190415

**\*자료형에게 가명 붙여 주기(typedef)**

의의) 코드를 유지 및 관리 하기 용이하다.

주의사항)

1. 같은 구조체 간의 대입 가능함.

2. 구조체 정의부에서 copy initialization한다는 것은 Default Value를 정해준다는 의미.

Syntax)

typedef [오브젝트] [새\_이름];

ㄴ 내 프로그램에서 ‘새\_이름’은 ‘오브젝트’로 쓰겠다. 라는 의미

가명 붙여주는 2가지 방식

1. typedef를 이용

typedef double Alias\_t;

2. using에 Assignment를 이용

using Alias\_t = double;

**\*구조체(Structure)**

정해진 형태의 자료형 들을 하나로 묶어 표현하고 싶었다.

Syntax 1) Definition

Struct Person //Person은 이때 하나의 사용자 정의 자료형이 된다.

{

Double weight;

Int age;

};

Syntax 2) Declaration

Person me;

Syntax 3) Initialization

Tip) Uniform Initialization이 유용하다.

me = {70.5, 24}; //weight, age

. : 멤버 선택 연산자(Member Selection Operator)

**\*[중요!!] 메모리 얼라인먼트(Memory Alignment)**

의의) 컴파일러가 메모리의 효율을 좋게 하기 위해서 구조체 변수의 최대 자료형의 크기를 기준으로 정렬하게 된다.

데이터를 저장하고 가져오는데 1word=4byte=32bit 단위로 가져와야 CPU효율이 좋다.

다만, 이런 설계구조를 가지면서 생기는 문제점이 padding이 있다.

Ex) 다음과 같은 구조체 정의를 했을 때 메모리 구조는 다음과 같다. Padding이 들어가는 이유는 cnt인 int형 데이터 타입 기준으로 메모리가 정렬되기 때문이다.

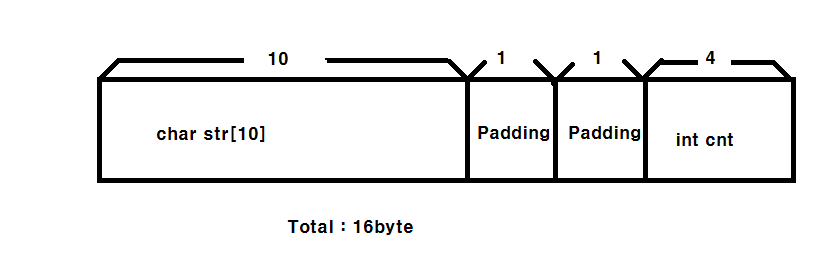
Struct \_my\_st\_b

{

Char str[10];

Int cnt;

};



이와 같이, 메모리 정렬 구조 상 빈 메모리 공간을 잡고 있는 걸 padding 이라고 한다.