**추상화**

* 복잡한 자료, 모듈, 시스템 등으로부터 핵심적인 개념 또는 기능을 간추려 내는 것
  + 1. 제공해야하는 기능이 구현되지 않은 정의된 상태.
  + 2. 기능이 구현되어있지 않기 때문에 인스턴스화 할 수 없음
  + 3. 추상 클래스나 인터페이스를 구현하는 경우 추상 메소드를 반드시 구현해야 함
  + 4. 추상 메소드를 구현할 때 구조를 변경할 수 없음
  + 5. 따라서 추상화란, 기능에 대한 표준이며 규약

**추상화의 이점**

1. 인터페이스나 추상클래스만으로 코드 작성이 가능

2. 하나의 인터페이스나 추상클래스로 여러 개의 구현 클래스를 만들 수 있음(다형성)

3. 여러 개의 구현 클래스를 이용해 유연한 상황대처가 가능

**추상 클래스와 인터페이스의 차이**

추상클래스와 인터페이스는 추상 메소드를 만든다는 점에서 동일.

일반적으로 추상 메소드의 비율로 추상화 정도를 나타냄

**추상클래스 – class**

추상 메소드와 일반 메소드가 혼재할 가능성이 매우 높음

비 완전 추상화

**인터페이스 – interface**

추상 메소드만 존재함.

완전 추상화

**인터페이스가 추상클래스보다 추상화의 정도가 높음.**

**추상클래스의 활용**

기존 코드를 확장하려 할 때

기존의 코드는 수정하지 않고 추상 클래스를 활용해 코드 전 후에   
해야 할 일을 작성.

**인터페이스의 활용**

제공되는 기능에 대해서는 알고 있지만 환경에 따라 기능의 구현이 변경되어야 할 때

환경: OS, Database, 사용자 입력 값 등 여러 요인이 존재

**다른 걸 중괄호 없는 함수로 제작**

**같은 건 중괄호 있는 함수로 제작**

인터페이스: 메소드 완성이 불가능 100% 완전 추상화일 떄 사용

즉 공통적 기능을 넣을 수 있느냐 없느냐의 차이

추상화 정도가 추상화는 낮고 인터페이스는 높음