section 1.1

싱글톤 패턴 : 하나의 클래스에 오직 하나의 인스턴스만 가지는 패턴.

팩토리 패턴 : 하위 클래스에서 객체 생성에 관한 구체적인 내용을 결정하는 패턴

전략 패턴 : 직접 수정하지 않고 캡슐화한 알고리즘을 컨텍스트 안에서 바꿔주면서 상호 교체가 가능하게 만드는 패턴

옵저버 패턴 : 주체가 어떤 객체의 상태 변화를 관찰하다가 상태 변화가 있을 때 옵저버들에게 변화를 알려주는 패턴

프록시 패턴 : 대상 객체에 접근하기 전. 一름을 가로채 접근을 필터링, 수정 역할을 하는 계층을 가지는 패턴

이터레이터 패턴 : 이터레이터를 사용하여 컬렉션의 요소들에 접근하는 패턴

노출 패턴 : 즉시 실행 함수를 통해 접근 제어자를 만드는 패턴

MVC 페턴 : model view, controller로 이루어진 패턴

1. model : application의 데이터베이스, 상수, 변수
2. view : 사용자의 인터페이스
3. controller : 모델과 하나 이상의 뷰를 잇는 다리 역할

MVP 패턴 : model, view, presenter

mvc패턴 보다 더 강한 결합을 지닌 디자인 패턴

MVVM 패턴 : View Model로 바뀐 패턴

뷰 모델은 뷰를 더 추상화한 계층 데이터 바인딩을 가지는 것이 특징 → 단위 테스팅하기가 더 쉽다.