13주차 1차시 구조체 배열

【학습목표】

1. 구조체와 배열에 대해 설명할 수 있다.
2. typedef에 대해 설명할 수 있다.

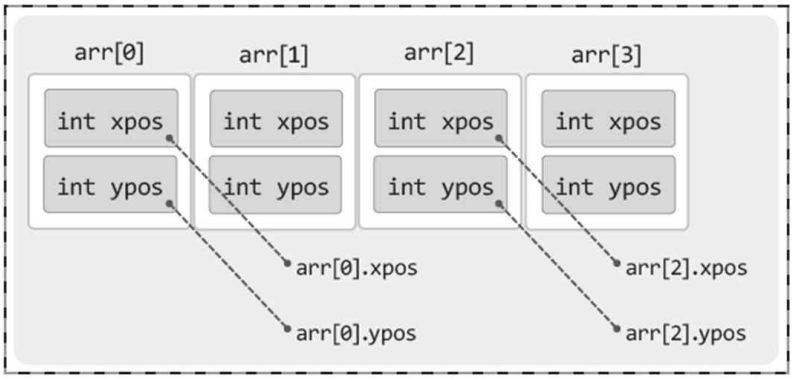
학습내용1 : 구조체와 배열 그리고 포인터

1. 구조체와 배열의 선언과 접근

```
struct point arr[4];
```

길이가 4인 구조체 배열의 선언방법

선언된 배열의 형태



```
struct point
{
    int xpos;
    int ypos;
};

int main(void)
{
    struct point arr[3];
    int i;

    for(i=0; i<3; i++)
    {
        printf("점의 좌표 입력: ");
        scanf("%d %d", &arr[i].xpos, &arr[i].ypos);
    }

    for(i=0; i<3; i++)
        printf("[%d, %d] ", arr[i].xpos, arr[i].ypos);

    return 0;
}
```

```
점의 좌표 입력: 2 4
점의 좌표 입력: 3 6
점의 좌표 입력: 8 9
[2, 4] [3, 6] [8, 9]
```

2. 구조체 배열의 초기화

■ 구조체 변수의 초기화

```
struct person man={"이승기", "010-1212-0001", 21};
```

구조체 변수 하나를 초기화하기 위해서 하나의 중괄호를 사용하듯이

■ 구조체 배열의 초기화

```
struct person arr[3]={
    {"이승기", "010-1212-0001", 21},     // 첫 번째 요소의 초기화
    {"정지영", "010-1313-0002", 22},     // 두 번째 요소의 초기화
    {"한지수", "010-1717-0003", 19}      // 세 번째 요소의 초기화
};
```

구조체 배열을 초기화하기 위해서 배열요소 각각의 초기화 값을 중괄호로 묶어서 표현한다.

3. 구조체 배열의 초기화 예제

```
struct person
{
    char name[20];
    char phoneNum[20];
    int age;
};

int main(void)
{
    struct person arr[3]={
        {"이승기", "010-1212-0001", 21},    // 첫 번째 요소의 초기화
        {"정지영", "010-1313-0002", 22},    // 두 번째 요소의 초기화
        {"한지수", "010-1717-0003", 19}     // 세 번째 요소의 초기화
    };

    int i;
    for(i=0; i<3; i++)
        printf("%s %s %d \n", arr[i].name, arr[i].phoneNum, arr[i].age);

    return 0;
}
```

```
이승기 010-1212-0001 21
정지영 010-1313-0002 22
한지수 010-1717-0003 19
```

4. 구조체 변수와 포인터

■ 구조체 포인터 변수를 대상으로 하는 포인터 연산 및 멤버의 접근방법

```
struct point pos={11, 12};

struct point * pptr=&pos;
        구조체 point의 포인터 변수 선언
(*pptr).xpos=10;
        pptr이 가리키는 구조체변수의 멤버 xpos에 접근 pp

(*pptr).ypos=20;
        tr이 가리키는 구조체 변수의 멤버 ypos에 접근
```

■ 연산자를 기반으로 하는 구조체 변수의 멤버 접근 방법

```
(*pptr).xpos=10;      pptr->xpos=10;

(*pptr).ypos=20;      pptr->ypos=20;
```

5. 구조체 변수와 포인터 관련 예제

```
struct point
{
    int xpos;
    int ypos;
};

int main(void)
{
    struct point pos1={1, 2};
    struct point pos2={100, 200};
    struct point * pptr=&pos1;
    (*pptr).xpos += 4;
    (*pptr).ypos += 5;
    printf("[%d, %d] \n", pptr->xpos, pptr->ypos);
    pptr=&pos2;
    pptr->xpos += 1;
    pptr->ypos += 2;
    printf("[%d, %d] \n", (*pptr).xpos, (*pptr).ypos);
    return 0;
}
```

```
[5, 7]
[101, 202]
```

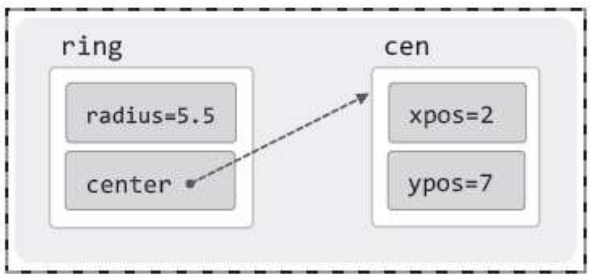
분홍색 안의 연산자는 프로그래머들이 주로 사용하는 연산자이니 연산자 사용에 익숙해지자.

6. 포인터 변수를 구조체의 멤버로 선언하기1

```
struct point
{
    int xpos;
    int ypos;
};
struct circle
{
    double radius;
    struct point * center;
};

int main(void)
{
    struct point cen={2, 7};
    double rad=5.5;
    struct circle ring={rad, &cen};
    printf("원의 반지름: %g \n", ring.radius);
    printf("원의 중심 [%d, %d] \n", (ring.center)->xpos, (ring.center)->ypos);
    return 0;
}
```

구조체 변수의 멤버로 구조체 포인터 변수가 선언될 수 있다




```
원의 반지름: 5.5
원의 중심 [2, 7]
```

7. 포인터 변수를 구조체의 멤버로 선언하기2

```
struct point
{
    int xpos;
    int ypos;
    struct point * ptr;
};

int main(void)
{
    struct point pos1={1, 1};
    struct point pos2={2, 2};
    struct point pos3={3, 3};
    pos1.ptr = &pos2;     // pos1과 pos2를 연결
    pos2.ptr = &pos3;     // pos2와 pos3를 연결
    pos3.ptr = &pos1;     // pos3를 pos1과 연결
    printf("점의 연결관계... \n");
    printf("[%d, %d]와(과) [%d, %d] 연결 \n",
        pos1.xpos, pos1.ypos, pos1.ptr->xpos, pos1.ptr->ypos);
    printf("[%d, %d]와(과) [%d, %d] 연결 \n",
        pos2.xpos, pos2.ypos, pos2.ptr->xpos, pos2.ptr->ypos);
    printf("[%d, %d]와(과) [%d, %d] 연결 \n",
        pos3.xpos, pos3.ypos, pos3.ptr->xpos, pos3.ptr->ypos);
    return 0;
}
```

type형 구조체 변수의 멤버로 type형 포인터 변수를 둘 수있다.

```
점의 연결관계...
[1, 1]와(과) [2, 2] 연결
[2, 2]와(과) [3, 3] 연결
[3, 3]와(과) [1, 1] 연결
```

8. 구조체 변수와 첫 번째 멤버의 주소 값

```
struct point
{
    int xpos;
    int ypos;
};
struct person
{
    char name[20];
    char phoneNum[20];
    int age;
};
int main(void)
{
    struct point pos={10, 20};
    struct person man={"이승기", "010-1212-0001", 21};

    printf("%p %p \n", &pos, &pos.xpos);
    printf("%p %p \n", &man, man.name);
    return 0;
}
```

```
003EF7B8 003EF7B8
003EF784 003EF784
```

구조체 변수의 주소값과 구조체 변수의 첫 번째 멤버의 주소값은 일치한다.
응용 프로그램 분야에서는 이 사실을 이용해서 프로그램을 작성하기도 한다.

학습내용2 : 구조체 정의와 typedef 선언

1. typedef 선언

typedef int INT; — 자료형의 이름 *int*에 *INT*라는 이름을 추가로 붙여줍니다.

위의 *typedef* 선언으로 인해서 !!!

INT num; *int num;* 과 동일한 선언

INT * ptr; *int * ptr;* 과 동일한 선언

```
typedef int INT;
typedef int * PTR_INT;

typedef unsigned int UINT;
typedef unsigned int * PTR_UINT;

typedef unsigned char UCHAR;
typedef unsigned char * PTR_UCHAR;

int main(void)
{
    INT num1 = 120;             // int num1 = 120;
    PTR_INT pnum1 = &num1;      // int * pnum1 = &num1;

    UINT num2 = 190;            // unsigned int num2 = 190;
    PTR_UINT pnum2 = &num2;     // unsigned int * pnum2 = &num2;

    UCHAR ch = 'Z';             // unsigned char ch = 'Z';
    PTR_UCHAR pch = &ch;        // unsigned char * pch = &ch;

    printf("%d, %u, %c \n", *pnum1, *pnum2, *pch);
    return 0;
}
```

```
120, 190, Z
```

정의되는 이름들

```
새로 부여된 이름      대상 자료형
INT                  int
PTR_INT              int *
UINT                 unsigned int
PTR_UINT             unsigned int *
UCHAR                unsigned char
PTR_UCHAR            unsigned char *
```

2. 구조체 정의와 typedef 선언

```
struct point
{
    int xpos;
    int ypos;
};

typedef struct point Point;
```

구조체 point 정의 후 struct point에 Point라는 이름을 부여하기 위한 typedef 선언 추가!

위의 것을 합친 형태는 아래와 같다.

```
typedef struct point
{
    int xpos;
    int ypos;
} Point;
```

구조체 point의 정의와 Point에 대한 typedef 선언을 한데 묶은 형태

3. 구조체 정의와 typedef 선언 관련 예제

```
struct point
{
    int xpos;
    int ypos;
};

typedef struct point Point;
```

구조체 *point*의 정의와 *typedef* 선언

```
typedef struct person
{
    char name[20];
    char phoneNum[20];
    int age;
} Person;
```

구조체 *person*의 정의와 *Person*이라는 이름의 *typedef* 선언을 하나로!

```
int main(void)
{
    Point pos={10, 20};
    Person man={"이승기", "010-1212-0001", 21};
    printf("%d %d \n", pos.xpos, pos.ypos);
    printf("%s %s %d \n", man.name, man.phoneNum, man.age);
    return 0;
}
```

```
10 20
이승기 010-1212-0001 21
```

4. 구조체의 이름 생략

```
typedef struct person
{
    char name[20];
    char phoneNum[20];
    int age;
} Person;
```

⇨

```
typedef struct
{
    char name[20];
    char phoneNum[20];
    int age;
} Person;
```

typedef 선언으로 인해서 새로운 이름 Person이 정의되었으니, 구조체의 이름 persons은 큰 의미가 없다 따라서 이렇듯 구조체의 이름을 생략하는 것도 가능하다.

【학습정리】

1. 구조체의 배열을 선언하는 방법은 일반적인 배열의 선언방법과 동일하다.

2. typedef 선언에 있어서 새로운 이름의 부여는 가장 마지막에 등장하는 단어를 중심으로 이루어진다.

3. typedef 선언은 기존에 존재하는 자료형의 이름에 새 이름을 부여하는 것을 목적으로 하는 선언이다.