Java 에서의 캡슐화(Encapsulation)란

캡슐화는 관련 있는 변수와 함수를 하나의 클래스로 묶고 외부에서 쉽게 접근하지 못하도록

은닉 하는게 핵심이다. 객체에 직접적인 접근을 막고 외부에서 내부의 정보에 직접 접근하거나

변경할 수 없고, 객체가 제공하는 필드와 메소드를 통해서만 접근이 가능하다.

캡슐화의 가장 큰 장점은 정보은닉에 있다.

외부에서 객체접근 하는데 있어서 정보를 숨기고 객체의 연산을 통해서만 접근이 가능하게 하는 것 이다. 정보은닉의 장점은 외부에서 특정 객체의 데이터 및 함수를 직접 접근을 막음으로써 변경을 못하게 하고 유지보수나 확장 시 오류의 범위를 최소화 할 수 있고, 객체내 정보손상, 오용을 방지하고, 조작법이 바뀌어도 사용방법 자체는 바뀌지 않고, 데이터가 변경되어도 다른 객체에 영향을 주지 않기 때문에 독립성이 좋고, 처리된 결과 사용으로 이식성이 좋고, 객체를 모듈화 할 수 있어서 새로운 시스템의 구성에 하나의 모듈처럼 사용이 가능하다.

캡슐화를 하기 위해서는 접근 제어자를 통해 설계가 잘 이루어져야 한다.

자신 내부의 모듈은 감추고, 다른 모듈의 내부 작업도 직접적으로 개입하지 못하도록 설계해야 한다.

클래스 접근 제어자

Default : 동일 패키지의 클래스에서만 인스턴스(객체)를 생성 할 수 있다.

Public : 다른 패키지에서 인스턴스(객체)를 생성 할 수 있다.

메소드 접근 제어자

Private : 동일한 클래스 안에서만 접근이 가능하고, this를 사용하는 것들은 외부에서 접근이 불가능하고, 상속도 안된다.

Default : 접근제어자가 없는 형태로 동일한 패키지 안에서만 접근이 가능하다.

Protected : 동일한 패키지 안에서 사용이 가능하고, 다른 패키지라도 상속반은 클래스에서는 접근이 가능하다.

Pubilc : 모든 객체에서 접근이 가능하다.

접근을 위해 Setter와 Getter 라는 장치를 만들어 내부에서 접근 할 수 있다.

입력할때는 Set, 가져올때는 Get을 사용하여 작동할 수 있다.