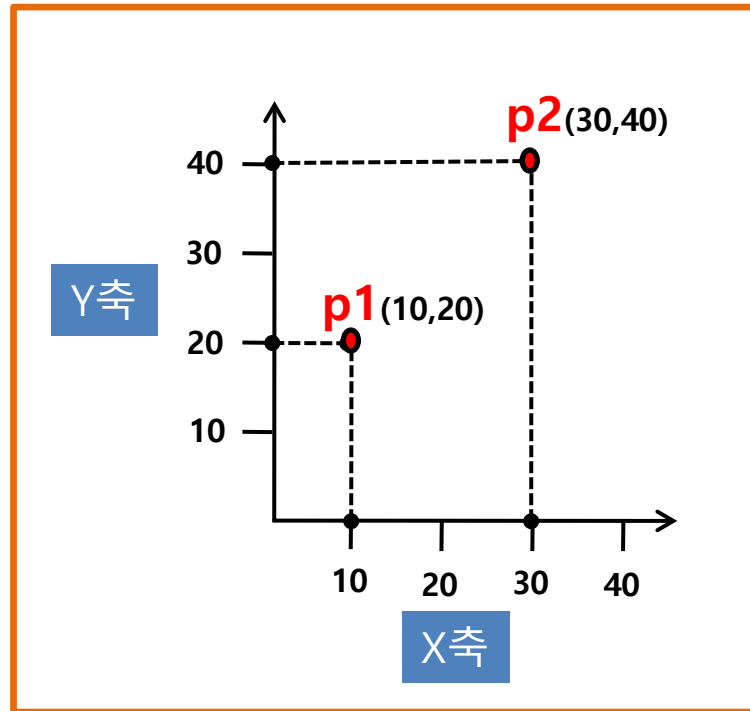
구조체란

```
/* 15-4.c */
#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;
}
```



구조체란

- 구조체 변수의 초기화

- 중괄호를 이용한 '구조체 변수'의 초기화 시 주의사항
 - 구조체 변수의 선언과 구조체 변수의 초기화를 따로 하면 에러가 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
```

구조체란

- 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
```

구조체란

```
/* 15-5.c */
#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;
}
```


구조체란

```
/* 15-6.c */
#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

중첩 구조체

```
/* 15-7.c */
#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;
}
```

중첩 구조체

```
/* 15-8.c */
#include <stdio.h>
struct score
{
    double math;
    double english;
    double total;
};

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



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

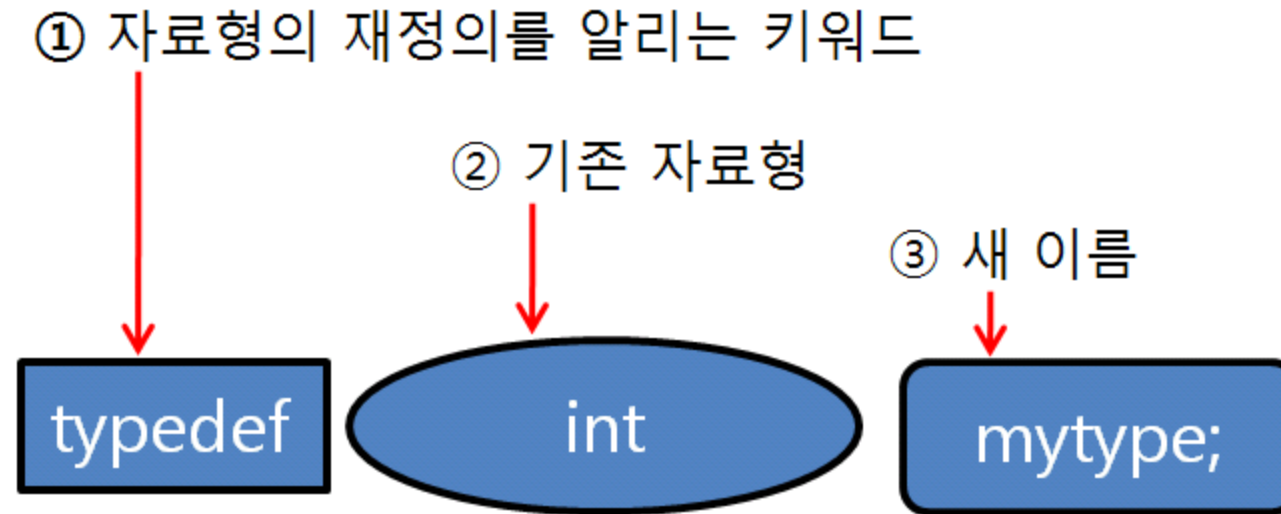
    stu.s.total=stu.s.math+stu.s.english;
    printf("학번: %d \n", stu.no);
    printf("총점: %lf \n", stu.s.total);

    return 0;
}
```

중첩 구조체의 초기화 방법 2가지

typedef를 이용한 구조체의 재정의

- typedef의 사용 방법



typedef를 이용한 구조체의 재정의

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;
```

typedef를 이용한 구조체의 재정의

```
/* 15-9.c */
#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;
}
```

구조체와 배열

```
/* 15-10.c */
#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
}
```

구조체와 배열

```
    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;
}
```


구조체와 배열

```
/* 15-11.c */
#include <stdio.h>
struct student
{
    char no[10];
    char name[20];
    double math;
    double english;
    double total;
};
```



```
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;
}
```

구조체와 배열

```
/* 15-13.c */
#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


구조체와 배열

```
/* 15-14.c */
#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



구조체와 배열

```
#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)

구조체와 포인터

- 멤버 변수로 포인터 사용하기

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

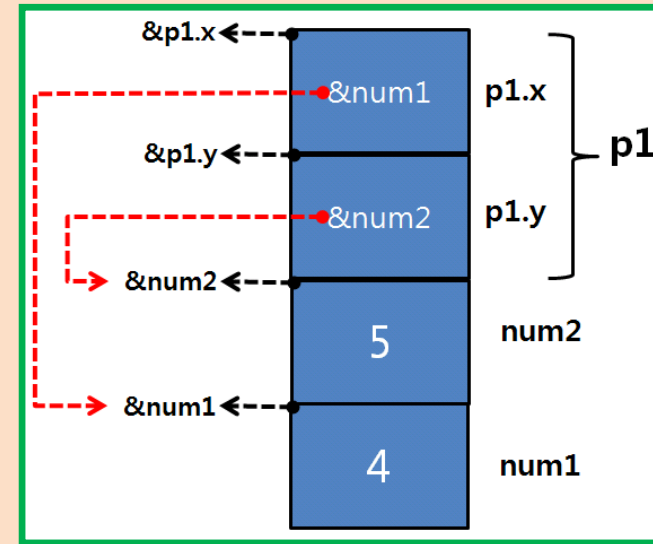
```
struct point
{
    int* x;    // 멤버 변수로 1차원 포인터 선언
    int** y;   // 멤버 변수로 2차원 포인터 선언
}
```

구조체와 포인터

```
/* 15-15.c */
#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.

구조체와 포인터

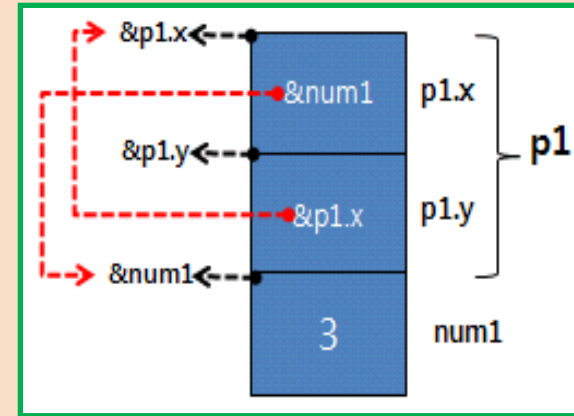
```
/* 15-16.c */
#include<stdio.h>
struct point
{
    int*   x;   // 1차원 포인터 멤버변수
    int**  y;   // 2차원 포인터 멤버변수
};

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);

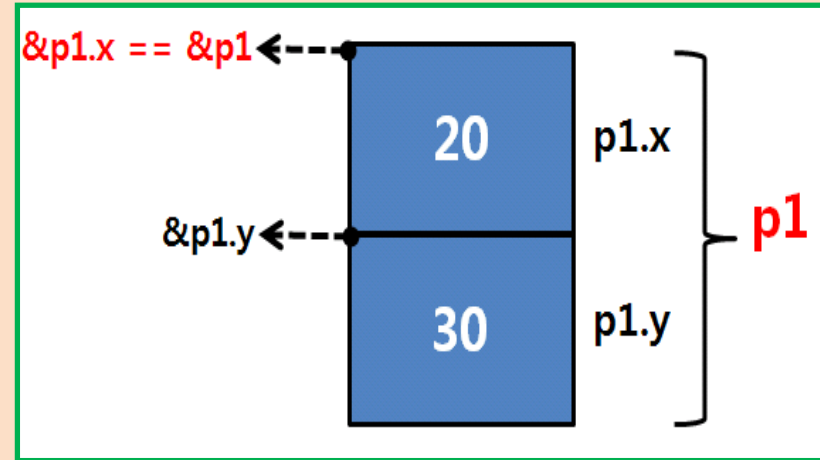
    return 0;
}
```



구조체와 포인터

```
/* 15-17.c */
#include<stdio.h>
struct point
{
    int x;
    int y;
};
int main(void)
{
    struct point p1={20,30};
    printf("구조체 변수 p1의 주소: %p \n", &p1);
    printf("멤버 변수 p1.x의 주소: %p \n", &p1.x);

    return 0;
}
```

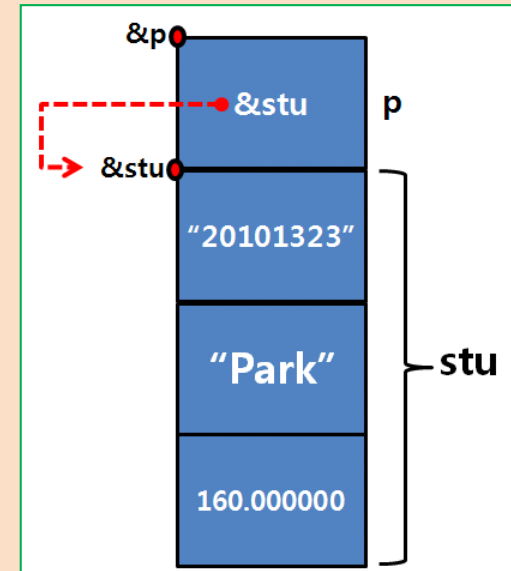


구조체와 포인터

```
/* 15-18.c */
#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);

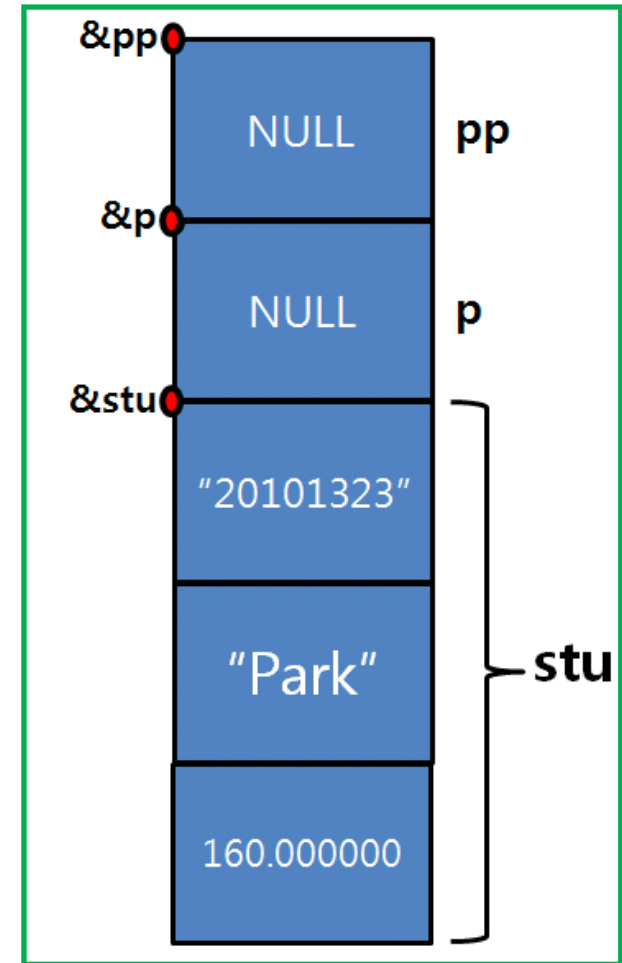
    return 0;
}
```



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

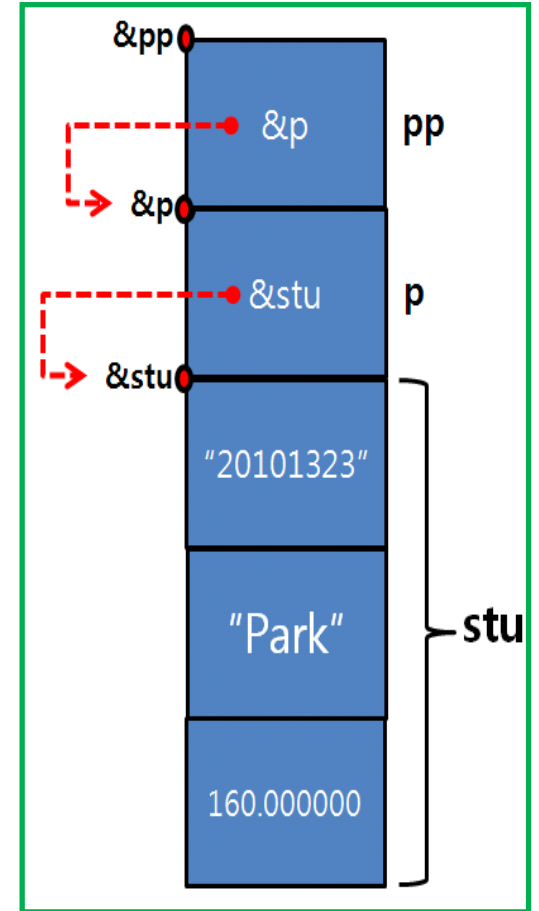
구조체와 포인터

```
/* 15-19.c */
#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;
```



구조체와 포인터

```
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;  
}
```



구조체와 포인터

자기 참조 구조체

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

구조체 내에서 자기 구조체 참조



외부 참조 구조체

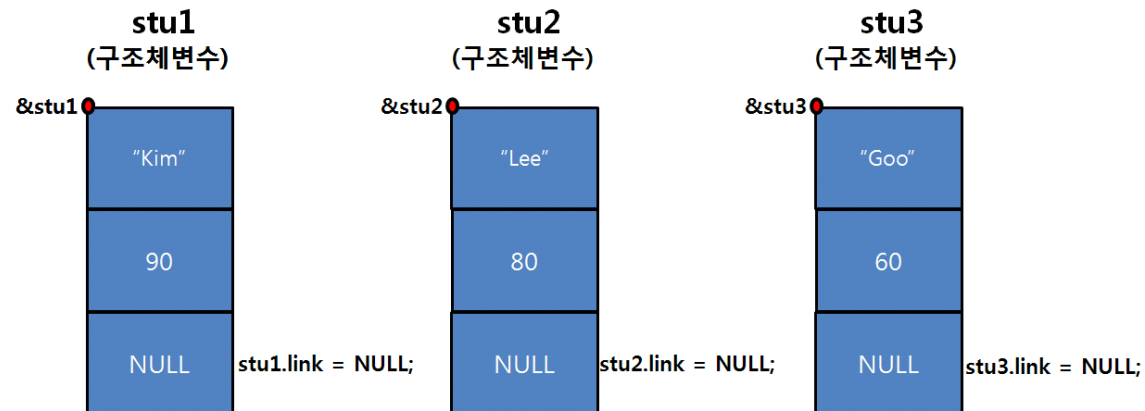
```
struct student
{
    char name[20];
    int age;
    struct score* p;
};
```

구조체 내에서 외부 구조체 참조



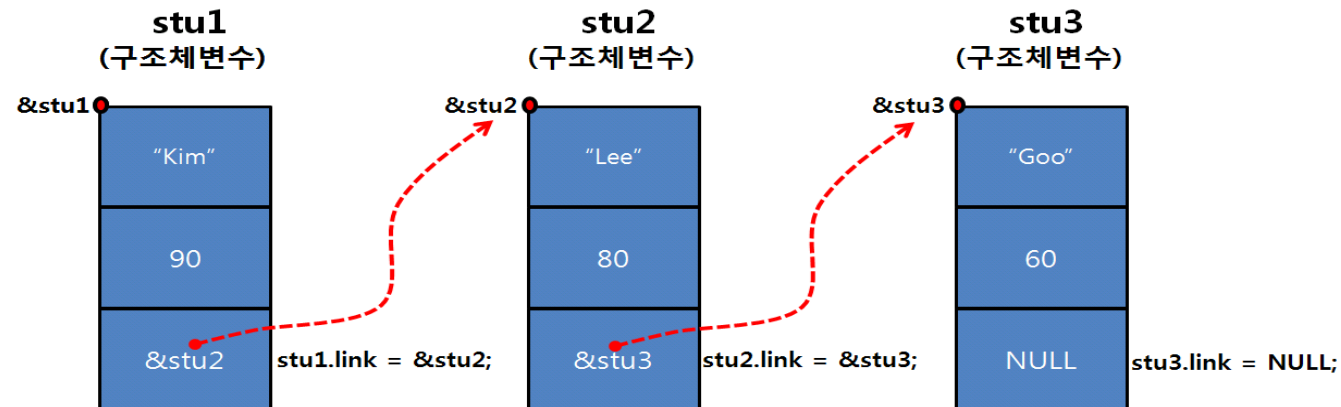
구조체와 포인터

```
/* 15-20.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};
}
```



구조체와 포인터

```
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;  
}
```



구조체와 함수

- 값에 의한 호출(Call by value)

함수의 인자 : 정수형 변수

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

함수의 인자 : 구조체형 변수

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

구조체와 함수

```
/* 15-23.c */
#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
}
```


구조체와 함수

- 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)
{
    ...
}
```

구조체와 함수

```
/* 15-24.c */
#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);
}
```

구조체와 함수

- 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

구조체와 함수


```
/* 15-25.c */
#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
}
```

구조체와 함수

- 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

구조체와 함수

```
/* 15-26.c */
#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
}
```

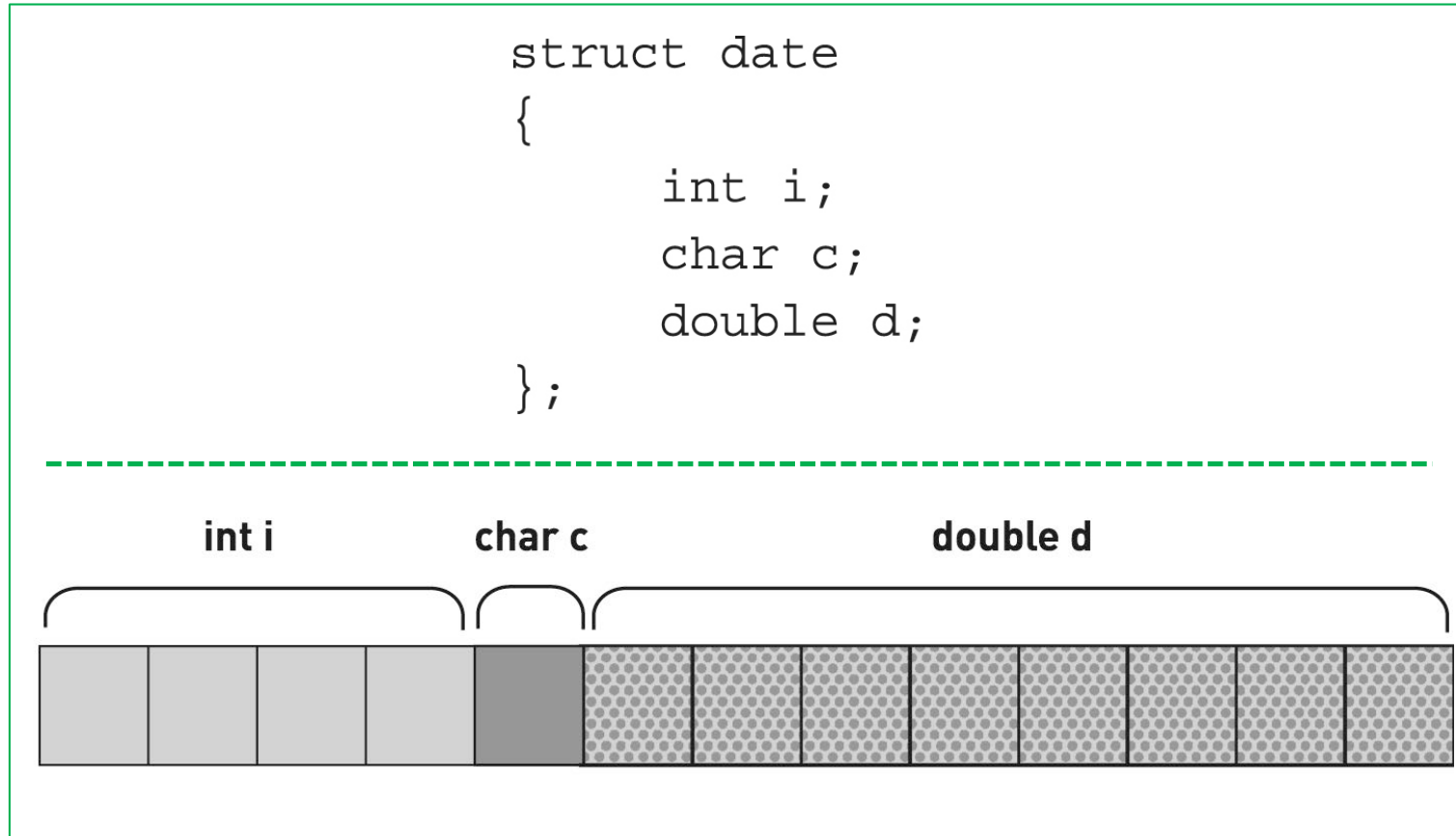
공용체와 열거형

- 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

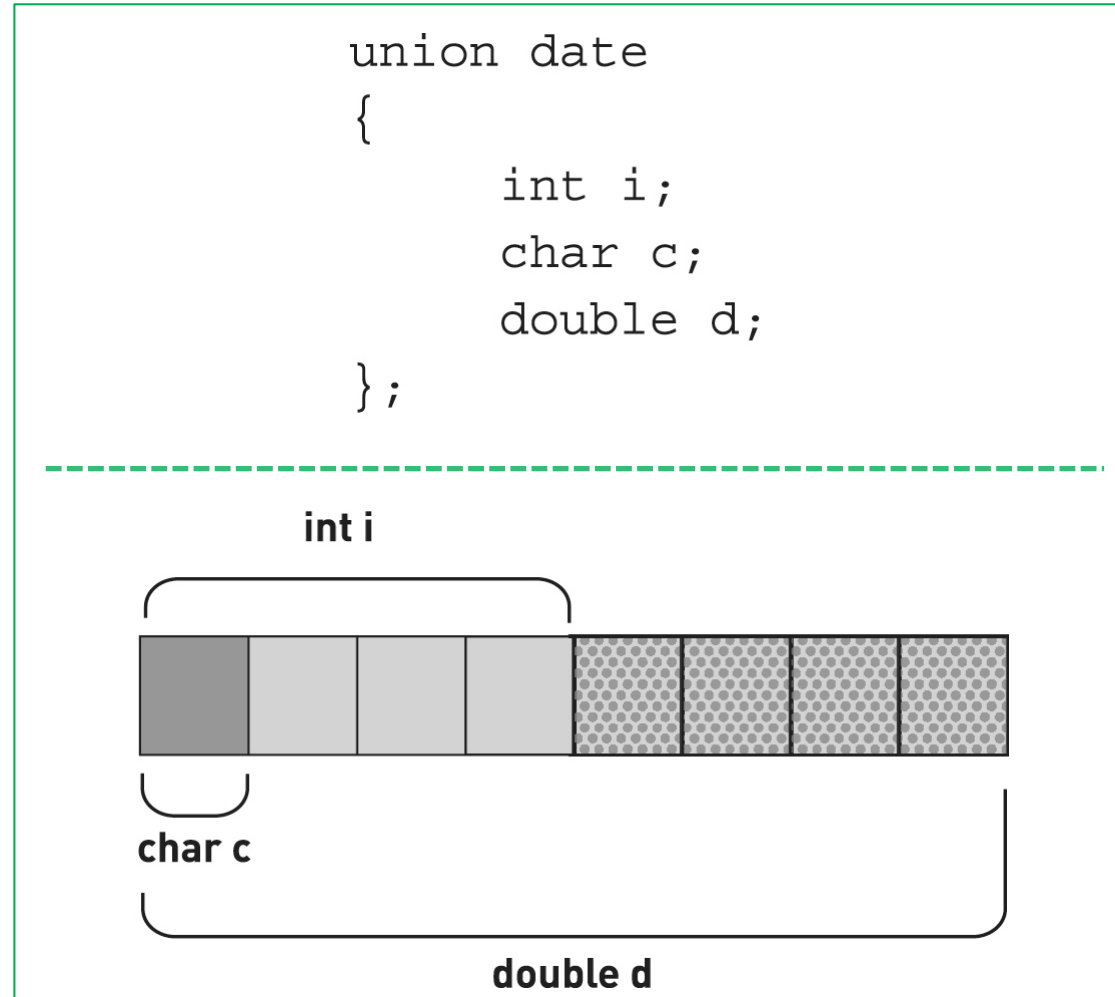
공용체와 열거형

- 구조체



공용체와 열거형

- 공용체



공용체와 열거형

```
/* 15-27.c */
#include<stdio.h>
union point                // 공용체 정의
{
    int x;
    int y;
};
struct student            // 구조체 정의
{
    int a;
    int b;
};

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

공용체와 열거형

```
/* 15-28.c */
#include<stdio.h>

union point                // 공용체 정의
{
    int x;
    int y;
};

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

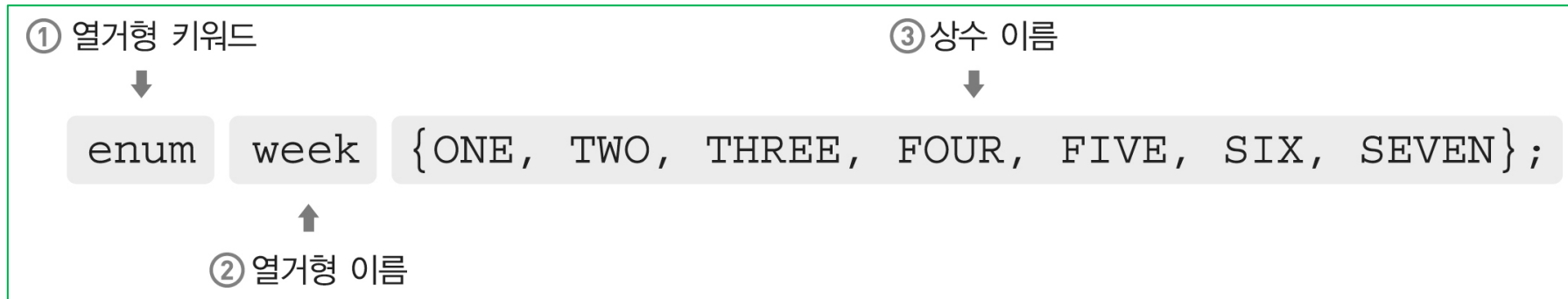
    printf("%d %d \n", p.x, p.y);
    return 0;
}
```

공용체와 열거형

• 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

공용체와 열거형

```
/* 15-29.c */
#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;
}
```

Summary

- 구조체의 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 배열과 구조체 포인터
- 구조체와 함수, 공용체와 열거형