**4장 팩토리 패턴**

**-요구 사항-**

3장 요구사항은 유지한다.

(팩토리메소드) 새로운 유닛 – 적팀 유닛 (유닛 종류별), 청팀 유닛 (유닛 종류별)

1. 이제 모든 유닛은 팀유닛, 일반유닛으로 나뉜다.

2. 팀 유닛에는 청팀, 적팀이 있으며 팩토리 패턴으로 생성한다. (기존 유닛은 그대로)

4. 팩토리 메소드 패턴을 사용하여 구현한다.  
4-1. 청팀, 적팀팩토리를 통해 적팀유닛, 청팀유닛 생성

4-2. 팩토리는 청팀유닛, 적팀유닛 팩토리가 존재하고, 적팀, 청팀 유닛 클래스는 다른차이는 없고, 단지 toString함수를 오버라이딩하여 이름을 달리한다.

---------------------------------------------------------------------------------------------------  
(추상팩토리) 새로운 유닛 - 팀유닛  
(추상팩토리) 새로운 클래스 – 팀 정보

(추상팩토리) 팀 정보 – 팀 이름, 팀 구호

(추상팩토리) 팀 유닛 상태 – 팀 정보

5. 추상팩토리 패턴을 사용하여 구현한다.

5-1. 팩토리는 청팀정보, 적팀정보 팩토리가 존재하고, 유닛생성시 추상팩토리가 팀유닛의 팀정보 초기화를 대행

5-2. 추상팩토리 패턴에서는 적팀, 청팀유닛클래스가 아닌 ‘팀유닛’의 팀정보객체를 통해 팀구별

추상팩토리 패턴과 팩토리 메소드 패턴 구현은 서로 완전 독립. 두 패턴 사이 공통의 클래스가 없을것이며 아예 다른 방식으로 패키지를 분리하여 두번 구현한다.

**-실행 시나리오 -**

1. 중립 유닛 종류별로 생성한다. (기본 정보출력)

2. 추상 팩토리패턴을 이용하여 청팀유닛을 종류별로 생성한다. (기본 정보출력)

3. 추상 팩토리패턴을 이용하여 적팀유닛을 종류별로 생성한다. (기본 정보출력)

4. 팩토리 메소드 패턴을 이용하여 청팀유닛을 종류별로 생성한다. (기본 정보출력)

5. 팩토리 메소드 패턴을 이용하여 적팀유닛을 종류별로 생성한다. (기본 정보출력)