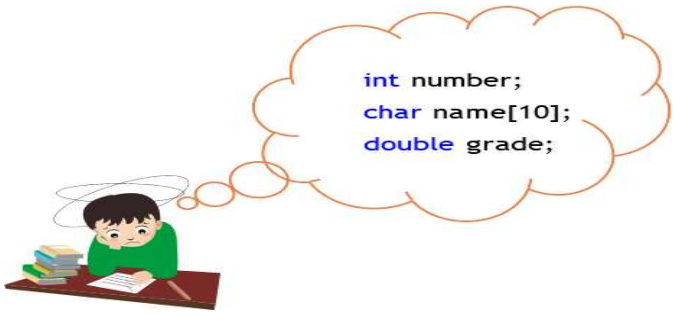
14. 구조체

■ 구조체란?

● 구조체 : 하나 이상의 변수(포인터 변수와 배열 포함)를 묶어서 새로운 자료형을 만들 수 있는 도구

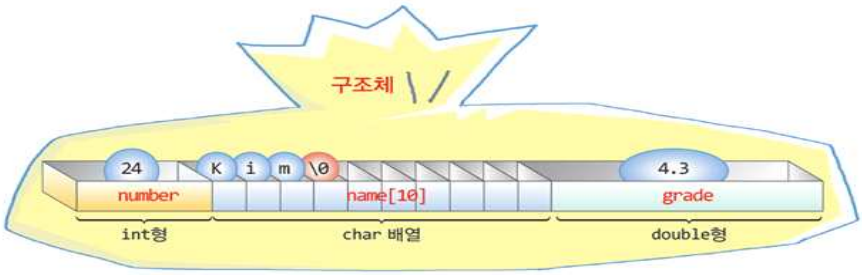
※ 배열은 동일한 종류의 자료형을 묶는 것이라면 구조체는 서로 다른 종류의 자료형을 묶을 수 있음.

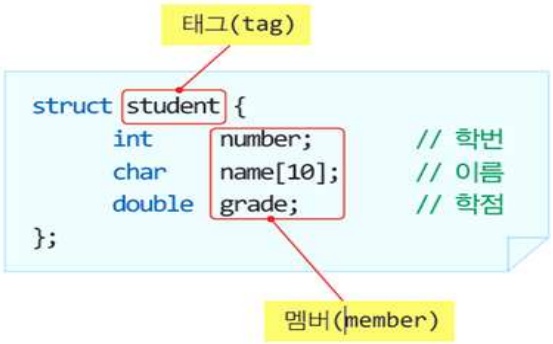
● 구조체를 기반으로 우리는 새로운 자료형을 정의할 수 있음




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

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





● 구조체는 struct, 태그(tag), 멤버(member)를 사용하여 위와 같은 형식으로 구조체를 정의함.

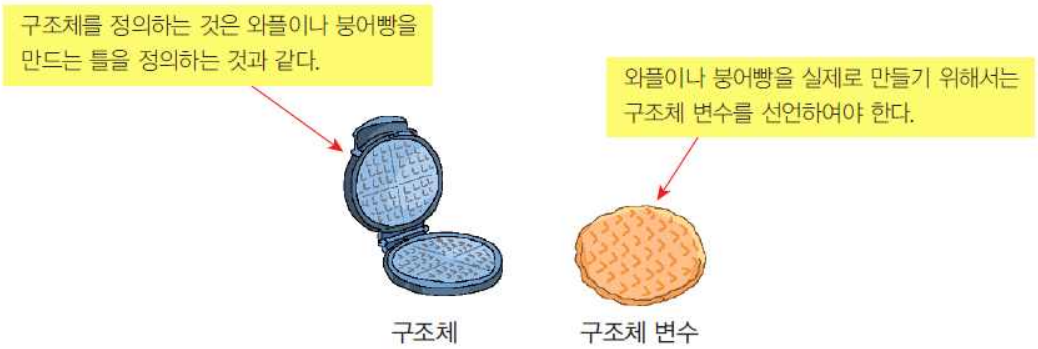
● struct는 구조체를 선언할 때 사용하는 키워드.

● 태그(tag)는 서로 다른 구조체를 구별하기 위해 구조체에 붙여지는 이름.

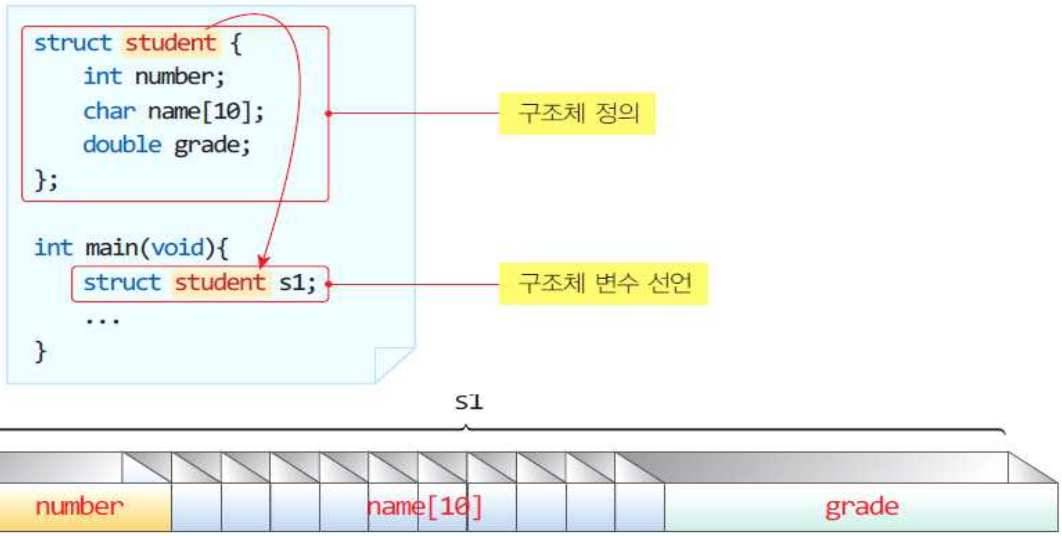
● 멤버(member)는 구조체에 포함되는 변수.

● 구조체의 정의가 끝나면 반드시 세미콜론을 붙여주어야 함. 구조체를 정의하는 것도 하나의 문장에 해당하기 때문임.

- 여기서 각별히 주의할 점은 **구조체 정의는 구조체 변수를 선언한 것이 아니라는 점**이다. 구조체 정의는 구조체 안에 어떤 변수들이 들어간다는 것만 말해주는 것임. 즉, 구조체의 형태(틀)만 정의한 것이고 아직 구조체를 이용하여 변수를 선언한 것이 아니라는 점을 유의해야 한다.



- 구조체 변수 선언



- 위의 문장은 student라는 구조체를 정의하고 s1이라는 구조체 변수를 선언한 것이다. s1이라는 변수 안에는 구조체의 멤버인 number, name, grade가 들어 있다.
- **구조체 변수가 선언된 후에야 s1에 실제 메모리 공간이 할당된다**. 그렇다면 s1이 차지하는 메모리 공간의 크기는 몇 바이트일까요?

4 + 10 + 8 = 22바이트

- 하나의 문장으로 여러 개의 구조체 변수를 선언할 수도 있다.

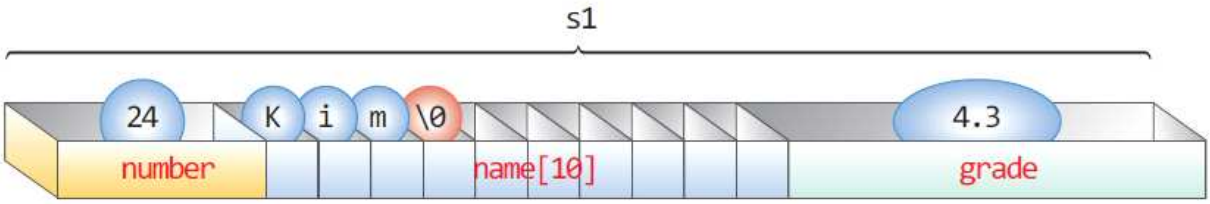
```
struct student s2 s3;
```

- 구조체 정의와 구조체 변수 선언을 동시에 할 수도 있다.

```
struct student {
        int number;
        char name[10];
        double grade;
} s1;  // s1은 구조체 student의 변수이다
```

● 구조체의 초기화

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



※ 구조체 정의와 구조체 변수 선언 모두 세미콜론으로 끝나는 것에 유의

● 구조체의 정의와 변수 선언을 동시에 하는 경우에 어떻게 초기화할까?

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

● 구조체 멤버 변수 참조

구조체 멤버를 참조하려면 멤버 연산자(.)를 이용하여야 한다.

```
s1.number = 20170001;     // 정수 멤버
strcpy(s1.name, "Kim");   // 문자열 멤버
s1.grade = 4.3;           // 실수 멤버
```

※ 첫 번째는 구조체 변수 s1의 멤버 변수인 number에 20170001을 대입하는 문장
두 번째는 멤버가 문자열이라면 멤버에 값을 대입할 때, strcpy()를 사용해야 한다.
#include<string.h> 필요
문자배열 name[]의 경우, 다음과 같은 문장은 허용되지 않는다.
s1.name = “kim”;

● 구조체 정의

```
// Point 라는 이름 or 태그(tag)의 구조체 정의
struct Point
{
    int x; // Point 구조체를 구성하는 멤버변수 x
    int y; // Point 구조체를 구성하는 멤버변수 y
};
```

```
// Person 이라는 이름의 구조체 정의
struct Person
{
    char name[20];      // 이름을 저장하는 char형 배열 멤버변수 name
    char phoneNum[20];  // 전화번호를 저장하는 char형 배열 멤버변수 phoneNum
    int age;            // 나이를 저장하는 int 형 멤버변수 age
};
```

```
// Record 이라는 이름의 구조체 정의
struct Record
{
    double fKo;     // 국어 성적을 저장하는 double형 멤버변수 fKo
    double fMa;     // 수학 성적을 저장하는 double형 멤버변수 fMa
    double fEng;    // 영어 성적을 저장하는 double형 멤버변수 fEng
    double fInfo;   // 정보 성적을 저장하는 double형 멤버변수 fInfo
};
```

```
// Product 이라는 이름의 구조체 정의
struct Product
{
    char name[20];      // 이름을 저장하는 char형 배열 멤버 name
    char serial[20];    // 일련번호를 저장하는 char형 배열 멤버 serial
    int year;           // 생산연도를 저장하는 int형 배열 멤버 year
    int price;          // 가격을 저장하는 int형 배열 멤버 price
};
```

● 구조체 변수 선언

```
    struct Point myPoint;           // struct Point형 변수 myPoint 선언
    struct Person myPerson;         // struct Person형 변수 myPoint 선언
    struct Record classRecord[20];  // struct Record형 배열 변수 classRecord 선언
                                    // classRecord[0]~classRecord[19] 사용가능
    struct Product Laptop[100];     // struct Product형 배열 변수 Laptop 선언
                                    // Laptop[0]~Laptop[99] 사용가능
```

- 구조체 선언(함수 외부)

```c
#include <stdio.h>
// 함수 정의
void MyFunc();

struct Point  // Point 라는 이름의 구조체 정의 (함수 외부)
{
    int x; // Point 구조체를 구성하는 멤버 x
    int y; // Point 구조체를 구성하는 멤버 y
};

struct Point g_myPoint; // struct Point형 (전역)변수 선언 가능

int main()
{
    struct Point myPoint0; // struct Point형 (지역)변수 선언 가능
    return 0;
}
void MyFunc()
{
    struct Point myPoint1; // struct Point형 (지역)변수 선언 가능
}
```

- 구조체 선언(함수 내부)

```c
#include <stdio.h>
// 함수 정의
void MyFunc();

// struct Point g_myPoint; // struct Point형 (전역)변수 선언 불가(구조체 정의되어있지 않음)

int main()
{
    // Point 라는 이름의 구조체 정의(main() 함수 내부에서만 사용가능)
    struct Point
    {
        int x; // Point 구조체를 구성하는 멤버 x
        int y; // Point 구조체를 구성하는 멤버 y
    };
    struct Point myPoint0; // struct Point형 (지역)변수 선언 가능
    return 0;
}
void MyFunc()
{
//   struct Point myPoint1; // struct Point형 (지역)변수 선언 불가(구조체 정의되어있지 않음)
}
```

● 구조체 멤버 접근(멤버접근 연산자 .)

```c
    struct Person
    {
        double height;
        double weight;
        char name[10];
        short grade;
    };

    struct Person person;

    person.height = 174.2;
    person.weight = 67.8;
    person.grade = 1;
//  person.name = "David"; // 멤버 변수가 문자열인 경우는 이건 허용 안됨
    person.name[0] = 'D';  // 배열 인덱스 접근은 가능!
    strcpy(person.name, "David"); // 문자열 복사 함수. #include <string.h> 필요
```

● 구조체 사용 예제(1)

```c
#include <stdio.h>
#include <string.h>
struct Person // struct Person 구조체 선언
{
    double height;
    double weight;
    char name[10];
    short grade;
};
int main()
{
    // struct Person 구조체 변수 person 선언


    // 구조체 멤버 접근 및 값 채우기









    // 구조체 멤버 접근 및 출력
    printf("person.height : %.1lf\n",           );
    printf("person.weight : %.1lf\n",          );
    printf("person.name : %s\n",           );
    printf("person.grade : %d\n",         );

    return 0;
}
```

실행결과

```
person.height : 174.2
person.weight : 67.8
person.name : 홍길동
person.grade : 1
```

● 구조체 사용 예제(2)

```c
#include <stdio.h>
#include <string.h>
struct Person
{
    double height;
    double weight;
    char name[10];
    short grade;
};
int main()
{
    // struct Person 구조체 변수 person 선언 및 멤버 초기화


    // 구조체 멤버 접근 및 출력
    printf("person.height : %.1lf\n", person.height);
    printf("person.weight : %.1lf\n", person.weight);
    printf("person.name : %s\n", person.name);
    printf("person.grade : %d\n", person.grade);

    return 0;
}
```

실행결과

```
person.height : 170.1
person.weight : 68.2
person.name : 가나다
person.grade : 1
```

● 구조체 변수 선언할 때 매번 struct 예약어를 항상 기술해야 함.

● typedef를 이용하여 자료형을 재선언하면 struct 예약어를 생략할 수 있다.

● typedef는 기존 자료형(type)을 새롭게 정의(define)하는 것이다. 사용 형식은

“typedef 기존자료형 새로운자료형” 형식으로 작성한다.

ex. typedef unsigned char Byte; //부호없는(0~255) char형 변수를 Byte라고 재정의한 것임

Byte c; // c라는 부호 없는 char형 변수가 선언됨.

```c
struct Person{
    double height;
    double weight;
    char name[10];
    short grade;
};
typedef struct Person Per; // 기존 자료형 struct person을 새로운 자료형 Per로 정의
Per p1;  // Per형(struct Person형) 변수 p1을 선언
```

```c
typedef struct Person {
    double height;
    double weight;
    char name[10];
    short grade;
} Per;       // 기존 자료형 struct Person을 새로운 자료형 Per로 정의
Per p1; // struct 예약어 생략 가능
```

● 구조체 사용 예제(3)

```c
#include <stdio.h>
typedef struct Person_
{
    double height;
    double weight;
    char name[10];
    short grade;
} Person;

int main()
{
    // struct Person 구조체 변수 person 선언
    Person person;

    // 멤버 입력받기
    printf("이름 : ");
    scanf("%s", person.name);
    printf("키 : ");
    scanf("%lf", &person.height);
    printf("몸무게 : ");
    scanf("%lf", &person.weight);
    printf("등급 : ");
    scanf("%d", &person.grade);

    // 입력받은 구조체 데이터 출력
    printf("\n데이터 출력\n");
    printf("person.height : %.1lf\n", person.height);
    printf("person.weight : %.1lf\n", person.weight);
    printf("person.name : %s\n", person.name);
    printf("person.grade : %d\n", person.grade);

    return 0;
}
```

실행결과

```
이름 : 마이스
키 : 170.1
몸무게 : 80.1
등급 : 3

데이터 출력
person.height : 170.1
person.weight : 80.1
person.name : 마이스
person.grade : 3
```

● 구조체 연습

문자열 형태의 '종업원 이름'과 문자열 형태의 '주민등록번호' 그리고 정수형태의 '급여정보'를 저장할 수 있는 employee라는 이름의 구조체를 정의해보자. 그리고 나서 employee 구조체 변수를 하나 선언한 다음, 프로그램 사용자가 입력하는 정보로 이 변수를 채우자, 그리고 마지막으로 구조체 변수에 채워진 데이터를 출력해보자.

- 구조체 멤버 순서에 따른 sizeof 차이 확인!

```c
#include <string.h>
typedef struct Person_1
{
    char ch1;
    short num;
    char ch2;
    int score;
    double grade;
    char ch3;
} Person1;

Person1 p1;

int main()
{
    printf("p1 sizeof : %d\n", sizeof(p1));
    return 0;
}
```

```c
#include <string.h>
typedef struct Person_2
{
    char ch1;
    char ch2;
    short num;
    int score;
    double grade;
    char ch3;
} Person2;

Person2 p2;

int main()
{
    printf("p2 sizeof : %d\n", sizeof(p2));
    return 0;
}
```

- 구조체 배열

```c
// 구조체 선언
struct Point
{
    int xpos;
    int ypos;
};
// struct Point형 구조체 배열 선언
struct Point arr[3];
```

```c
 // 구조체 선언 및 형 재선언
typedef struct point
{
    int xpos;
    int ypos;
} Point;
// Point형 구조체 배열 선언
Point arr[3];
```

```c
arr[0].xpos = 10; //    (*arr).xpos = 10;
arr[0].ypos = 10; //    (*arr).ypos = 10;
arr[1].xpos = 20; //    (*(arr + 1)).xpos = 20;
arr[1].ypos = 20; //    (*(arr + 1)).ypos = 20;
arr[2].xpos = 30; //    (*(arr + 2)).xpos = 30;
arr[2].ypos = 30; //    (*(arr + 2)).ypos = 30;
```