2장 C++ 함수와 네임스페이스.

Alt+8 (디스어셈블리 보기)

* 디폴트 매개변수.

피호출 함수의 "형식인수"

(void TestFunc (int a, int b) //callee (main이 caller))

↳ 호출자에서 실인수 a, b 지정 필요. ↳ default 값.

↳ Void TestFunc (int a , int b=10) ⇒ C++에서는 빌드 O

but, 호출자 실인수 적용: 좌 → 우.

↳ 함수 선언 main 앞, 정의 main 뒤일 때, default parameter는 선언에서.

↳ 매개변수가 여러개인 경우 중간에 있는 디폴트값 (생략불가).

오른쪽부터 쭉 지정해야 함.

⇒ 호출자 코드만 보고는 원형을 짐작할 수 없다. / caller에서 모호성 발생.

C++ → 모호성 탈피 중요: Default parameter 자제.

↳ 다형성: 여러개가 공존 (단일함수 다중구현 가능)

⇒ 유연성 ↑ / 사용에 대한 주의 필요.

[override - 재정의
[overload - 다중정의

☆ 모호성: 헷갈림, 유지보수 어려움, 문제.

↳ 디폴트 parameter와 함께 쓰였을 때 함수 특정이 불가할 수 있음 <제거 요망>

* 함수 다중정의

: 하나의 함수(혹은 클래스 등)가 여러 의미를 동시에 갖는 것. <C++ compile : name mangling>

↳ 중복불가.

↳ 내부적 → 중복함수 X

↳ 다중정의를 가능케 하는 기능

extern "C"

↳ 맹글링 제외

↳ C언어 형식.

☆ 함수템플릿 ☆

~~typename~~ template <typename T>

T Add (T a, T b) 와 같이 선언가능. (T를 자료형으로 사용)

템플릿 [함수 / 클래스.] → 찍어낸다.

형식이 다르면

(형식)을 기술할 자리를 변수같이 설정 → caller에서 type 결정 / 호출시에 함수 여러개 호출

* 인라인 함수. p111. 독C강의 전처리기 / 16장, 15장.

↳ "일반함수 + 매크로"의 특성.

↳ 컴파일 최적화: 자동 인라인 변환.

* 네임스페이스. (소속)

↳ 전역변수일 때 중요.

(함수

namespace Test { } ⇒ Test :: a.

↳ 범위지정 연산자.

↳ 범위, 소속.

Q. 클래스랑 다른점...? 생성자 차이인가?

↳ 호출자에서 소속명시 필요.

motemote

* Using namespace ~
네임스페이스 지정 생략 가능.
↳ 중첩 → 네임스페이스 안에 네임스페이스 생성 可
⇒ 여러사람 개발 시 범위나 분야에 따라 같은 이름이 존재할 수 있게 함.
전역 → 개념상 무소속.
[Test() - 묵시적 전역
:: Test() - 명시적 전역. ⇒ 권장!

*식별자 검색순서☆

<전역함수인 경우>
① 현재 블록 범위.
② 현재 블록 범위를 포함하고 있는 상위 블록 범위.
③ 가장 최근 선언된 전역 변수나 함수.

※ 전역 ~~변수함수~~ 변수 경우
선언순서 >> 네임스페이스.

<클래스 메소드 인 경우>
① 현재 블록 범위
② 현재 블록 범위를 포함하고 있는 상위 블록 범위.
③ 클래스의 멤버
④ 부모 클래스의 멤버
⑤ 가장 최근에 선언된 전역변수나 함수.
⑥ 호출자 코드가 속한 네임스페이스의 상위 네임스페이스.
⑦ using 선언된 네임스페이스 or 전역 네임스페이스.
↳ 두곳에 중복된 식별자 존재시 컴파일 오류.