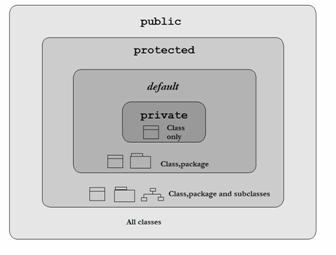
**객체 지향 프로그래밍(Object Oriented Programming, OOP)**은 각각의 기능들을 객체화(클래스화)시켜 메인함수에서는 객체화시킨 기능들을 호출해서 사용하는 프로그래밍 기법으로 **캡슐화, 상속, 다형성**의 특징을 가지고 있다.

**캡슐화(Encapsulation)**

**캡슐화는 일반적으로 변수와 함수를 클래스로 묶는 작업을 말한다.** 캡슐화를 할 때에는 무작위로 클래스를 지정하는 것이 아니라**연관된 목적을 가지고 있는 변수와 함수별로 작업**을 해야 한다.

 또한, **캡슐화는 외부의 잘못된 접근으로 값이 변하는 것을 막기 위해 클래스 내의 변수나 함수를 감추거나 드러내는 은닉성을 지니고 있다.** 이러한 은닉성은 자바에서 접근제어자를 통해 실현시킬 수 있다.

**접근제어자(Access Modifier)**



**· public** : **어떠한 클래스**에서라도 접근 가능

**· protected** : **동일 패키지 내의 클래스**나 동일 패키지가 아니더라도 **해당 클래스를 상속받은 클래스**에서 접근 가능

**· default** : **동일 패키지 내의 클래스**에서 접근 가능, 접근제어자를 별도로 설정하지 않으면 default

**· private** : **해당 클래스에서만** 접근 가능

**상속(Inheritance)**

 자바에서의 상속은 일반적으로 알고 있는 사전적 의미와 크게 다르지 않다. **자식 클래스에서 부모 클래스로부터 부모의 모든 자원을 물려 받는 것을 의미**하며 자바에서는 지정예약어 extends에 의해서 정해진다. 하나의 부모 클래스가 여러 자식 클래스를 가지는 것은 가능하지만 하나의 자식 클래스가 여러 부모 클래스를 상속받을 수는 없다.

 이러한 상속을 통해서 개발자는 **비효율적인 코드 중복을 피할 수 있다.** 각각의 자식 클래스에 부모 클래스의 자원을 일일히 적어주어야 하는 수고를 덜어주기 때문이다. 또한, 부모 클래스를 한번만 수정하여 자식 클래스의 전체가 수정이 되기 때문에 **유지 보수의 편리성도 얻을 수 있다.**

**다형성(Polymorphism)**

 다형성은 말 그대로 여러 형태를 가질 수 있는 능력으로 하나의 객체가 여러 가지 타입을 가질 수 있는것을 의미한다. **즉, 부모 클래스 타입의 참조변수로 자손 클래스의 인스턴스를 참조할 수 있도록 해준다.**

**각각의 자손 클래스의 특징을 가진 인스턴스들이 동일한 데이터 타입을 가지게 되면 메소드를 실행시키기 간편해진다.** 각각의 인스턴스를 이용하여 동일한 기능의 메소드를 실행시켜야 한다고 가정해보자. 만약 전부 다른 데이터 타입을 가진다면 각각의 데이터 타입에 맞는 매개변수를 가진 메소드를 하나하나 정의해주어야한다. 그러나, 같은 데이터 타입을 가지게 된다면 단 하나의 메소드만 정의해주어도 된다.

**자식 클래스에서는 부모로부터 상속받은 메서드를 그대로 사용해도 되지만 오버라이딩(Overriding)을 통해 수정하여 사용할 수 있다.** 이는 공통 부모를 가진 자식 클래스들을 세분화시킬 수 있다. 이렇게 각각 세분화된 자식 클래스의 인스턴스를 부모 클래스의 데이터 타입으로 통일하여 인스턴스를 생성할 수 있다