Chapter 13

구조체, 공용체, 열거형

목차

01
구조체

1. 구조체

1. 구조체의 개념

그림 13-1 구조체와 비빔밥의 개념 비교

- 비빔밥 : 별도의 재료들을 하나의 그릇에 담음
- 구조체 : 서로 다른 변수의 형태를 하나의 블록으로 묶음
- 비빔밥에 들어 있는 당근을 '비빔밥 안에 있는 당근'이라고 부를 수 있듯이 구조체 안에 있는 변수도 '구조체 안에 있는 변수'라고 표현할 수 있음
- 이를 코드 형태로 표현하면 '구조체 이름.변수 이름'

1. 구조체

- **구조체 만들기**

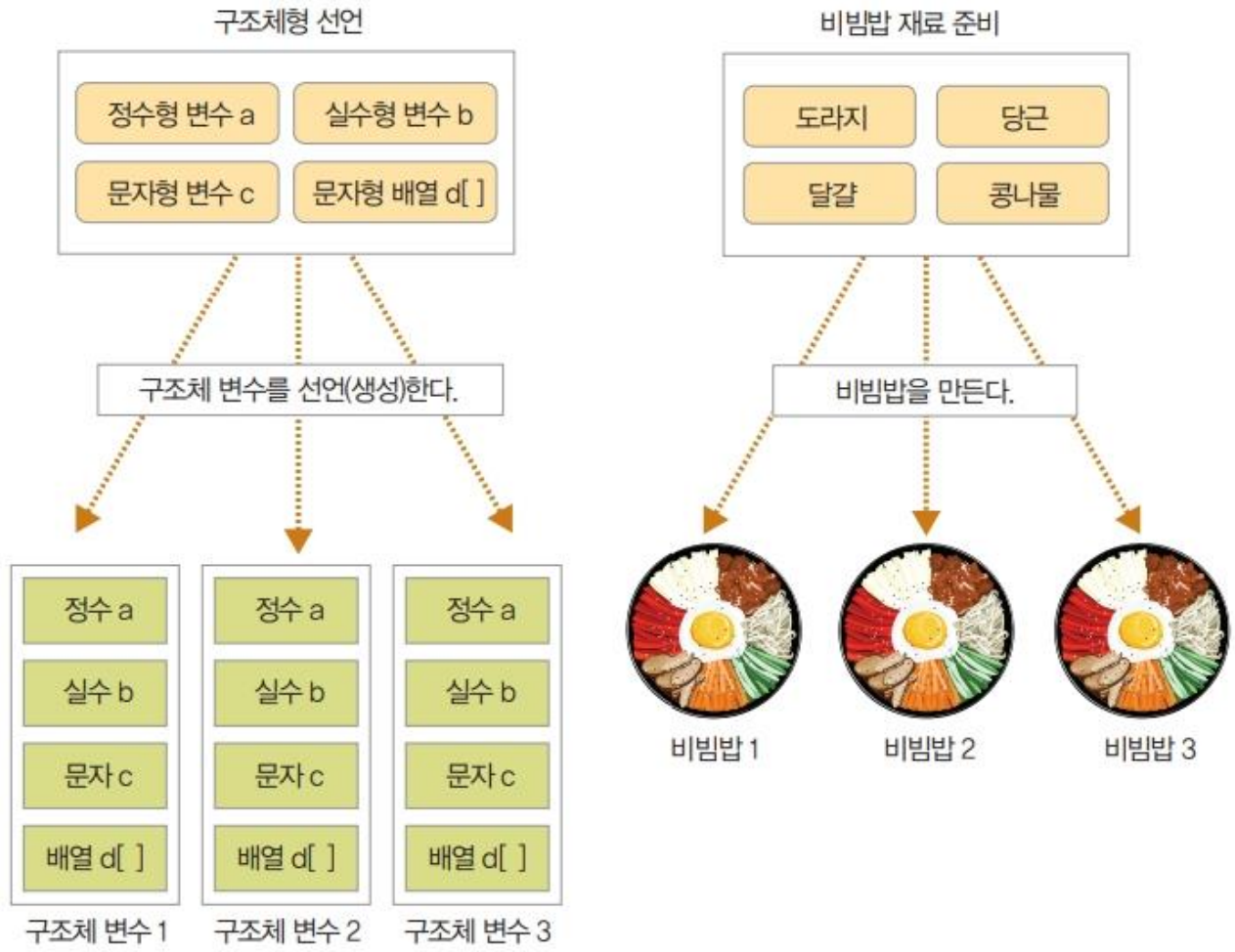

그림 13-2 구조체 변수와 비빔밥을 만드는 방법

1. 구조체

- **구조체로 표현**

❶ 비빔밥 만들기

```
비빔밥 재료{
    도라지 dora;
    당근 dang;
    계란 egg;
    콩나물 kong;
} ;
비빔밥 재료 비빔밥1;
비빔밥 재료 비빔밥2;
비빔밥 재료 비빔밥3;
```

❷ 구조체로 표현하기

```
struct bibim {
    int a;
    char b;
    float c;
    char d[5];
} ;
struct bibim b1;
struct bibim b2;
struct bibim b3;
```

❶에서는 비빔밥 재료를 준비해서 이 재료들로 비빔밥 세 그릇을 만드는 것을 보여줌

❷에서도 bibim이라는 구조체형을 선언한 후 구조체 변수 세 개를 생성

1. 구조체

2. 구조체의 문법

- 멤버 변수 : 구조체 안에서 정의된 변수를 뜻하며 일반적인 변수 선언과 동일한 방법으로 선언

```
struct  구조체형_이름 {
    데이터_형식  멤버_변수 1;
    데이터_형식  멤버_변수 2;
    …
};
struct  구조체형_이름  구조체_변수;
```

기본 13-1 구조체 사용 예 13-1.c

```
01 #define  _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 #include <string.h>
04 void main( )
05 {
06    struct  bibim {          ----- 구조체형 bibim을 선언한다(아직 저장 공간이 없다).
07       int  a;
08       float  b;
09       char  c;
10       char  d[5];
11    };
```

1. 구조체

2. 구조체의 문법

```
12
13    struct  bibim  b1;          ----- 구조체 변수 b1을 선언한다(실제 저장 공간을 확보한다).
14
15    b1.a = 10;                  ----- 구조체 변수의 멤버 변수에 값을 대입한다.
16    b1.b = 1.1f;
17    b1.c = 'A';
18    strcpy(b1.d, "ABCD");
19
20    printf(" b1.a ==> %d \n", b1.a);    ----- 구조체 변수의 멤버 변숫값을 출력한다.
21    printf(" b1.b ==> %f \n", b1.b);
22    printf(" b1.c ==> %c \n", b1.c);
23    printf(" b1.d ==> %s \n", b1.d);
24 }
```

실행 결과

```
b1.a ==> 10
b1.b ==> 1.100000
b1.c ==> A
b1.d ==> ABCD
```

1. 구조체

2. 구조체의 문법

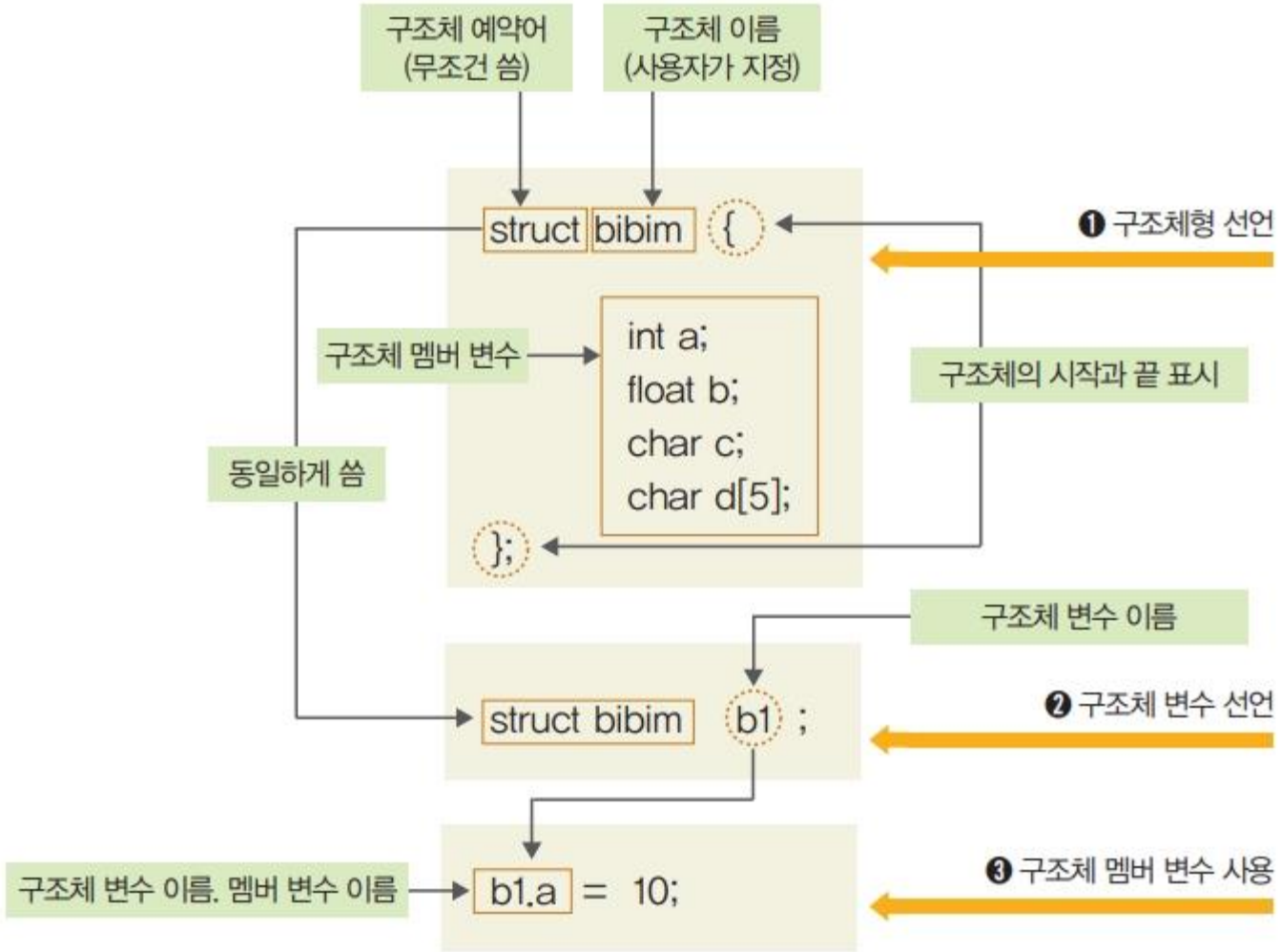

그림 13-3 구조체의 문법 구조

2. 구조체의 문법

- [그림 13-3]은 'bibim'이라는 이름의 구조체형을 선언하는 과정을 나타낸 것
- 이 구조체 에는 멤버 변수 4개가 있음
- 구조체형의 마지막은 반드시 };으로 끝나야 함
- 구조체형을 만들었으면 구조체형의 모양대로 구조체 변수를 선언
- 구조체 변수를 선언하려면 선언할 구조체 이름을 그대로 쓰고 생성할 구조체 변수 이름을 이어서 씀
- 그런 다음 '구조체 변수 이 름.멤버 변수 이름' 형태로 쓰고 일반 변수처럼 사용

여기서 잠깐 구조체형과 구조체 변수

- 붕어빵(구조체 변수)을 만들고 싶으면 먼저 붕어빵 기계(구조체형)를 만들어야 함
- 붕어빵 기계(구조체형)는 한 번 만들어놓으면 붕어빵(구조체 변수)을 여러 개 찍어낼 수 있음

3. 구조체의 활용

▪ 구조체의 기본 활용

기본 13-2 구조체를 사용하지 않은 예 13-2.c

```
01 #define _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 void main( )
04 {
05    char name[10];
06    int kor;
07    int eng;
08    float avg;
09
10    printf("이름 : ");
11    scanf("%s", name, 9);
12
13    printf("국어 점수 : ");
14    scanf("%d", &kor);
15
```

05~08 ---- 학생 이름, 국어 점수, 영어 점수, 평균 점수 변수를 선언한다.

11 ---- 학생 이름을 입력한다. 최대 9자를 입력할 수 있다.

14 ---- 국어 점수를 입력한다.

3. 구조체의 활용

▪ 구조체의 기본 활용

```
16    printf("영어 점수 : ");
17    scanf("%d", &eng);          ----- 영어 점수를 입력한다.
18
19    avg = (kor + eng) / 2.0f;   ----- 평균 점수를 계산한다.
20
21    printf("\n");
22    printf("학생 이름 ==> %s\n", name);   ----- 학생 이름, 국어 점수, 영어 점수,
23    printf("국어 점수   ==> %d\n", kor);          평균 점수를 출력한다.
24    printf("영어 점수   ==> %d\n", eng);
25    printf("평균 점수   ==> %5.1f\n", avg);
26 }
```

실행 결과

```
이름 : James
국어 점수 : 87
영어 점수 : 72

학생 이름 ==> James
국어 점수   ==> 87
영어 점수   ==> 72
평균 점수   ==> 79.5
```

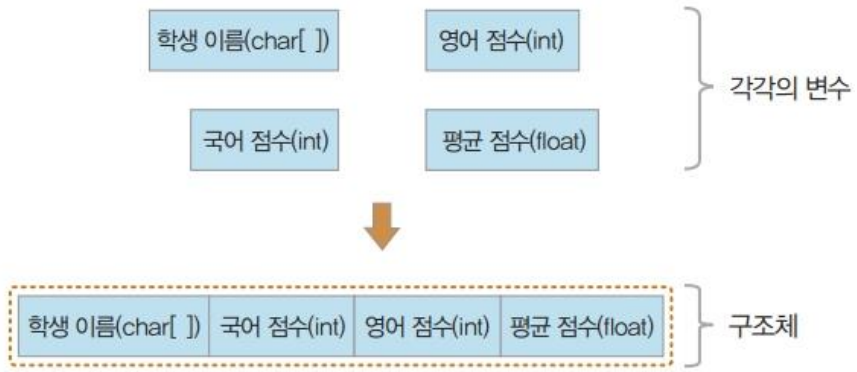

그림 13-4 구조체를 사용한 변수 선언

3. 구조체의 활용

▪ 구조체의 기본 활용

기본 13-3 구조체를 사용하여 변경한 [기본 13-2] 13-3.c

```
01 #define _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 void main( )
04 {
05    struct student {
06       char name[10];
07       int  kor;
08       int  eng;
09       float  avg;
10    };
11
12    struct student s;
13
14    printf("이름 : ");
15    scanf("%s", s.name, 9);
16
```

----- student 구조체형을 선언한 후 멤버 변수를 선언한다(학생 이름, 국어 점수, 영어 점수, 평균 점수).

----- 구조체 변수 s를 선언한다.

----- 학생 이름을 입력한다.

3. 구조체의 활용

▪ 구조체의 기본 활용

```
17    printf("국어 점수 : ");
18    scanf("%d", &s.kor);          ----- 국어 점수를 입력한다.
19
20    printf("영어 점수 : ");
21    scanf("%d", &s.eng);          ----- 영어 점수를 입력한다.
22
23    s.avg = (s.kor + s.eng) / 2.0f;   ----- 평균 점수를 계산한다.
24
25    printf("\n--- 구조체 활용 ---\n");
26    printf("학생 이름 ==> %s\n", s.name);   ----- 학생 이름, 국어 점수, 영어 점수,
27    printf("국어 점수   ==> %d\n", s.kor);        평균 점수를 출력한다.
28    printf("영어 점수   ==> %d\n", s.eng);
29    printf("평균 점수   ==> %5.1f\n", s.avg);
30 }
```

실행 결과

```
이름 : James
국어 점수 : 87
영어 점수 : 72

--- 구조체 활용 ---
학생 이름 ==> James
국어 점수   ==> 87
영어 점수   ==> 72
평균 점수   ==>  79.5
```

1. 구조체

▪ 구조체 변수의 초깃값 대입

❶ 일반 변수의 초깃값 대입

```
char name[10] = "Woo";
int kor = 90;
int eng = 80;
float avg;
```

→

❷ 구조체 변수의 초깃값 대입

```
struct student {
   char name[10];
   int kor;
   int eng;
   float avg;
};

struct student s = {"Woo", 90, 80 };
```

- 구조체 변수를 초기화할 때는 중괄호({ }) 안의 초깃값을 콤마(,)로 분리해서 대입

1. 구조체

▪ 구조체 변수의 다른 선언 방법

❶ 구조체형과 변수를 별도로 선언

```
struct student {
   char name[10];
   int kor;
   int eng;
   float avg;
};

struct student s;
```

❷ 구조체형과 변수를 동시에 선언

```
struct student {
   char name[10];
   int kor;
   int eng;
   float avg;
}s;
```

❸ typedef를 이용하여 선언

```
typedef struct _student {
   char name[10];
   int kor;
   int eng;
   float avg;
}student;

student s;
```

- ❶은 구조체형을 만든 후 구조체 변수 s를 별도로 선언하는 방식
- ❷는 구조체형을 선언하는 동시에 구조체 변수 s를 선언하는 방식
- ❸은 실무에서 많이 사용하는 방법

1. 구조체

▪ 구조체 배열

기본 13-4 구조체 배열을 사용하지 않은 예 13-4.c

```
01 #define  _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 #include <string.h>
04 void main( )
05 {
06    char name[3][10];          ----- 학생 이름 배열을 선언한다.
07    int  kor[3];               ----- 국어 점수 배열을 선언한다.
08    int  eng[3];               ----- 영어 점수 배열을 선언한다.
09    float  avg[3];             ----- 평균 점수 배열을 선언한다.
10
11    int i;
12
13    strcpy(name[0], "Kim");    ----- 첫 번째 학생의 정보를 대입한다(학생
14    kor[0] = 90;                     이름, 국어 점수, 영어 점수, 평균 점수).
15    eng[0] = 80;
16    avg[0] = (kor[0] + eng[0]) / 2.0f;
17
```

1. 구조체

▪ 구조체 배열

```
18    strcpy(name[1], "Lee");
19    kor[1] = 70;
20    eng[1] = 60;
21    avg[1] = (kor[1] + eng[1]) / 2.0f;
22
23    strcpy(name[2], "Park");
24    kor[2] = 50;
25    eng[2] = 40;
26    avg[2] = (kor[2] + eng[2]) / 2.0f;
27
28    for(i=0; i < 3; i++)
29    {
30       printf("학생 이름 ==> %s\n", name[i]);
31       printf("국어 점수 ==> %d\n", kor[i]);
32       printf("영어 점수 ==> %d\n", eng[i]);
33       printf("평균 점수 ==> %5.1f\n", avg[i]);
34       printf("\n");
35    }
36 }
```

----- 두 번째 학생의 정보를 대입한다(학생 이름, 국어 점수, 영어 점수, 평균 점수).

----- 세 번째 학생의 정보를 대입한다(학생 이름, 국어 점수, 영어 점수, 평균 점수).

----- 세 번 반복하고 배열의 내용을 출력한다.

실행 결과

```
학생 이름 ==> Kim
국어 점수 ==> 90
영어 점수 ==> 80
평균 점수 ==>  85.0

학생 이름 ==> Lee
국어 점수 ==> 70
영어 점수 ==> 60
평균 점수 ==>  65.0

학생 이름 ==> Park
국어 점수 ==> 50
영어 점수 ==> 40
평균 점수 ==>  45.0
```

1. 구조체

▪ 구조체 배열

- 6~9행에서 학생 이름 3개, 국어 점수 3개, 영어 점수 3개, 평균 점수 3개로 각각 배열을 선언 하고 13~26행에서 각각의 배열에 값을 입력
- 28~35행에서는 입력된 배열값을 출력
- 프로그램 자체는 틀리지 않았지만 [그림 13-5]와 같이 구조체 배열로 변환하면 훨씬 논리적으로 표현할 수 있음

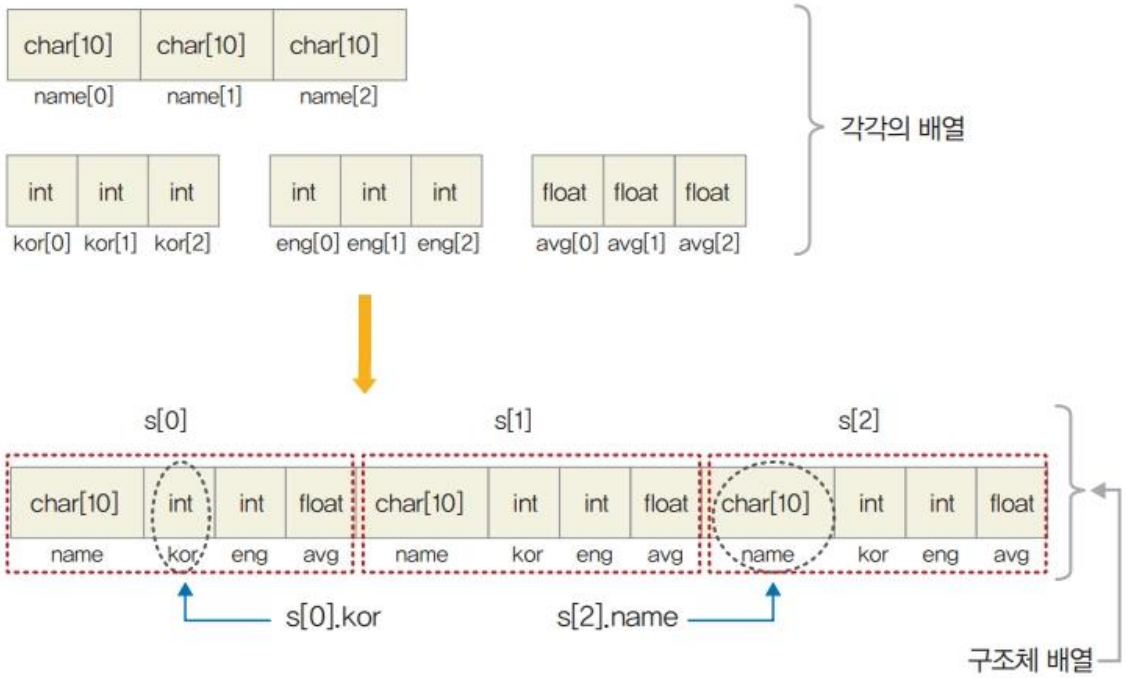

그림 13-5 구조체를 사용한 배열 선언

1. 구조체

- **구조체 배열**

응용 13-5 구조체 배열을 사용하여 변경한 [기본 13-4] 13-5.c

```
01 #define _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 #include <string.h>
04 void main( )
05 {
06    struct student {                    ----- 구조체형을 선언한다.
07       char name[10];
08       int  kor;
09       int  eng;
10       float  avg;
11    };
12
13    __1__  s[3];                        ----- 구조체 배열 s[3]을 선언한다.
14
15    int i;
16
```

1. 구조체

- **구조체 배열**

```
17    strcpy(s[0].name, "Kim");
18    s[0].kor = 90;
19    s[0].eng = 80;
20    s[0].avg = (s[0].kor + s[0].eng) / 2.0f;
21
22    strcpy( __2__, "Lee");
23    s[1].kor = 70;
24    s[1].eng = 60;
25    s[1].avg = (s[1].kor + s[1].eng) / 2.0f;
26
27    strcpy(s[2].name, "Park");
28    s[2].kor = 50;
29    s[2].eng = 40;
30    s[2].avg = (s[2].kor + s[2].eng) / 2.0f;
31
32    printf("-- 구조체 배열 -- \n");
33    for(i=0; i < 3; i++)
34    {
35      printf("학생 이름 ==> %s\n", s[i].name);
36      printf("국어 점수   ==> %d\n", __3__);
```

----- 첫 번째 학생의 정보를 대입한다 (학생 이름, 국어 점수, 영어 점수, 평균 점수).

----- 두 번째 학생의 정보를 대입한다 (학생 이름, 국어 점수, 영어 점수, 평균 점수).

----- 세 번째 학생의 정보를 대입한다 (학생 이름, 국어 점수, 영어 점수, 평균 점수).

----- 세 번 반복하고 구조체 배열의 내용을 출력한다.

1. 구조체

▪ 구조체 배열

```
37      printf("영어 점수   ==> %d\n", s[i].eng);
38      printf("평균 점수   ==> %5.1f\n", s[i].avg);
39      printf("\n");
40    }
41  }
```

정답_ 1 struct student 2 s[i].name 3 s[i].kor

실행 결과

```
-- 구조체 배열 --
학생 이름 ==> Kim
국어 점수   ==> 90
영어 점수   ==> 80
평균 점수   ==>  85.0

학생 이름 ==> Lee
국어 점수   ==> 70
영어 점수   ==> 60
평균 점수   ==>  65.0

학생 이름 ==> Park
국어 점수   ==> 50
영어 점수   ==> 40
평균 점수   ==>  45.0
```

1. 구조체

- **구조체 포인터** : 구조체 포인터 변수도 구조체의 주소를 가지고 있음

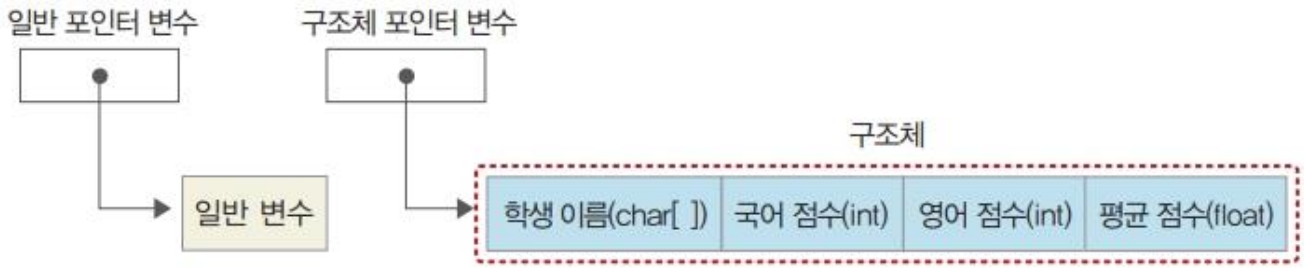

그림 13-6 일반 포인터 변수와 구조체 포인터 변수

- 일반 포인터 변수와 구조체 포인터 변수를 사용하는 경우

❶ 일반 포인터 변수 사용

```
int a;
int* p;
p = &a;
*p = 100;
```

❷ 구조체 포인터 변수 사용

```
struct student {
    char name[10];
    int kor;
    int eng;
    float avg;
};
struct student s;
struct student *p;
p = &s;
p->kor = 100;
```

1. 구조체

▪ 구조체 포인터

기본 13-6 구조체 포인터 사용 예 — 13-6.c

```
01 #define  _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 void main( )
04 {
05    struct student {                  ----- student 구조체형을 선언한 후 멤버
06       char name[10];                       변수를 선언한다(학생 이름, 국어 점수,
07       int  kor;                            영어 점수, 평균 점수).
08       int  eng;
09       float  avg;
10    };
11
12    struct student s;                 ----- 구조체 변수 s를 선언한다.
13    struct student *p;                ----- 구조체 포인터 변수 p를 선언한다.
14
15    p = &s;                           ----- 포인터 변수 p에 s의 주소를 대입한다.
16
17    printf("이름 입력 : ");
18    scanf("%s", p->name);             ----- 이름을 입력한다.
19
20    printf("국어 점수 : ");
```

1. 구조체

▪ 구조체 포인터

```
21    scanf("%d", &p->kor);                  ----- 국어 점수를 입력한다.
22
23    printf("영어 점수 : ");
24    scanf("%d", &p->eng);                  ----- 영어 점수를 입력한다.
25
26    p->avg = (p->kor + p->eng) / 2.0f;     ----- 평균 점수를 계산한다.
27
28    printf("\n--- 구조체 포인터 활용 ---\n");
29    printf("학생 이름 ==> %s\n", p->name);  ----- 학생 이름을 출력한다.
30    printf("국어 점수   ==> %d\n", p->kor);  ----- 국어 점수를 출력한다.
31    printf("영어 점수   ==> %d\n", p->eng);  ----- 영어 점수를 출력한다.
32    printf("평균 점수   ==> %5.1f\n", p->avg);  ----- 평균 점수를 출력한다.
33 }
```

실행 결과

```
이름 입력 : James
국어 점수 : 87
영어 점수 : 72

--- 구조체 포인터 활용 ---
학생 이름 ==> James
국어 점수   ==> 87
영어 점수   ==> 72
평균 점수   ==>  79.5
```

- 13행에서 구조체 포인터 변수를 선언하고 15행에서 구조체 변수 s의 주소를 p에 대입
- 변수에 값을 입력하려면 '구조체 포인터 변 수->멤버 이름'과 같은 형식(p->kor)을 사용

02
공용체

2. 공용체

1. 공용체의 개념

- 공용체 : 하나의 공간(메모리)을 서로 다른 두 변수가 같이 사용하는 것

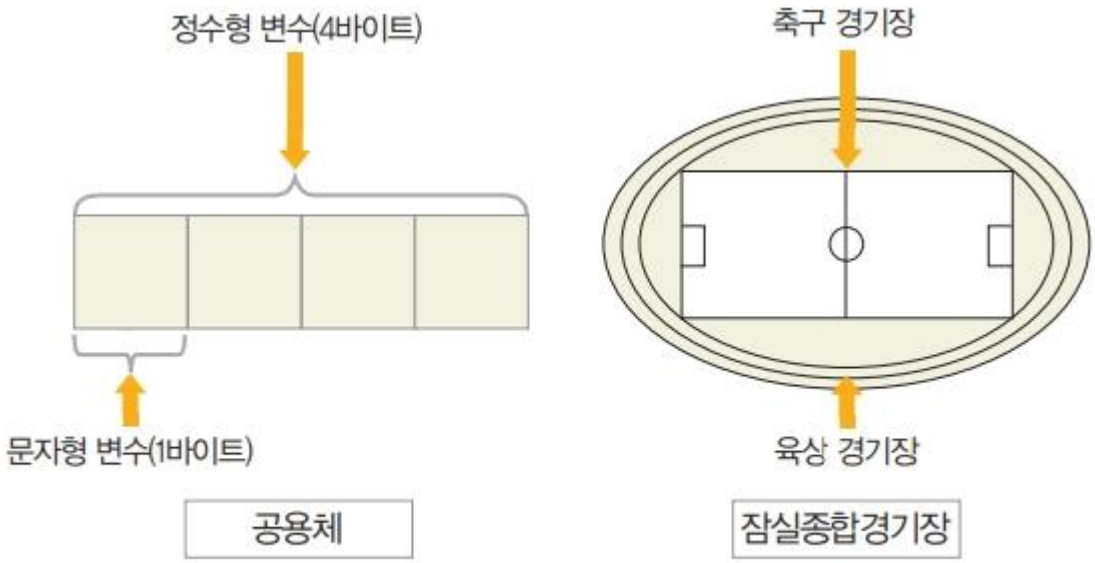

그림 13-7 공용체와 잠실종합경기장의 개념 비교

2. 공용체

1. 공용체의 문법

- 공용체를 사용하는 방법은 구조체와 거의 비슷한데 struct 대신 union을 사용
- 차이점은 구조체의 경우 멤버 변수 각각에 별도의 공간을 할당하는 반면 공용체에서는 멤버 변수가 공간을 공유한다는 것

```
union  공용체형_이름 {
    데이터_형식  멤버_변수_1;
    데이터_형식  멤버_변수_2;
                ⋮
};

union  공용체형_이름  공용체_변수;
```

2. 공용체

1. 공용체의 문법

기본 13-7 공용체 사용 예 13-7.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    union student {                  ---- 공용체 변수 student를 선언한다.
06       int tot;
07       char grade;
08    };
09
10    union student u;                 ---- 공용체 변수 u를 선언한다.
11
12    u.tot = 300;                     ---- 공용체 변수의 멤버 변수에 값을 대입한다.
13    u.grade = 'A';
14
15    printf("\n--- 공용체 활용 ---\n");
16    printf("총점 ==> %d\n", u.tot);  ---- 공용체 변수의 멤버 변숫값을 출력한다.
17    printf("등급 ==> %c\n", u.grade);
18 }
```

실행 결과

```
--- 공용체 활용 ---
총점 ==> 321
등급 ==> A
```

2. 공용체

1. 공용체의 문법

- 5행에서 union 예약어를 사용하여 공용체를 선언하고 10행에서 공용체 변수를 선언
- 12행과 13행에서 각각 '300'과 'A'라는 데이터를 입력하고 16~17행에서 입력된 데이터를 출력

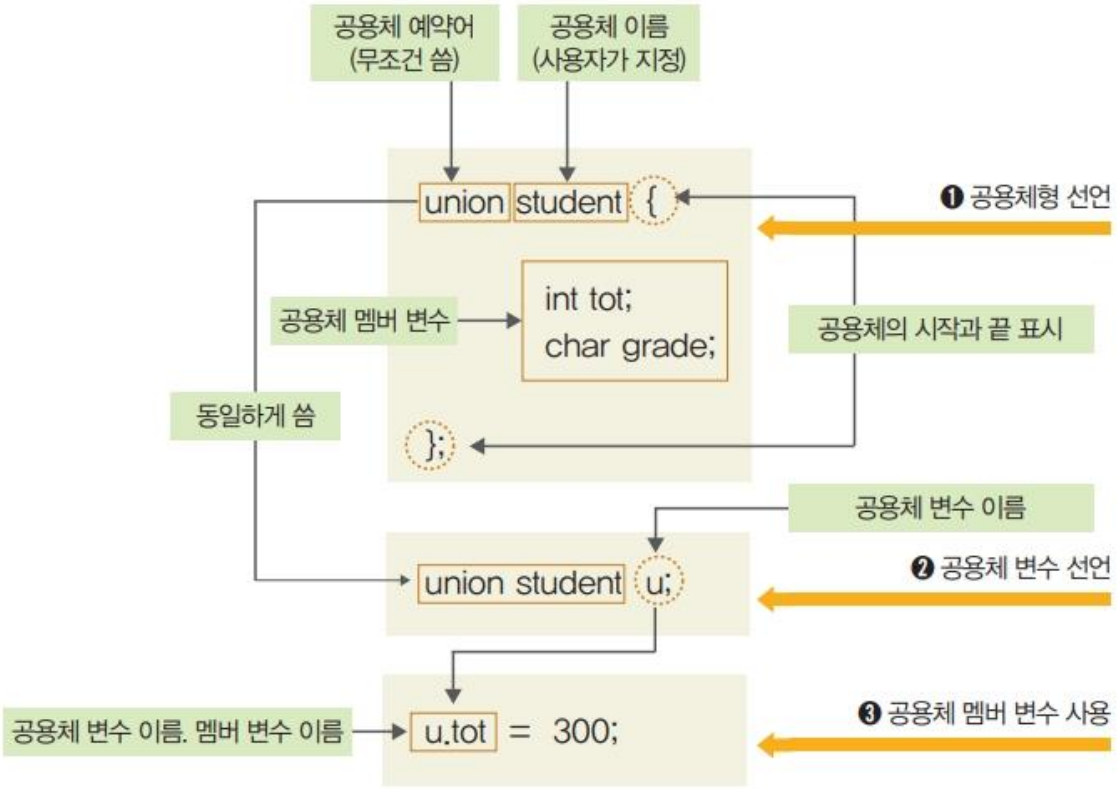

그림 13-8 공용체의 문법 구조

2. 공용체

1. 공용체의 문법

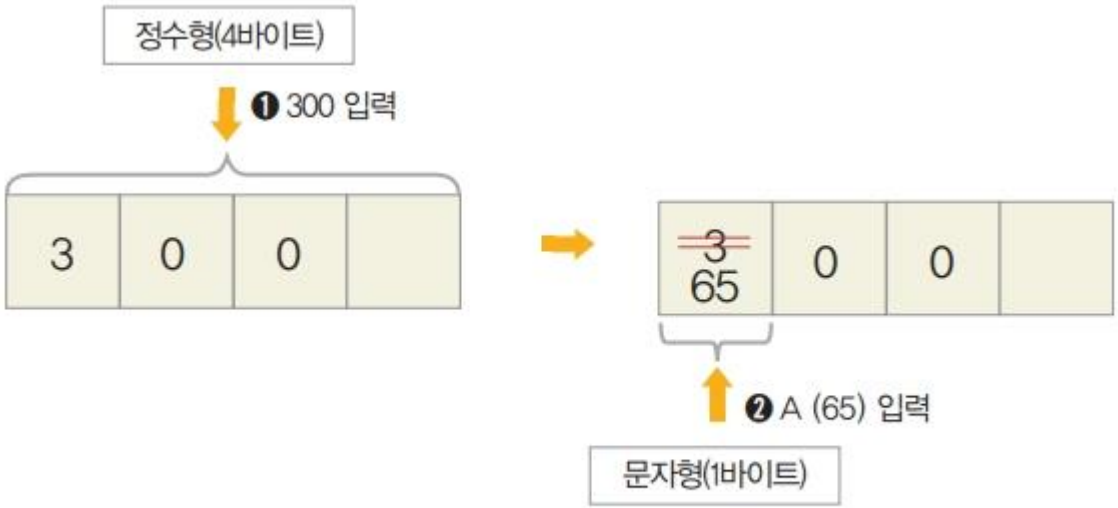

그림 13-9 [기본 13-7]의 저장 과정

여기서 잠깐 공용체의 크기

공용체 멤버 변수가 여러 개일 때는 그중 가장 큰 저장 공간이 공용체의 크기가 됨

03
열거형

3. 열거형

- **열거형의 형식**
  - 단순히 1, 2, 3, 4, … 와 같은 숫자를 기호를 써서 표현함

```
enum  열거형_이름{
   기호 1;
   기호 2;
   …
};

enum  열거형_이름  열거형_변수;
```

- **요일을 열거형으로 표현**
  - 0은 sun, 1은 mon, 2는 tue, … 등과 같이 의미가 좀더 명확해짐
  - 나열한 데이터의 값은 0에서부터 1씩 차례대로 증가함

그림 13-10 열거형의 개념

3. 열거형

- 요일을 열거형으로 표현

기본 13-8 열거형 사용 예 — 13-8.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    enum  week {sun, mon, tue, wed, thu, fri, sat};
06
07    enum  week  ww;
08
09    ww = sat;
10
11    if(ww == sun)
12       printf("오늘은 일요일입니다.\n");
13    else
14       printf("오늘은 일요일이 아닙니다.\n");
15 }
```

----- 0부터 6까지의 열거형이다.

----- 열거형 변수 ww를 선언한다.

----- 변수 ww에 값을 대입한다.

----- ww가 sun(0)인지 아닌지의 여부에 따라 문장을 출력한다.

실행 결과

```
오늘은 일요일이 아닙니다.
```

3. 열거형

- **요일을 열거형으로 표현**
  - 5행에서 열거형의 형식을 선언했는데 enum week라는 열거형은 0~6의 내용을 담고 있음
  - 7행에서는 ww라는 열거형 변수를 선언했는데 여기에는 sun~sat(실제 값은 0~6)를 대입 가능
  - 9행에서 ww에 sat(실제 값은 6)를 대입하고 11행에서 ww의 값에 따라 일요일인지 여부를 출력

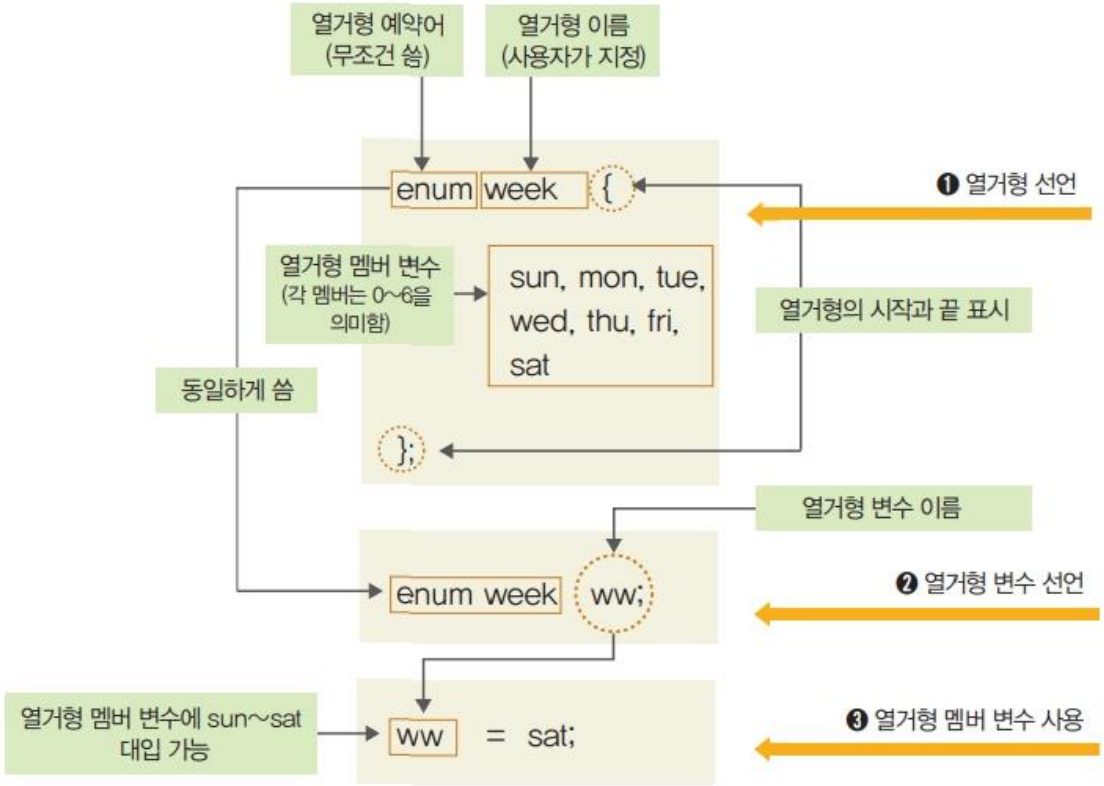

그림 13-11 열거형의 문법 구조

*

예제 모음

[예제모음 34] 구조체 포인터를 활용한 학생 관리

예제 설명 구조체 포인터와 메모리 할당 함수를 이용하여 학생의 이름과 나이를 입력받아 출력하는 프로그램이다.

실행 결과

```
입력할 학생 수 : 3
이름과 나이 입력 : 지효  25
이름과 나이 입력 : 정연  25
이름과 나이 입력 : 다현  23


-- 학생 명단 --
이름:지효 , 나이:25
이름:정연 , 나이:25
이름:다현 , 나이:23
```

[예제모음 34] 구조체 포인터를 활용한 학생 관리

```
01 #define  _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 #include <malloc.h>
04 void main( )
05 {
06    struct student{                          ----- 구조체형을 선언한다.
07       char name[10];
08       int  age;
09    };
10
11    struct student *s;                       ----- 구조체 포인터 변수를 선언한다.
12
13    int cnt, i;
14
15    printf("입력할 학생 수 : ");
16    scanf("%d", &cnt);                       ----- 관리할 학생 수를 입력한다.
17
18    s = (struct student*) malloc((sizeof(struct student)) * cnt);
19                                             ----- 학생 수만큼 메모리를 할당한다.
20    for(i=0; i < cnt; i++)                   ----- 학생 수만큼 반복하며 이름과
21    {                                              나이를 입력한다.
22       printf("이름과 나이 입력 : ");
```

[예제모음 35] 구조체와 공용체의 혼합

예제 설명 사용자 이름을 입력하고 전화번호 또는 주민번호 중 한 가지만 입력하는 프로그램이다. 전화번호와 주민번호는 공용체로 동일한 메모리를 차지함으로써 공간을 절약한다. 구조체와 공용체를 혼합해서 사용해보자.

실행 결과

```
이름 --> 채영
전화번호 또는 주민번호 --> 990423-1234567

--- 구조체/공용체 혼합 활용 ---
이름 ==> 채영
전화번호/주민번호 ==> 990423-1234567
```

[예제모음 35] 구조체와 공용체의 혼합

```
01 #define  _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 void main( )
04 {
05    typedef struct _person {          ------ 구조체형을 정의할 때 그 내부에 공용체를
06       char name[10];                       사용한다.
07       union _id {                    ------- 공용체 변수 id를 선언한다.
08          char phone[15];
09          char jumin[15];
10       } id;
11    } person;
12
13    person p1;                        ------ p1의 크기는 총 25바이트이다.
14
15    printf("이름 --> ");
16    scanf("%s", p1.name);
17    printf("전화번호 또는 주민번호 --> ");
18    scanf("%s", p1.id.jumin);         ------ p1 안 공용체 id의 멤버 변수 jumin에 입력받는다.
19                                            이 부분은 phone으로 해도 동일하다.
20    printf("\n--- 구조체/공용체 혼합 활용 ---\n");
21    printf("이름 ==> %s\n", p1.name);
22    printf("전화번호/주민번호 ==> %s\n", p1.id.phone );
23 }        └---- p1 안 공용체 id의 멤버 변수 phone을 출력한다. 이 부분은 jumin으로 해도 동일하다.
```

[예제모음 36] 열거형을 활용한 월 이름 출력

예제 설명 열거형을 활용해서 입력된 월의 이름을 출력하는 프로그램이다.

실행 결과

```
월 입력 : 8
8월은 August 입니다.
```

[예제모음 36] 열거형을 활용한 월 이름 출력

```
01 #define  _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 void main( )
04 {
05    enum  month {                          ---- 열거형을 선언한다(1부터 시작).
06       January=1, February, March, April,
07       May, June, July, August,
08       September, October, November, December
09    };
10
11    enum month mm;                         ---- 열거형 변수 mm을 선언한다.
12
13    printf("월 입력 : ");
14    scanf("%d", &mm);                      ---- 값을 입력한다.
```

[예제모음 36] 열거형을 활용한 월 이름 출력

```
15                                          ┌---- 입력값에 따라 해당 월을 출력한다.
16    switch (mm)
17    {
18       case January  : printf("%d월은 January 입니다.", mm); break;
19       case February : printf("%d월은 February 입니다.", mm); break;
20       case March    : printf("%d월은 March 입니다.", mm); break;
21       case April    : printf("%d월은 April 입니다.", mm); break;
22       case May      : printf("%d월은 May 입니다.", mm); break;
23       case June     : printf("%d월은 June 입니다.", mm); break;
24       case July     : printf("%d월은 July 입니다.", mm); break;
25       case August   : printf("%d월은 August 입니다.", mm); break;
26       case September: printf("%d월은 September 입니다.", mm); break;
27       case October  : printf("%d월은 October 입니다.", mm); break;
28       case November : printf("%d월은 November 입니다.", mm); break;
29       case December : printf("%d월은 December 입니다.", mm); break;
30       default :printf("잘못 입력했군요.");
31    }
32    printf("\n\n");
33 }
```

감사합니다!