11\_구조체, 공용체 , 열거형

학습목표

구조체를 이해할 수 있다.

구조체의 선언과 초기화 방법을 이해할 수 있다.

구조체의 선언과 구조 변수의 차이점을 알 수 있다.

구조체를 포인터로 가르킬 수 있다.

공용체를 이해하고 사용할 수 있다.

Typedef을 이용하여 사용자 정의 자료형을 만들 수 있다.

1. 구조체란 무엇인가?

우리는 지금가지 동일한 종류의 데이터를 하나로 묶기 위하여 배열을 사용하였다. 그러나 만약 서로 다른 종류의 데이터들을 하나로 묶어야 된다면 어떻게 할 것인가? 이러한 경우에 사용할 수 있는 방법이 구조체(structure)이다. 구조체는 프로그래머가 여러 개의 기초 자료형들을 묶어서 새로운 자료형을 정의할 수 있는 방법이다. 구조체는 객체지향 프로그래밍에서 말하는 클래스의 모체가 된다.

구조체의 태그는 구조체와 구조체를 구별하기 위해서 구조체에 붙여진 이름이다. 구조체 태그는 변수가 아니다. 중괄호 안에 쓰인 변수들을 구조체 멤버라고 부른다. 구조체 멤버는 구조체에 포함되는 변수이다. 구조체 정의가 끝나면 반드시 세미콜론을 붙여주어야 한다. 구조체를 정의하는 것도 하나의 문장에 해당하기 때문이다.

여기서 주의할 점은 위의 구조체 정의는 변수 선언이 아니라는 점이다. 구조체 정의는 구조체 안에 어떤 변수들이 들어간다는 것만 말해주는 것이다. 구조체는 와플을 찍어내는 와플틀과 같은 존재이다.

구조체 정의를 마쳤다면 구조체 변수를 생성해볼 차례이다. 구조체 변수는 struct ‘구조체 이름’ 을 앞에 붙여서 변수들을 생성하면 된다.

Ex) struct student s1; << 구조체 변수 선언

S1이라는 변수 안에는 구조체의 멤버인 number, name[10], grade가 들어 있다. S1에는 실제 메모리 공간이 할당되며 s1이 차지하는 메모리 공간의 크기는 각 멤버들의 크기를 합치면 알 수 있는데 대략 4+10+8 = 22바이트가 된다. 컴파일러는 액세스 속도를 빠르게 하기 위하여 더 많은 메모리를 할당하는 경우도 있으므로 sizeof 연산자를 이용하는 편이 정확하다.

구조체 변수의 초기화는 배열과 비슷하다. 중괄호 안에 나열하면 된다.

구조체에서 중요한 것은 멤버 변수들을 참조하는 것이다. 구조체 변수를 통하여 멤버들을 참조하려면 특별한 연산자가 필요하다. 멤버 연산자(.)를 이용하여 액세스할 수 있다. 구조체 변수 이름을 쓰고 멤버 연산자를 찍은 다음, 멤버 변수의 이름을 써주면 된다.

S1.number = 2017270194; << 예시

만약 멤버가 문자열이라면 멤버에 값을 대입할 때, strcpy()를 사용해야 한다.

Strcpy(s1.name, “KIM”);

새로운 초기화 방법

-struct.c 참고하기

1. 구조체 변수의 대입과 비교

다른 자료형의 변수들은 대입 연산, 비교 연산들을 할 수 있다. 구조체 변수는 다른 구조체 변수에 대입하는 것은 가능하다. 즉 하나의 구조체 변수에 들어 있는 자료들을 다른 구조체 변수로 복사할 수 있다. 이것이 개별 변수들을 사용하는 것보다 구조체를 사용하는 것이 편리한 이유이다. 그러나 구조체 변수와 구조체 변수를 서로 비교하는 것은 허용되지 않는다. 구조체 변수를 비교하려면 멤버마다 별도의 비교 수식을 적어주어야 한다.

1. 구조체의 배열

여러 개의 구조체가 필요할 때, 구조체의 배열을 사용하게 된다. 구조체의 배열이란 배열 원소가 구조체인 것을 말한다. 즉 구조체가 여러 개 모인 것이다. 구조체 배열의 선언은 배열 선언과 별반 다르지 않다.

Struct student list[100]; << 100개의 student 구조체 만들기

구조체 배열의 초기화도 가능하다. 다만 배열 초기화 안에 구조체 초기화가 들어가야 하므로 중괄호 안에 또 중괄호가 필요하다.

Struct.c 참조

1. 구조체와 함수

구조체는 함수의 인수로도 사용이 가능하고 함수에서 반환 값으로 반환될 수 있다. 구조체가 인수나 반환 값으로 사용될 때는 “값에 의한 호출” 원칙이 적용된다. 즉 구조체 변수의 모든 내용이 복사되어 함수로 전달되고 반환된다. 복사본이 인수로 전달되는 것이므로 원본 구조체는 영향을 받지 않는다. 단점으로 구조체 크기가 클 경우 상당한 시간이 소요된다.

1. 구조체를 함수의 인수로 넘기는 방법

다른 기본 자료형을 넘길 때와 크게 다르지 않다. Structfunction.c 참조

1. 구조체를 함수의 반환 값으로 넘기는 방법

구조체를 반환하기 위해서는 반환 값의 형을 구조체로 표시해주면 된다.

구조체를 반환하게 하면 함수가 하나 이상의 값을 반환할 수 있게 된다.

구조체의 포인터를 함수의 인수로 전달하는 경우: 시간과 공간을 절약할 수 있다, 원본 훼손 가능성이 있기 때문에 const을 붙여 값이 변경이 없도록 해야한다. Structfunction.c참조

1. 구조체와 포인터

변수를 가리키는 포인터를 만들 수 있는 것처럼 구조체를 가리키는 포인터도 만들 수 있다.

Struct student \*p;

포인터 p는 student 구조체를 가리킬 수 있는 포인터이다. 구조체 변수 p와 구조체 변수를 연결하려면 구조체 변수의 주소를 추출하여 대입해주면 된다.

P = &s;;

포인터를 이용하여 구조체의 멤버를 액세스하려면 (\*p).number 처럼 하면 된다. \*p가 구조체를 가리키고 여기에 마침표 기호를 붙이면 멤버가 되는 것이다. 괄호를 사용하는 이유는 (.)연산자가 (\*)연산자보다 우선순위가 높기 때문이다.

포인터를 이용하여 구조체의 멤버를 가리키는 것은 프로그램에서 자주 등장하기 때문에 이것을 위한 특수한 연산자 ->가 있다. ->연산자는 간접 멤버 연산자(indirect membership operator)라고 불리는 것으로 구조체 포인터를 이용하여 멤버에 접근하기 위하여 사용된다. 의미는 완전히 일치하지만 -> 연산자가 훨씬 간편하여 많이 사용된다.

p->number의 의미는 포인터 p가 가리키는 구조체의 멤버 number라는 의미이다.

1. 공용체

C언어에서는 같은 메모리 영역을 여러 개의 변수들이 공유할 수 있게 하는 기능이 있다. 이것은 공용체(union)라 불리운다. 같은 메모리 영역을 여러 개의 변수가 공유하도록 하는 것은 메모리를 절약하기 위해서이다. 공용체의 사용법은 구조체와 아주 비슷하다. 다만 공용체는 멤버들이 같은 공간을 공유하기 때문에 동시에 모든 멤버 변수들의 값을 저장할 수는 없으며 한 순간에는 하나의 멤버만 존재할 수 있다. 공용체는 가장 큰 멤버의 크기만큼의 메모리가 할당된다. Union.c참조

여기서 주의할 점은 선택된 멤버에 따라 저장된 값이 다르게 해석된다는 점이다. 따라서 올바르게 값을 저장하고 사용해야 한다. 공용체는 주로 동일한 메모리 영역에 대하여 여러 가지 해석을 요구하는 응용 프로그램에서 유용하다.

1. 열거형

열거형(enumeration)이란 변수가 가질 수 있는 값들을 미리 열거해 놓은 자료형이다.

열거형은 enum이라는 키워드를 사용하여 만들어진다.

Enum levels = { low, medium, high }; // low = 0, medium = 1, high = 2

//level 안에 들어 있는 식별자들은 컴파일러에 의하여 0부터 1씩 증가하는 값으로 설정된다. 식별자에 =값을 붙이면 값을 임의로 설정할 수 있다.

열거형도 사용자가 새로운 자료형을 정의하는 방법의 하나이다. 구조체와 형식은 비슷하지만 중괄호 안에 나열된 값 중에서 하나의 값만 가질 수 있다. 위의 문장은 열거형 자체의 정의일 뿐이지 변수가 선언된 것은 아니다.

Enum levels English;

English = high;

English는 열거형 변수로서 열거형 levels에 정의된 값들만 가질 수 있다.

1. Typedef

구조채는 사용자가 기존의 자료형들을 모아서 새로운 자료형을 정의한다고 볼 수 있다. 그러나 기존의 자료형과는 다르게 항상 앞에 struct를 써주어야 한다. C언어에서는 이것을 극복할 수 있는 방법을 제공한다. 바로 typedef를 사용하는 것이다. Typedef은 말 그대로 새로운 자료형(type)을 정의(define)하는 것이다. 이 키워드는 C의 기본 자료형을 확장시키는 역할을 한다.

Typedef의 사용 형식

Typedef old\_type new\_type;

Typedef unsigned char BYTE;

Old-type은 기존의 자료형이고 new\_type은 새롭게 정의하려고 하는 자료형의 이름이다. 위의 문장은 새로운 자료형 new\_type을 정의하는 것으로 그 내용은 old\_type과 같다는 의미이다.

1. 구조체로 새로운 자료형 만들기

기존의 자료형에 새로운 이름을 붙이는 것은 별다른 이점이 없어 보인다. 그러나 typedef문은 상당히 복잡한 데이터의 형식도 새로운 자료형으로 만들 수 있는 능력이 있다. 또한 구조체와 같이 사용하여 한 번에 정의할 수도 있다.