

컴퓨터 프로그래밍 및 실습

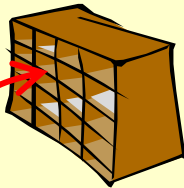
강의자료 4

- 구조체 (struct)

배열과 함수

- 함수의 인자가 배열인 경우 실제인자 배열이 (사본이 아닌) 원본이 참조된다.

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

    get_average(, int n);

    ...
}
```

```
int get_average(int score[], int n)
{
    ...

    sum += score[i];

    ...
}
```

구조체의 필요성

- 학생에 대한 다음 데이터를 저장하는 방법

학번: 20140001 (정수)

이름: 홍길동 (문자열)

학점: 4.25 (실수)

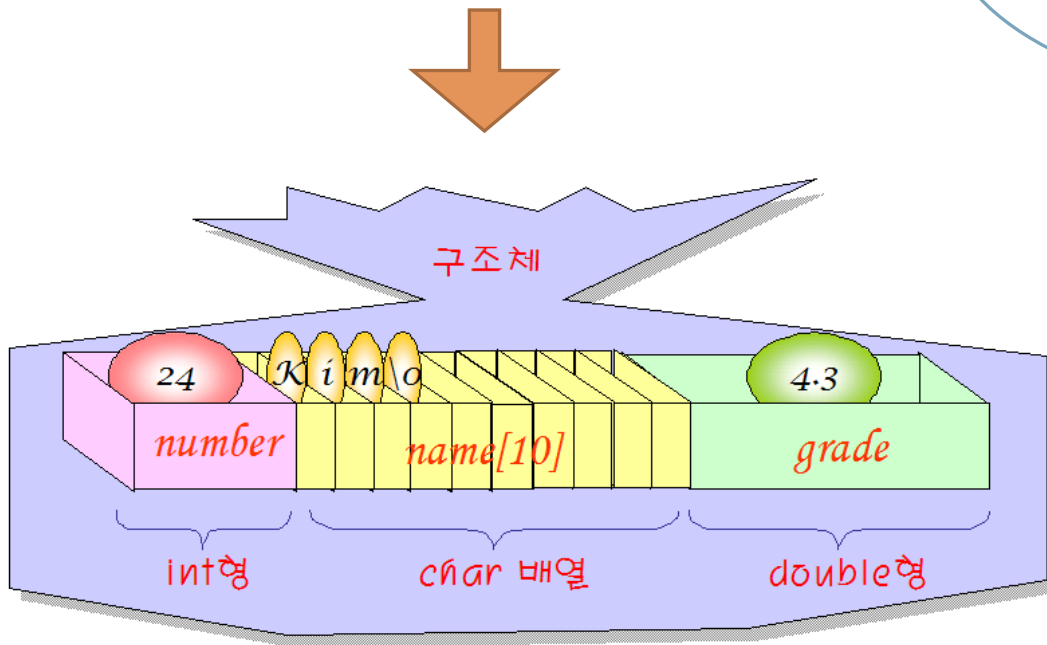
- `int no;`
`char name[10];`
`float grade;`

- 구조체를 이용하면 위의 데이터를 하나로 묶을 수 있다

구조체의 필요성

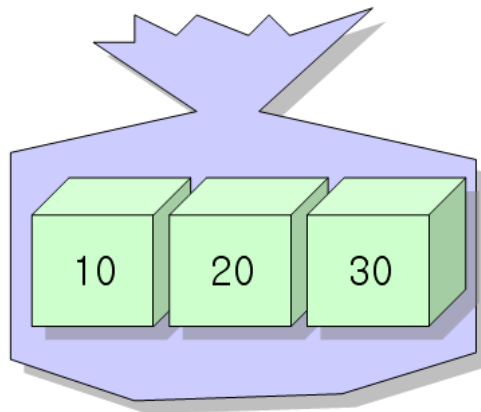
```
int number;  
char name[10];  
double grade;
```

구조체를 사용하면
변수들을 하나로
묶을 수 있다.



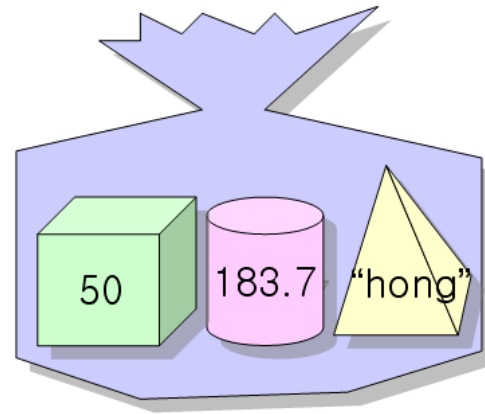
구조체와 배열

- 구조체 vs 배열



배열

같은 타입의 원소들 모임



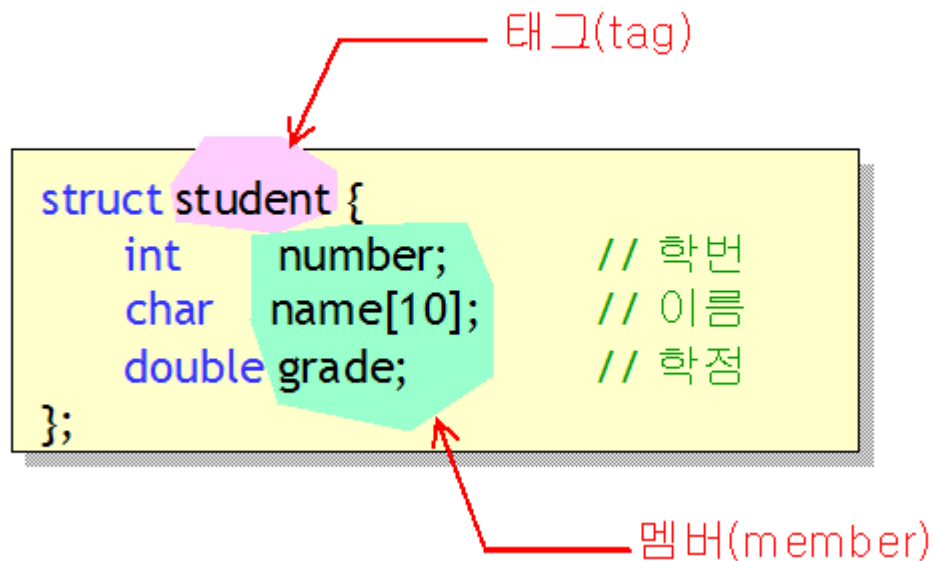
구조체

다른 타입의 원소들 모임

구조체 선언

- 구조체 선언 형식

```
struct 태그(구조체 이름) {  
    자료형      멤버1;  
    자료형      멤버2;  
    ...  
};
```



구조체 선언의 예

// x값과 y값으로 이루어지는 화면의 좌표

```
struct point {  
    int x;           // x 좌표  
    int y;           // y 좌표  
};
```

// 복소수

```
struct complex {  
    double real;      // 실수부  
    double imag;      // 허수부  
};
```

// 날짜

```
struct date {  
    int month;  
    int day;  
    int year;  
};
```

// 사각형

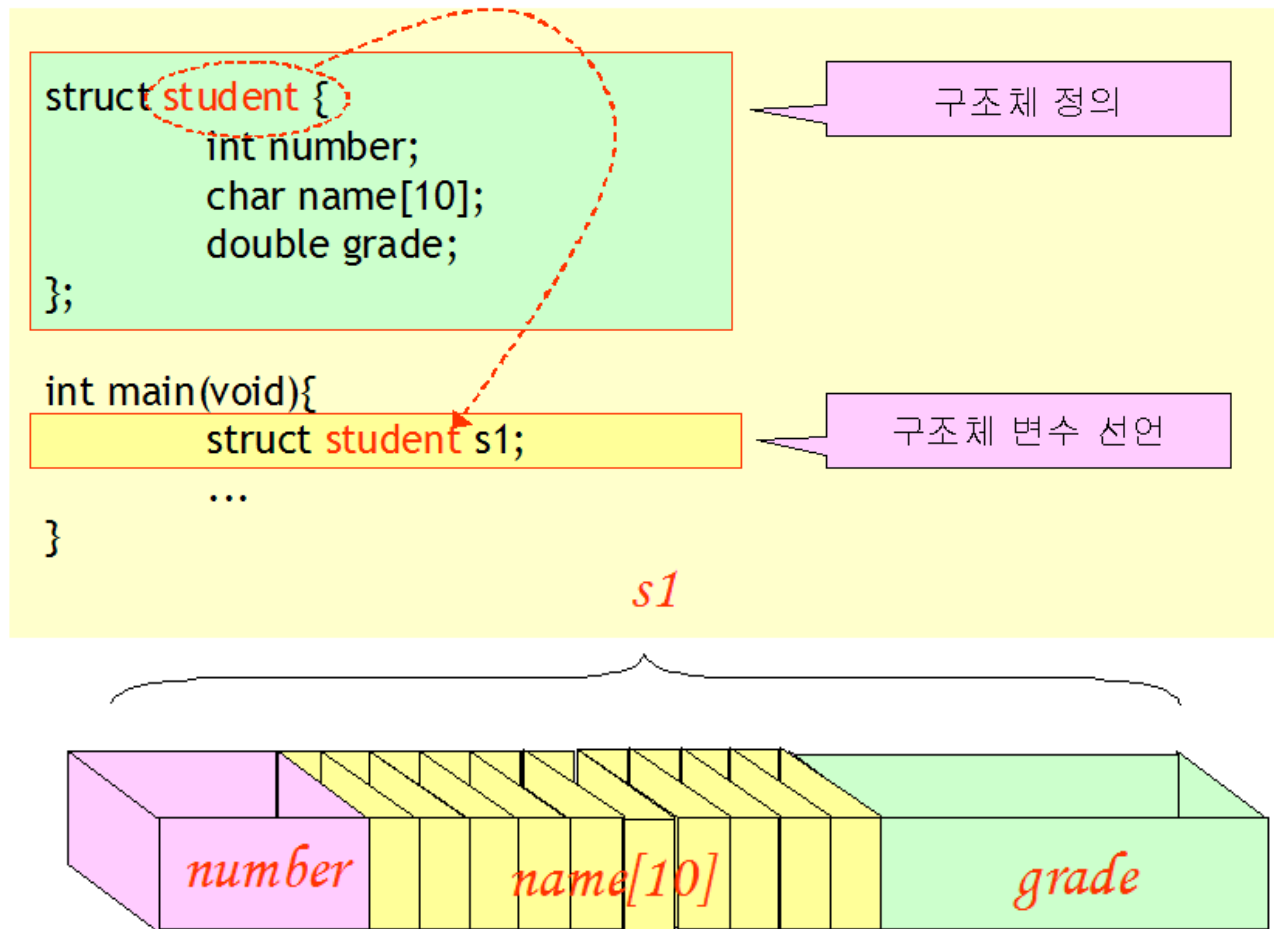
```
struct rect {  
    int x;  
    int y;  
    int width;  
    int grade;  
};
```

// 직원

```
struct employee {  
    char name[20];    // 이름  
    int age;           // 나이  
    int gender;        // 성별  
    int salary;        // 월급  
};
```

구조체 변수 선언

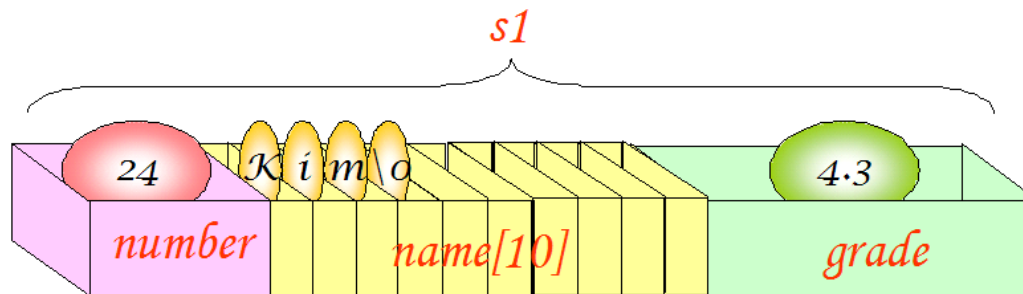
- 구조체 정의와 구조체 변수 선언은 다르다.



구조체의 초기화

- 중괄호를 이용하여 초기값을 나열한다.

```
struct student {  
    int number;  
    char name[10];  
    double grade;  
};  
struct student s1 = { 24, "Kim", 4.3 };
```



구조체 멤버 참조

- 구조체 멤버를 참조하려면 다음과 같이 . 연산자를 사용한다.

```
s1.number = 26;           // 정수 멤버  
strcpy(s1.name, "Kim");   // 문자열 멤버  
s1.grade = 4.3;           // 실수 멤버
```

예제 #1

...

```
struct student {  
    int number;  
    char name[10];  
    double grade;  
};
```

구조체 선언

```
int main(void)
```

```
{
```

```
    struct student s;
```

구조체 변수 선언

```
    s.number = 20070001;  
    strcpy(s.name, "홍길동");  
    s.grade = 4.3;
```

구조체 멤버 참조

```
    printf("학번: %d\n", s.number);  
    printf("이름: %s\n", s.name);  
    printf("학점: %f\n", s.grade);  
    return 0;
```

```
}
```

학번: 20070001

이름: 홍길동

학점: 4.300000

예제 #2

```
struct student {  
    int number;  
    char name[10];  
    double grade;  
};
```

구조체 선언

학번을 입력하시오: 20070001
이름을 입력하시오: 홍길동
학점을 입력하시오(실수): 4.3
학번: 20070001
이름: 홍길동
학점: 4.300000

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

```
    struct student s;
```

구조체 변수 선언

```
    printf("학번을 입력하시오: ");  
    scanf("%d", &s.number);
```

구조체 멤버의 주소 전달

```
    printf("이름을 입력하시오: ");  
    scanf("%s", s.name);
```

```
    printf("학점을 입력하시오(실수): ");  
    scanf("%lf", &s.grade);
```

```
    printf("학번: %d\n", s.number);  
    printf("이름: %s\n", s.name);  
    printf("학점: %f\n", s.grade);  
    return 0;
```

```
}
```

예제 #3

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

int main(void)
{
    struct point p1, p2;
    int xdiff, ydiff;
    double dist;

    printf("점의 좌표를 입력하시오(x y): ");
    scanf("%d %d", &p1.x, &p1.y);

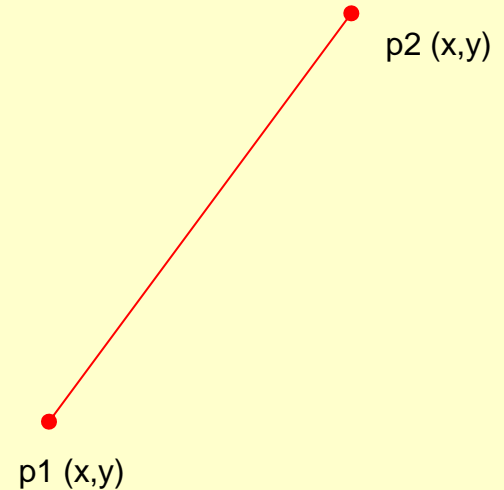
    printf("점의 좌표를 입력하시오(x y): ");
    scanf("%d %d", &p2.x, &p2.y);

    xdiff = p1.x - p2.x;
    ydiff = p1.y - p2.y;

    dist = sqrt(xdiff * xdiff + ydiff * ydiff);

    printf("두 점사이의 거리는 %f입니다.\n", dist);
    return 0;
}
```

점의 좌표를 입력하시오(x y): 10 10
점의 좌표를 입력하시오(x y): 20 20
두 점사이의 거리는 14.142136입니다.



예제 4

```
#include <stdio.h>
```

```
struct point {  
    int x;  
    int y;  
};
```

```
struct rect {  
    struct point p1;  
    struct point p2;  
};
```

```
int main(void)  
{  
    struct rect r;  
    int w, h, area, peri;
```



예제 4 (계속)

```
printf("왼쪽 상단의 좌표를 입력하시오: ");
scanf("%d %d", &r.p1.x, &r.p1.y);

printf("오른쪽 상단의 좌표를 입력하시오: ");
scanf("%d %d", &r.p2.x, &r.p2.y);

w = r.p2.x - r.p1.x;
h = r.p2.y - r.p1.y;

area = w * h;
peri = 2 * w + 2 * h;
printf("면적은 %d이고 둘레는 %d입니다.\n", area, peri);

return 0;
}
```



왼쪽 상단의 좌표를 입력하시오: 1 1
오른쪽 상단의 좌표를 입력하시오: 6 6
면적은 25이고 둘레는 20입니다.

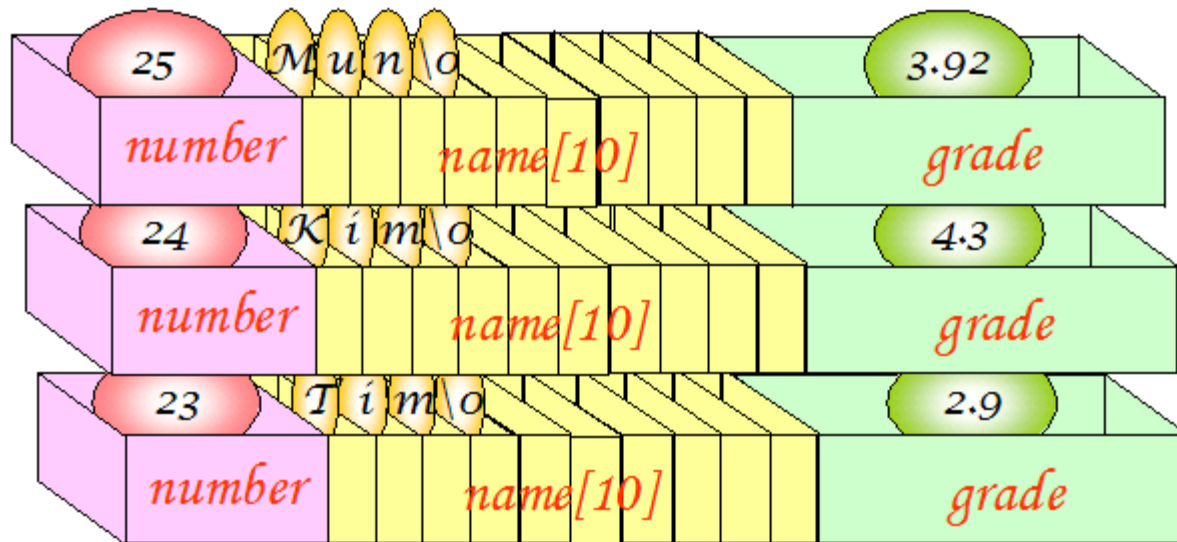
구조체 변수의 대입과 비교

- 같은 구조체 변수끼리 대입은 가능하지만 비교는 불가능하다.

```
struct point {  
    int x;  
    int y;  
};  
  
int main(void)  
{  
    struct point p1 = {10, 20};  
    struct point p2 = {30, 40};  
  
    p2 = p1; // 대입 가능  
  
    if( p1 == p2 ) // 비교 -> 컴파일 오류!!  
        printf("p1와 p2이 같습니다.")  
  
    if( (p1.x == p2.x) && (p1.y == p2.y) ) // 올바른 비교  
        printf("p1와 p2이 같습니다.")  
}
```


구조체 배열

- 구조체를 여러 개 모은 것



구조체 배열

- 구조체 배열의 선언

```
struct student {  
    int number;  
    char name[20];  
    double grade;  
};  
  
int main(void)  
{  
    struct student list[100]; // 구조체의 배열 선언  
  
    list[2].number = 27;  
    strcpy(list[2].name, "홍길동");  
    list[2].grade = 178.0;  
}
```

구조체 배열 예제

```
#define SIZE 3

struct student {
    int number;
    char name[20];
    double grade;
};

int main(void)
{
    struct student list[SIZE];
    int i;

    for(i = 0; i < SIZE; i++)
    {
        printf("학번을 입력하시오: ");
        scanf("%d", &list[i].number);
        printf("이름을 입력하시오: ");
        scanf("%s", list[i].name);
        printf("학점을 입력하시오(실수): ");
        scanf("%lf", &list[i].grade);
    }

    for(i = 0; i < SIZE; i++)
        printf("학번: %d, 이름: %s, 학점: %f\n", list[i].number, list[i].name, list[i].grade);
    return 0;
}
```

학번을 입력하시오: 20070001

이름을 입력하시오: 홍길동

학점을 입력하시오(실수): 4.3

학번을 입력하시오: 20070002

이름을 입력하시오: 김유신

학점을 입력하시오(실수): 3.92

학번을 입력하시오: 20070003

이름을 입력하시오: 이성계

학점을 입력하시오(실수): 2.87

학번: 20070001, 이름: 홍길동, 학점: 4.300000

학번: 20070002, 이름: 김유신, 학점: 3.920000

학번: 20070003, 이름: 이성계, 학점: 2.870000

구조체와 함수

- 구조체를 함수의 인수로 전달하는 경우
 - 구조체의 복사본이 함수로 전달되게 된다.
 - 만약 구조체의 크기가 크면 그만큼 시간과 메모리가 소요된다.

```
int equal(struct student s1, struct student s2)
{
    if( strcmp(s1.name, s2.name) == 0 )
        return 1;
    else
        return 0;
}
```

구조체를 반환하는 경우

- 복사본이 반환된다.

```
struct student make_student(void)
{
    struct student s;

    printf("나이:");
    scanf("%d", &s.age);
    printf("이름:");
    scanf("%s", s.name);
    printf("키:");
    scanf("%f", &s.grade);

    return s;
}
```

구조체 s의
복사본이 반환된다.



예제

```
#include <stdio.h>

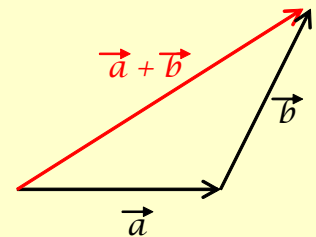
struct vector {
    float x;
    float y;
};

struct vector get_vector_sum(struct vector a, struct vector b);

int main(void)
{
    struct vector a = { 2.0, 3.0 };
    struct vector b = { 5.0, 6.0 };
    struct vector sum;

    sum = get_vector_sum(a, b);
    printf("벡터의 합은 (%f, %f)입니다.\n", sum.x, sum.y);

    return 0;
}
```

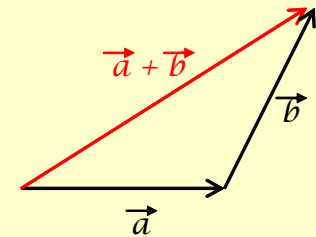


예제

```
struct vector get_vector_sum(struct vector a, struct vector b)
{
    struct vector result;

    result.x = a.x + b.x;
    result.y = a.y + b.y;

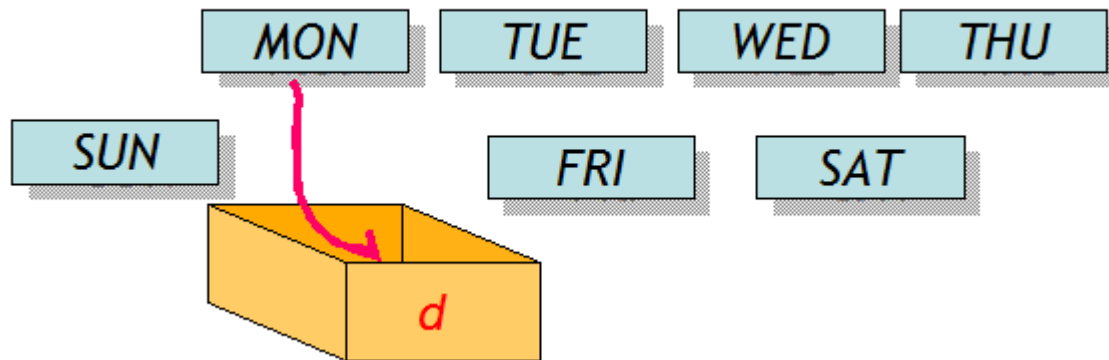
    return result;
}
```



벡터의 합은 (7.000000, 9.000000)입니다.

열거형

- *열거형(enumeration)*이란 변수가 가질 수 있는 값들을 미리 열거해놓은 자료형
- (예) 요일을 저장하고 있는 변수는 { 일요일, 월요일, 화요일, 수요일, 목요일, 금요일, 토요일 } 중의 하나의 값만 가질 수 있다.



열거형의 선언

```
enum days { SUN, MON, TUE, WED, THU, FRI, SAT };
```

태그 이름

값들을 나열

열거형 변수 선언

```
enum days today;  
today = SUN; // OK!
```

열거형이 필요한 이유

- 다음과 같이 프로그램을 작성할 수 있다.
 - `int today;`
 - `today = 0; // 일요일`
 - `today = 1; // 월요일`
- 되도록 오류를 줄이고 가독성을 높여야 된다.
- 0보다는 SUN라는 기호상수가 더 바람직하다.
의미를 쉽게 알 수 있기 때문이다.
- `today`에 9와 같은 의미없는 값이 대입되지 않도록 미리 차단하는 것도 필요하다.

열거형의 예

```
enum colors { white, red, blue, green, black };
```

```
enum boolean { false, true };
```

```
enum levels { low, medium, high };
```

```
enum car_types { sedan, suv, sports_car, van, pickup, convertible };
```

예제

```
#include <stdio.h>

enum days { SUN, MON, TUE, WED, THU, FRI, SAT };
char *days_name[] = {
    "sunday", "monday", "tuesday", "wednesday", "thursday", "friday",
    "saturday" };

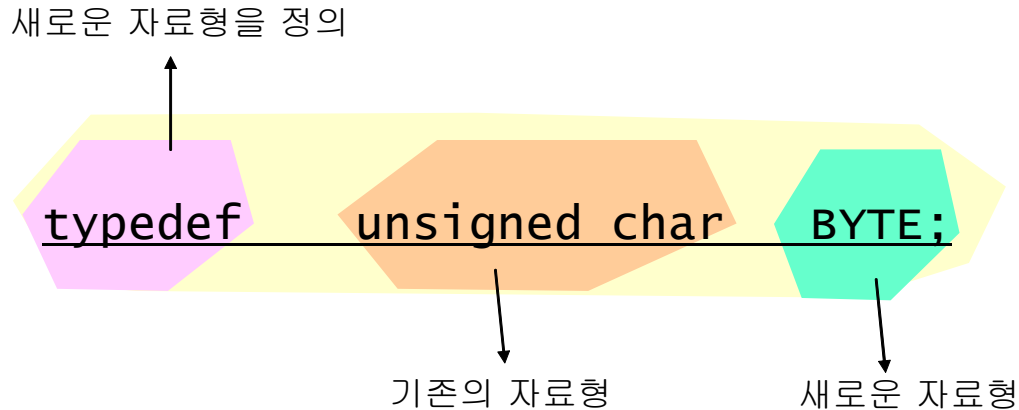
int main(void)
{
    enum days d;
    d = WED;
    printf("%d번째 요일은 %s입니다\n", d, days_name[d]);
    return 0;
}
```

3번째 요일은 wednesday입니다

typedef

- typedef은 새로운 자료형(type)을 정의(define)
- C의 기본 자료형을 확장시키는 역할

```
typedef    old_type    new_type;
```



typedef의 예

```
typedef unsigned char BYTE;  
BYTE index;           // unsigned int index;와 같다.  
  
typedef int INT32;  
typedef unsigned int UINT32;  
  
INT32 i;               // int i;와 같다.  
UINT32 k;              // unsigned int k;와 같다.
```

구조체로 새로운 타입 정의

- 구조체로 새로운 타입을 정의할 수 있다.

```
struct point {  
    int x;  
    int y;  
};  
typedef struct point POINT;  
POINT a, b;
```

예제

```
#include <stdio.h>

typedef struct point {
    int x;
    int y;
} POINT;

POINT translate(POINT p, POINT delta);

int main(void)
{
    POINT p = { 2, 3 };
    POINT delta = { 10, 10 };
    POINT result;

    result = translate(p, delta);
    printf("새로운 점의 좌표는(%d, %d)입니다.\n", result.x, result.y);

    return 0;
}
```


예제

```
POINT translate(POINT p, POINT delta)
{
    POINT new_p;

    new_p.x = p.x + delta.x;
    new_p.y = p.y + delta.y;

    return new_p;
}
```

새로운 점의 좌표는 (12, 13)입니다.

구조체 배열 연습

- 예: 10명의 학생의 자료를 입력 받아서 평점이 가장 높은 학생의 학번과 이름 및 평점을 출력

```
struct student {  
    int id; string name; int grade;  
};  
  
main()  
{  
    int i;  
    struct student s[10];           /* s는 구조체의 배열 */  
    for (i=0; i<10; i++)  
        cin >> s[i].id >> s[i].name >> s[i].dept;  
    ....  
}
```