13. 구조체와 공용체

10주차

구조체의 개념

- 구조체
  - 연관선 있는 다양한 자료형을 묶어 하나의 자료형으로 이용하는 것
    - ❖ 동일한 자료형들의 모임인 배열과 비교됨.
  - C언어에서는 예약어 struct 로 지원
  - predefined-typw, user-defined type 모두 포함 가능.
- 구조체 사용 방법
  - 구조체 정의
  - 구조체 타입의 변수 선언
  - 구조체 변수의 멤버 사용

```
struct lecture {
    char name[20];
    int credit;
    int hour;
};
```

```
struct lecture lecture1;
```

```
lecture1.hour = 2;
```

구조체 변수 선언과 초기화

- 변수 선언 시 중괄호를 이용한 변수 초기화 가능
- 초기화 값은 중괄호 내부에서 각 멤버 정의 순서대로 초기값을 쉼표로 구분하여 기술
- 기술되지 않은 멤버값은 default값 0을 가짐.

```
struct lecture {
    char name[20];
    int credit;
    int hour;
} lecture = { "홍길동", 1000, 2 };
```

구조체 정의와 변수 선언을 함께하는 방법

멤버 초기화

구조체 정의와 구조체 변수 선언
(structbasic.c)

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <string.h>
struct account {
    char name[12];        //계좌주 이름
    int actnum;           //계좌번호
    double balance;       //잔고
};
int main(void) {
    struct account mine = { "홍길동", 1001, 300000 };
    struct account yours;
    strcpy(yours.name, "이동원");
    yours.actnum = 1002;
    yours.balance = 500000;
    printf("구조체크기: %d₩n", sizeof(mine));
    printf("%s %d %.2f₩n", mine.name, mine.actnum, mine.balance);
    printf("%s %d %.2f₩n", yours.name, yours.actnum, yours.balance);
    return 0;
}
```

은행 계좌를 위한 구조체 정의

[결과]
구조체크기: 24
홍길동 1001 300000.00
이동원 1002 500000.00

실제 구조체의 크기는 멤버의 크기의 합보다 크거나 같을 수 있음

구조체 멤버로 다른 구조체 허용
(nestedstruct.c)

```
#include <stdio.h>
#include <string.h>
struct date {
    int year;       //년
    int month;      //월
    int day;        //일
};
struct account {
    struct date open;     //계좌 개설일자
    char name[12];        //계좌주 이름
    int actnum;           //계좌번호
    double balance;       //잔고
};
int main(void) {
    struct account me = { { 2018, 3, 9 }, "홍길동", 1001, 300000 };
    printf("구조체크기: %d₩n", sizeof(me));
    printf("[%d, %d, %d]₩n", me.open.year, me.open.month, me.open.day);
    printf("%s %d %.2f₩n", me.name, me.actnum, me.balance);
}
```

날짜를 위한 구조체

은행계좌를 위한 구조체

멤버접근 연산자(.)를 연속으로 사용

[결과]
구조체크기: 40
[2018, 3, 9]
홍길동 1001 300000.00

구조체 변수간 대입, 동등비교
(structstudent.c)

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <string.h>
int main(void) {
    struct student {
        int snum;          //학번
        char *dept;        //학과 이름
        char name[12];     //학생 이름
    };
    struct student hong = { 201800001, "컴퓨터정보공학과", "홍길동" };
    struct student na = { 201800002 };
    struct student bae = { 201800003 };
```

dept : 다른 문자열 주소를 가질 수는 있지만 복사할 수는 없다.

name : 메모리가 12 byte잡혀 있으므로 문자열을 입력 받거나 복사할 수 있다.

```
[결과]
학생이름: jychoi
[201800001, 컴퓨터정보공학과, 홍길동]
[201800002, 컴퓨터정보공학과, jychoi]
[201800003, 기계공학과, 배상문]
학번이 201800003로 동일합니다.
내용이 같은 구조체입니다.
```

구조체 변수간 대입, 동등비교
(structstudent.c)

```
    printf("학생이름: "); scanf("%s", na.name);
    na.dept = "컴퓨터정보공학과";
    bae.dept = "기계공학과";
    memcpy(bae.name, "배상문", 7);
    strcpy(bae.name, "배상문");
    strcpy_s(bae.name, 7, "배상문");
    printf("[%d, %s, %s]₩n", hong.snum, hong.dept, hong.name);
    printf("[%d, %s, %s]₩n", na.snum, na.dept, na.name);
    printf("[%d, %s, %s]₩n", bae.snum, bae.dept, bae.name);
    struct student one;
    one = bae;
    if (one.snum == bae.snum)
        printf("학번이 %d로 동일합니다.₩n",
    //if ( one == bae ) //오류
    if (one.snum == bae.snum && !strcmp(one.name, bae.name)
        && !strcmp(one.dept, bae.dept))
        printf("내용이 같은 구조체입니다.₩n");
    return 0;
}
```

구조체 변수간 대입 가능! memcpy() 되므로
멤버 중 주소값이 있을 때는 사용 자제.

구조체 변수간 동등비교불가!
멤버끼리 하나씩 비교해야 함,

공용체 (union)

- 동일한 저장 장소에 여러 자료형을 저장하는 방법
- 공용체를 구성하는 멤버들은 한번에 한 멤버로만 저장/참조 가능.
- 구조체선언 방법과 동일 : struct 대신 union을 사용하는 것만 다름.
- 공용체 변수의 크기
  - 멤버 중 가장 큰 자료형의 크기로 정해짐
- 공용체의 초기화
  - 공용체 정의 시 처음 선언한 멤버의 초기값으로만 저장 가능
- 공용체 멤버 접근
  - 구조체와 같이 접근연산자 .를 사용

공용체의 정의와 변수 선언 및 사용 (union.c)

```
#include <stdio.h>
union data {
    char ch;
    int cnt;
    double real;
} data1;
int main(void) {
    union data data2 = { 'A' };
    union data data3 = data2;
    printf("%d %d₩n", sizeof(union data), sizeof(data3));
    data1.ch = 'a';
    printf("%c %d %f₩n", data1.ch, data1.cnt, data1.real);
    data1.cnt = 100;
    printf("%c %d %f₩n", data1.ch, data1.cnt, data1.real);
    data1.real = 3.156759;
    printf("%c %d %f₩n", data1.ch, data1.cnt, data1.real);
    return 0;
}
```

유니온 구조체를 정의하면서 변수 전역변수 data1도 선언한 문장

첫 멤버인 char형으로만 초기화 가능

다른 변수로 초기화 가능

[결과]

```
8 8
a 97 0.000000
d 100 0.000000
N -590162866 3.156759
```

자료형 재정의
(typedef 사용)

- typedef 구문
  - 이미 사용되는 자료 유형을 다른 새로운 자료형 이름으로 재정의 함.
  - typedef int profit;
  - profit을 int와 같은 자료형으로 새롭게 정의하는 문장
- 시스템 간 호환성 유지에 사용
  - int 형이 터보 C++에서는 2바이트, Visual C++에서는 4바이트 사용
  - Visual C++에서 작성할 때 myint로 재정의 후 모든 int 형을 myint형으로 사용
  - 이 소스를 터보 C++에서 컴파일 할 때는 Visual C++의 int형과 호환되는 자료형으로 선언.

```
[Visual C++]
typedef   int   myint;
……
myint salary = 2000000;
```

```
[Turbo C++]
typedef   long   myint;
……
myint salary = 2000000;
```

자료형 재정의 사용
(typedef.c)

```
#include <stdio.h>
typedef unsigned int budget;
int main(void) {
    budget year = 24500000;
    typedef int profit;
    profit  month = 4600000;

    printf("올 예산은 %d, 이달의 이익은 %d 입니다.₩n", year, month);
    return 0;
}

void test(void) {
    budget year = 24500000;
    //profit year;
}
```

새로운 자료형 budget 을 만들고 사용

함수 내부에서 새로운 자료형 선언하여 사용

이 위치에서,
budget 형은 사용 가능하지만
profit형은 사용 불가.

[결과]
올 예산은 24500000, 이달의 이익은 4600000 입니다.

구조체 자료형 재정의 사용
(typedefstruct.c)

```
#include <stdio.h>
struct date {
    int year; int month; int day;
};
typedef struct date date;
int main(void) {
    typedef struct {
        char title[30];        //제목
        char company[30];//제작회사
        date release;          //출시일
    } software;
    software vs = { "비주얼스튜디오", "MS", { 2018, 8, 29 } };
    printf("제품명: %s₩n", vs.title);
    printf("회사 : %s₩n", vs.company);
    printf("출시일: %d. %d. %d₩n",
        vs.release.year, vs.release.month, vs.release.day);
    return 0;
}
```

[결과]
제품명: 비주얼스튜디오 커뮤니티
회사 : MS
조르 : 통합개발환경
018. 8. 29

struct date 형을 간단히 date 형으로 재정의

구조체를 정의하면서 바로 자료형 software 로 정의하기 위한 구문

구조체 포인터의 선언과 사용
(structpointer.c)

```
#include <stdio.h>
struct lecture {
    char name[20];          //강좌명
    int type;//강좌구분 0: 교양, 1: 일반선택, 2: 전공필수, 3: 전공선택
    int credit;             //학점
    int hours;              //시수
};
typedef struct lecture lecture;
char *head[] = { "강좌명", "강좌구분", "학점", "시수" };
char *lectype[] = { "교양", "일반선택", "전공필수", "전공선택" };
```

```
[결과]
구조체크기: 32, 포인터크기: 8

    강좌명      강좌구분    학점    시수
    운영체제    전공필수      3       3
 C프로그래밍    전공선택      3       4
           C    전공선택      3       4
```

구조체 포인터의 선언과 사용
(structpointer.c)

```
int main(void) {
    lecture os = { "운영체제", 2, 3, 3 };
    lecture c = { "C프로그래밍", 3, 3, 4 };
    lecture *p = &os;
    printf("구조체크기: %d, 포인터크기: %d₩n₩n", sizeof(os), sizeof(p));
    printf("%10s %12s %6s %6s₩n", head[0], head[1], head[2], head[3]);
    printf("%12s %10s %5d %5d₩n",
        p->name, lectype[p->type], p->credit, p->hours);
    p = &c;
    printf("%12s %10s %5d %5d₩n",
        (*p).name, lectype[(*p).type], (*p).credit, (*p).hours);
    printf("%12c %10s %5d %5d₩n",
        *c.name, lectype[c.type], c.credit, c.hours);
    return 0;
}
```

-> 연산자는 (*). 연산자와 같음!

[결과]
구조체크기: 32, 포인터크기: 8

강좌명	강좌구분	학점	시수
운영체제	전공필수	2	3

공용체 포인터의 선언과 사용
(unionpointer.c)

```
#include <stdio.h>
int main(void) {
    union data {
        char ch;
        int cnt;
        double real;
    };
    typedef union data udata;
    udata value, *p;
    p = &value;
    p->ch = 'a';
    printf("%c %c₩n", p->ch, (*p).ch);
    p->cnt = 100;
    printf("%d ", p->cnt);
    p->real = 3.14;
    printf("%.2f ₩n", p->real);
    return 0;
}
```

공용체 타입을 선언하고 사용

같은 뜻임.

[결과]
a a
100 3.14

구조체의 배열 사용
(structarray.c)

```
#include <stdio.h>
struct lecture {
    char name[20];          //강좌명
    int type;               //강좌구분
    int credit;             //학점
    int hours;              //시수
};
typedef struct lecture lecture;
char *lectype[] = { "교양", "일반선택", "전공필수", "전공선택" };
char *head[] = { "강좌명", "강좌구분", "학점", "시수" };
```

[결과]
배열크기: 5

강좌명	강좌구분	학점	시수
인간과 사회	교양	2	2
경제학개론	일반선택	3	3
자료구조	전공필수	3	3
모바일프로그래밍	전공필수	3	4

구조체의 배열 사용
(structarray.c)

```
int main(void)  {
    lecture course[] = { { "인간과 사회", 0, 2, 2 },
    { "경제학개론", 1, 3, 3 },
    { "자료구조", 2, 3, 3 },
    { "모바일프로그래밍", 2, 3, 4 },
    { "고급 C프로그래밍", 3, 3, 4 } };
    int arysize = sizeof(course) / sizeof(course[0]);

    printf("배열크기: %d₩n₩n", arysize);
    printf("%12s    %12s %6s %6s₩n", head[0], head[1], head[2], head[3]);
    printf("=============================================₩n");
    for (int i = 0; i < arysize; i++)
        printf("%16s %10s %5d %5d₩n", course[i].name,
        lectype[course[i].type], course[i].credit, course[i].hours);
    return 0;
}
```

구조체 배열의 초기값 : 중괄호를 중첩하여 사용
외부 중괄호 : 배열 초기화
내부 중괄호 : 배열원소인 구조체 초기화를

[ ] 연산자로써 배열 원소인 구조체를 찾은 후
. 연산자로써 구조체 멤버에 접근.