객체 지향 프로그래밍(OOP, Object-Oriented Programming)

객체 지향 프로그래밍에서는 모든 데이터를 객체(object)로 취급하며, 이러한 객체가 바로 프로그래밍의 중심이 됩니다. 이 때 객체를 만들어 내기 위한 설계도 같은 개념을 클래스라고 합니다.

클래스

클래스란 객체를 정의하는 틀 또는 설계도와 같은 의미로 사용됩니다. 이러한 클래스를 가지고 여러 객체를 생성하여 사용하게 됩니다.

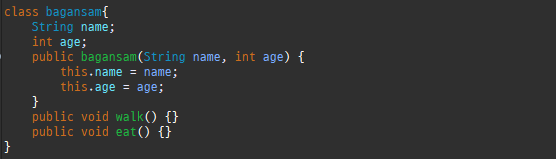
클래스는 객체의 상태를 나타내는 필드(클래스에 포함된 변수)와 객체의 행동을 나타내는 메소드로 구성됩니다.

인스턴스 = 메모리에 할당된 객체

자바에서 클래스를 사용하기 위해서 해당 클래스 타입의 객체를 선언해야하는데, 이 과정을 인스턴스화라고 하며, 이렇게 선언된 해당 클래스 타입의 객체를 인스턴스라고 합니다.

필드 – name : 바간샘, height : 1xxcm 등

메서드 – eat(), walk(), study() 등



상속

Class B extends A -> 물려받는 B가 자식 클래스, 물려주는 A가 부모클래스.

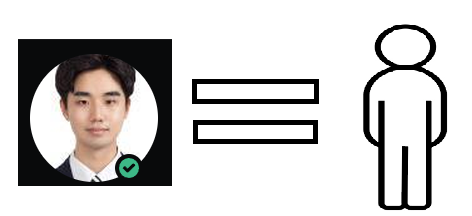
자식 클래스는 부모 클래스의 필드와 메서드를 물려받게 된다. 단, private 접근제어자의 경우 상속이 불가능하고, 패키지가 다른 경우 default인 경우에도 상속이 불가능하다.

다중 상속이 불가능하다. 여러 부모가 같은 이름의 함수를 갖는 경우 등 충돌이 일어날 수 있기 때문.

다중 상속의 경우 implements interface를 통해서 비슷하게나마 구현이 가능하다.

상속의 장점은 중복된 코드를 줄일 수 있고, 유지 보수가 편리하며, 통일성이 있고 다형성을 구현할 수 있다.

Is-A 개념

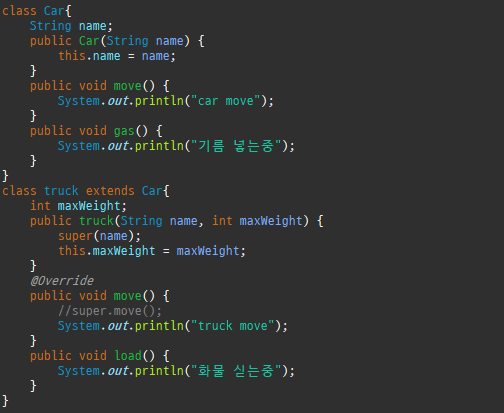


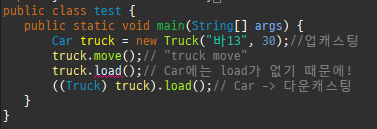
바간샘 is a human. -> 바간샘은 사람이다. -> 바간샘은 사람을 상속 받을 수 있다. 바간샘 extends human

Truck is a car -> Truck extends car

상속 오버라이딩

부모 클래스의 메소드의 내용을 변경하는 것. 부모와 자식의 차이가 있기 때문에 자식에게 맞는 방식으로 변경한다.





업캐스팅 -> 묵시적 형 변환

다운캐스팅 -> 명시적 형 변환

형 변환을 하는 이유 -> 상속을 받은 서브 클래스가 몇 개든 한 번에 묶어서 관리하기 위해서.

[[JAVA] 업캐스팅(up casting), 다운캐스팅(down casting) Casting을 하는 이유는? 다형성? — dev.j (tistory.com)](https://jhost.tistory.com/27)

패키지

비슷한 성격의 자바 클래스들을 모아 놓은 자바의 디렉토리

Ex) java.util.~, ssafy.model.dto

패키지 밑에 서브 패키지도 존재할 수 있다.

장점

비슷한 성격의 클래스들끼리 묶을 수 있어 클래스의 분류가 용이하다. 그리고 패지키명이 다르면 클래스명이 동일해도 충돌없이 사용할 수 있다.

명명 규칙

숫자로 시작해서는 안되고, \_,$를 뺀 특수문자는 사용하지 않는다.

Java로 시작하는 패키지는 표준 패키지이므로 사용하면 안된다.

패키지가 중복되지 않도록 회사의 도메인 이름으로 만드는데, 도메인 이름 역순으로 만든다. Ex)com.ssafy.~