#include <iostream>

전처리기(c++ Preprocessor)에 대한 지시문으로서 컴파일하기 전에<iostream>헤더 파일을 읽어 c++소스파일 안에 삽입할 것을 지시한다.

cout는 c++표준 출력 스트림 객체.

<< 연산자는 스트림 삽입 연산자로 불리며, 오른쪽 피 연산자 데이터를 왼쪽 스트림 객체에 삽입한다.

namespace :: 이름 이름 충돌방지를 위해 고유한 이름 공간을 생성가능함.

std:: 는 표준 이름 공간으로 모든 c++표준 라이브러리는 std이름 공간에 만들어져있다. 이를 생략하고자 한다면, ‘using namespace std;’를 사용하면 된다.

cin는 c++표준 입력 스트림 객체.

>>연산자는 스트림 추출 연산자로 불리며 왼쪽 피 연산자인 스트림 객체로부터 데이터를 읽어 오른쪽 피 연산자에 지정된 변수에 삽입한다.

c++클래스(class)란 객체를 정의하는 틀 혹은 설계도로서, 클래스에 멤버 변수와 멤버 함수를 선언한다.  
멤버 함수가 클래스 내부에 정의되면 자동적으로 인라인(inline) 함수가 된다.

멤버 함수가 클래스 외부에 정의되면 일반적인 함수와 동일하게 호출한다.

c++객체는 c++클래스라는 틀에서 찍어내어, 멤버 변수 메모리와 멤버 함수 코드를 실제 가지고 c++프로그램이 실행되는 동안 실존하는 실체 혹은 인스턴스(instance)이다.

접근 지정자가 선언되면, 다른 접근 지정자로 선언될 때까지 모든 맴버에 대해 적용된다.

public외의 private, protected 등 총 3가지 종류가 있다

public 접근 지정은 클래스 외부로부터의 접근을 허용한다.

private 선언은 클래스 외부에서 접근할 수 없다.

protected 접근 지정으로 선언된 멤버로서, 클래스 내의 멤버 함수와 이 클래스를 상속받은 파생 클래스의 멤버 함수에게만 접근이 허용된다.

객체는 클래스 모양과 기능을 그대로 간직한 실체로서 이름을 가진다. 기본 타입의 변수를 선언하는 것과 같은 방법으로 객체를 생성하고 이름을 붙인다.

생성자는 객체를 초기화 한다. 여러 개의 생성자를 둘 수 있으나, 이 중 하나만 실행된다.

생성자의 목적은 객체가 생성될 때 필요한 초기 작업을 하기 위함이다.

한번만 실행되며, 이름은 클래스 이름과 동일하다.

클래스의 멤버 변수들은 자동으로 초기화 되지 않기 때문에 생성자에서 초기화 한다.

소멸자는 객체가 소멸되는 시점에서 자동으로 호출되는 클래스의 멤버 함수이다.

소멸자의 이름은 클래스 이름 앞에 ~를 붙인다.

오직 하나이고, 리턴 타입이 없다.

객체의 동적 생성 및 반환

클래스이름 \*포인터변수 = new 클래스이름;

클래스이름 \*포인터변수 = new 클래스이름(생성자매개변수리스트);

new는 클래스 크기의 메모리를 할당 받아 객체를 생성하며, 이때 생성자를 호출한다. 앞의 첫 번째 형식의 경우 기본 생성자가 호출되며, 두번째의 경우 매개변수를 가진 생성자가 호출된다.

delete를 이용해 객체 반환을 할 수 있다.

delete 포인터변수;

this는 객체 자신에 대한 포인터로서 클래스의 멤버 함수내에서만 사용된다.

컴파일러에 의해 보이지 않게 전달되는 객체에 대한 주소이다.

다른 객체 속의 this와 서로 다른 포인터이다.

상속은 자식 클래스의 객체가 생성될 때, 자신의 멤버뿐 아니라 부모 클래스의 멤버를 포함할 것을 지시한다. C++에서 부모클래스를 기본 클래스, 자식클래스를 파생클래스 라고도 부른다.

파생 클래스의 객체가 생성될 때 파생 클래스의 생성자와 기본 클래스의 생성자가 모두 실행된다. 생성자는 객체를 초기화 할 목적으로 사용되므로, 각각 필요한 초기화를 수행한다.

파생 클래스의 생성자과 기본 클래스의 생성자 중 기본 클래스의 생성자가 먼저 실행된다.

UML(Unified Modeling Language): 애플리케이션을 구성하는 클래스들 간의 관계를 그리기 위하여 사용.

+----------------+ +는 public을 의미하며, -는 private을 의미

| Animal |

+----------------+

| - name: String |

| - age: int |

Animal 클래스와 Dog 클래스 사이에 일반화 관계가 존재한다.

 Dog 클래스는 Animal 클래스를 상속받아 name과 age 속성을 가지며 makeSound() 메서드를 사용할 수 있다. 또한 Dog 클래스는 자체적으로 breed 속성과 bark() 메서드를 가지고 있다.

+----------------+

| + makeSound() |

+----------------+

^

|

+----------------+

| Dog |

+----------------+

| - breed: String|

+----------------+

| + bark() |

+----------------+