1. 이름공간(namespace)

* 특정 이름 공간의 특정 함수만 사용할 경우

Ex) using NP::func; (NP 이름공간 안에 있는 func()를 기본적으로 사용할 것이다)

* 특정 이름 공간의 함수를 모두 사용할 경우

Ex) using namespace NP (NP 이름공간 안에 있는 모든 함수를 사용할 것이다.)

* 이름이 없는 이름공간을 사용할 경우 이름공간 내에서의 변수,함수는 이름공간 내에서만 사용 가능하다.

씹어먹는 C++ 강좌의 저자는 using namespace 사용을 지양하고 std::cout 처럼 이름공간을 명시하여 같은 이름 충돌을 보호하는 것이 좋다고 한다. std와 같은 이름공간에는 수많은 함수가 있기 때문에.

1. 변수명 짓기 TIP

* 낙타식(NumOfPeople)이나 언더바(num\_of\_people) 둘 다 가능하지만 혼합하면 안됨. 변수명이 길어져도 좋으니 축약하는 것보단 이해하기 쉽게 만드는 것이 중요.

1. 레퍼런스(참조, &)

* 정의할 때 반드시 초기화 해야함, 주소를 저장하는 개념이 아니라 ‘별명’처럼 초기화된 변수와 동일한 취급을 함. 변수처럼 메모리를 따로 차지하지 않음.
* 레퍼런스는 기본적으로 상수를 참조할 수 없으나 const 키워드를 붙이면 가능하다.
* 레퍼런스의 레퍼런스, 레퍼런스의 배열, 레퍼런스의 포인터는 존재할 수 없다.

Ex) const int &num = 5; // const를 빼면 불가능

* 배열의 레퍼런스는 가능하다.

Ex) int (&ref)[3][2] = arr; // [3][2]크기를 가지는 배열의 레퍼런스

1. 변수 선언 TIP

* 바뀌지 않을 값에 const를 넣어 상수로 선언하는 습관을 들이자.

1. 포인터

* 배열의 이름은 배열의 0번째 원소를 가리키는 상수 포인터다. 상수 포인터란 const 키워드 위치에 따라 값이나 주소를 수정할 수 없는 포인터를 뜻한다.

1. 메모리 할당

* 배열의 크기(메모리)는 컴파일 시간에 확정되야한다.
* Heap은 사용자가 자유롭게 할당하거나 해제할 수 있는 메모리의 부분이다.
* C언어에서의 동적 할당은 malloc/free, C++언어에서의 동적 할당은 new/delete

1. 클래스

* 추상화(Abstraction)는 현실에 존재하는 것을 객체로 나타내는 것이다.
* 인스턴스(Instance)는 클래스를 통해 만들어진 객체이다.
* 캡슐화(Encapsulation)는 인스턴스 변수의 값을 외부에서 바꿀 수 없고 인스턴스의 메소드를 통해 간접적으로 바꾸는 것을 뜻한다. 내부적으로 처리되는 것들을 클래스 사용자가 알 필요가 없게 되는 장점이 있다.
* 클래스의 변수와 함수는 멤버 변수/ 멤버 함수라고 부르며 실체가 없는 것이고 인스턴스로 생성된 객체의 변수와 함수는 인스턴스 변수/. 인스턴스 함수라고 부른다.

Ex) Class inst = new Class(); // 가능

Ex) Class inst; // 가능

Ex) Class inst(a,b,c) // 가능

Ex) Class inst(); // 오류, 반환타입이 Class이고 인자가 없는 함수를 정의하게 됨.

1. 멤버 이니셜라이저

* 선언과 동시에 초기화 되어야하는 상수, 레퍼런스를 클래스의 멤버로 둘 때, 반드시 멤버 이니셜라이저를 이용해야한다.

1. 메모리 구조

메모리는 코드, 데이터, 힙, 스택 영역으로 나누어진다.

* CODE(코드): 프로그램의 코드가 저장, 저장된 명령어를 순차적으로 처리
* DATA(데이터): global(전역) 변수와 static(정적) 변수가 할당된다. 프로그램 시작과 함께 할당되고 프로그램이 종료되면 소멸.
* STACK(스택): local(지역) 변수와 parameter(매개 변수)가 할당된다. 컴파일 시간에 크기가 결정된다. 함수가 호출될 때 할당, 함수가 종료되면 소멸. 스택 영역에서 함수의 호출 정보를 저장하는 stack frame(스택 프레임)은 메모리의 높은 주소부터 낮은 주소로 차례대로 할당된다.
* HEAP(힙): 사용자가 직접 관리하는 영역. 동적으로 할당, 해제. 메모리의 낮은 주소부터 높은 주소로 차례대로 할당된다.

1. 암시적 변환(implicit conversion)

사용자가 직접 변환하지 않아도 컴파일러가 자동으로 형 변환을 하는 경우. Explicit(명시적) 키워드를 통해 암시적 변환을 막을 수 있다.

1. 변이 가능 키워드(mutable)

const 함수 내에서도 해당 변수는 변경이 가능하도록 만들어 주는 키워드.