3주차 3차시 구조체와 재귀호출

【학습목표】

1. 조체 자료형을 이해하고 구조체의 구현 방법을 사용할 수 있다.
2. 귀호출의 의미를 이해하고 구현 방법을 설명할 수 있다.

학습내용1 : 구조체의 사용

1. 구조체

- 배열처럼 여러 개의 데이터를 그룹으로 묶어서 하나의 자료형으로 정의하고 사용하는 자료형
- 배열은 같은 자료형 만을 그룹으로 묶을 수 있지만, 구조체는 서로 다른 자료형을 그룹으로 묶을 수 있어 복잡한 자료형태를 정의하는데 유용

- 레코드(record) : 자료를 체계적으로 관리하기 위해서 구성한 일정한 단위 형식
- 필드(field) : 레코드를 구성하는 하위 항목
- 파일(file) : 여러 레코드가 모여서 하나의 파일을 구성

[그림 3-41] 필드와 레코드, 파일

2. 구조체 선언

- 여러 자료형의 변수들을 그룹으로 묶어서 하나의 자료형으로 선언
- 구조체는 구조체 이름, 자료형, 데이터 항목으로 구성
- 구조체의 이름 : 구조체로 정의하는 새로운 자료형의 이름
- 항목 : 구조체를 구성하는 내부 변수들의 이름
    - 구조체의 항목은 배열의 각 배열요소에 해당
    - 배열요소는 모든 같은 자료형으로 되어있으므로 배열요소에 대한 선언없이 사용
    - 구조체에서는 각 항목이 다른 자료형을 가질 수 있기 때문에 항목별로 자료형과 항목이름(변수이름)을 선언해야 함

* 구조체 선언 형식

```
struct 구조체이름 {
        자료형 항목 1;
        자료형 항목 2;
          :
        자료형 항목 n;
    };
```

* 사용할 구조체의 모양을 정의한 것뿐이므로 사용할 구조체 변수를 다시 선언해야 함

struct *구조체이름* 구조체변수;

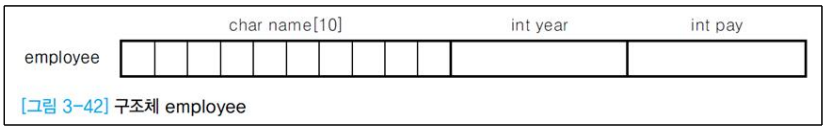
- 선언된 구조체 변수는 일반변수와 똑같이 취급하고 사용

* 구조체 사용 예 : 직원 관리 프로그램

- 구조체 선언

```
struct employee{
        char name[10];
        int year;
        int pay;
};
```

* 구조체 employee의 구조



[그림 3-42] 구조체 employee

* 구조체 변수 선언

- struct employee Lee, Kim, Park;

[그림 3-43] 구조체 employee에 대한 구조체 변수의 구조

* 구조체 변수의 선언 방법

[표 3-2] 구조체 변수의 선언 방법

①	②	③

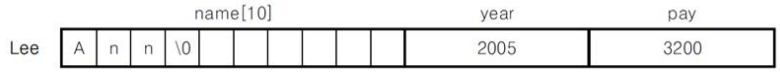
struct employee{
char name[10];
int year;
int pay;
};
struct employee Lee;
	struct employee{
	char name[10];
	int year;
	int pay;
	} Lee;
		struct {
		char name[10];
		int year;
		int pay;
		} Lee;

학습내용2 : 구조체의 연산

1. 구조체의 초기화

- 구조체는 여러 개의 내부 항목으로 구성되어 있으므로 내부 항목의 자료형과 개수, 순서에 맞추어 초기값을 지정하되 두 개 이상의 값을 나열해야 하므로 중괄호를 사용

```
struct employee Lee = { "Ann", 2005, 3200 };  // 구조체 변수의 초기화
```



[그림 3-44] 구조체 변수 Lee의 초기화된 구조

2. 구조체 데이터 항목의 참조

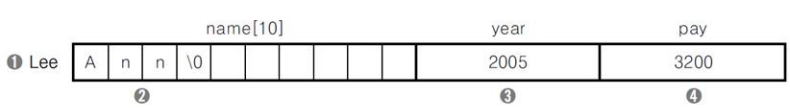
- 구조체 변수에 있는 각 데이터 항목을 참조하기 위해서는 구조체 연산자 사용

[표 3-3] 구조체 연산자

구조체 연산자	사용 의미
. (점 연산자)	구조체 변수의 데이터 항목을 지정한다.
→ (화살표 연산자)	구조체형 포인터에서 포인터가 가리키는 구조체 변수의 데이터 항목을 지정한다.

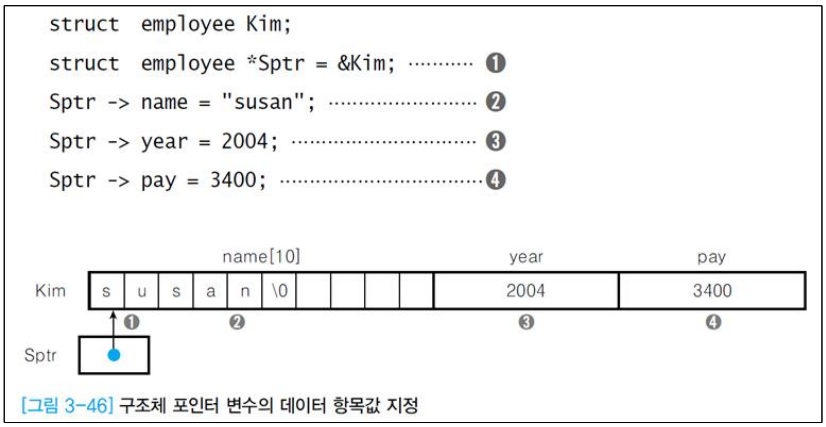
* 점 연산자 : •
- 구조체 변수에 있는 데이터 항목을 개별적으로 지정할 때 사용
- 예)

```
struct employee Lee; ····· ❶
Lee.name = "Ann"; ····· ❷
Lee.year = 2005; ····· ❸
Lee.pay = 3200; ····· ❹
```



[그림 3-45] 점 연산자를 이용한 데이터 항목 값 지정

* 화살표 연산자 : ->
- 구조체 포인터 변수에서 포인터가 가리키는 구조체 변수의 데이터 항목을 지정하기 위해서 화살표 연산자를 사용



[그림 3-46] 구조체 포인터 변수의 데이터 항목값 지정

3. 구조체의 연산

* 데이터 항목 참조 연산
- 점 연사자와 화살표 연사자를 이용하여 구조체의 데이터 항목을 개별적으로 참조
- 예)

```
struct employee Lee;
struct employee *Sptr;
Sptr = &Lee;
Lee.year = 2005;
Sptr->pay = 3000;
Sptr->name = “Ann”;
```

* 구조체 복사 연산
- 같은 구조체에 대한 구조체 변수 간에는 내용을 한 번에 복사하는 연산 사용 가능
- 예)

```
struct employee Lee, Kim, team[5];  경우

Lee = Kim;
Lee = team[2];
team[2] = team[3];
```

* 구조체 변수의 주소 구하기 연산
- 포인터의 주소 연산자를 사용하여 구조체 변수의 주소를 구하거나 구조체 변수가 배열인 경우 배열의 특성에 따라 구조체 배열 변수의 이름에서 주소를 구함

- 예)

```
struct employee Lee, team[5];
struct employee *Sptr1, *Sptr2;  로 선언된 경우

Sptr1 = &Lee;
Sptr2 = team;
```

학습내용3 : 재귀호출

1. 의미

- 자기 자신을 호출하여 순환 수행하는 것
- 함수에서 실행하는 작업의 특성상 일반적인 호출방식보다 재귀호출방식을 사용하여 함수를 만들면 프로그램의 크기를 줄이고 간단하게 작성 가능

2. 재귀호출의 예: factorial

- n에 대한 factorial 연산은 1부터 n까지의 모든 자연수를 곱하여 구하는 연산

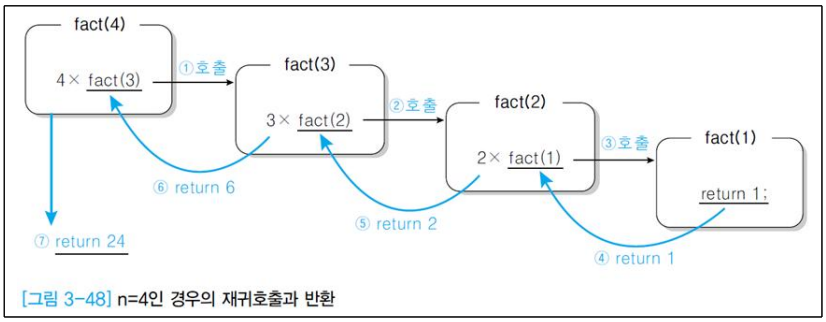
```
n! = n × (n-1)!
(n-1)! = (n-1) × (n-2)!
(n-2)! = (n-2) × (n-3)!
 ....
2! = 2 × 1!
1! = 1
```

- 마지막에 구한 하위값을 이용하여 상위값을 구하는 작업을 반복 수행

- 재귀호출을 사용하여 함수를 작성

```
long int fact(int n) {
    if (n<=1)
        return (1);
    else
        return (n * fact(n-1));
 }
```

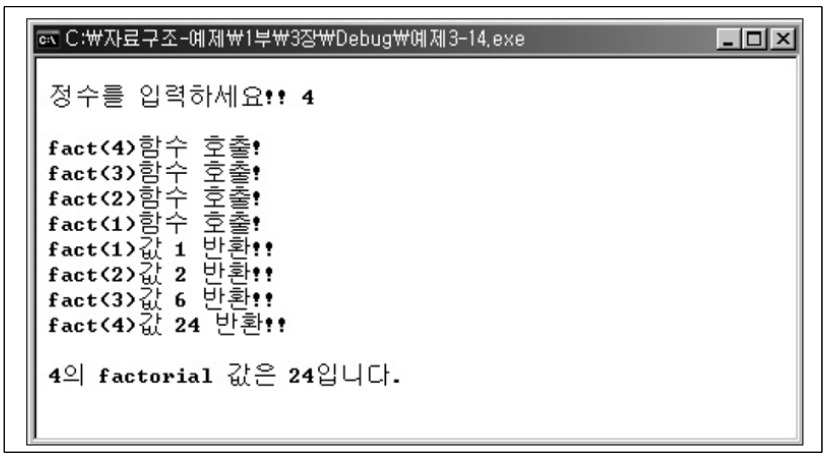
3. factorial 함수에서 n=4인 경우 실행



[그림 3-48] n=4인 경우의 재귀호출과 반환

4. 예제 3-14 : 재귀호출을 이용한 factorial 프로그램

* 실행화면



【학습정리】

1. 구조체란 서로 다른 자료형을 그룹으로 묶어서 새로운 자료형을 정의할 수 있다.

2. 구조체 연산자로는 구조체 변수의 데이터 항목을 지정하는 점 연산자와 구조체형 포인터에서 포인터가 가리키는 데이터 항목을 지정하는 화살표 연산자가 있다.

3. 재귀호출은 자기 자신을 호출하여 순환 수행되는 것으로 프로그램 크기를 줄이고 간단하게 작성할 수 있으나, 잘못 사용시 오히려 프로그램의 속도가 느려지고 무한반복이 발생될 수 있으므로 사용시 주의하여야 한다.