10.패턴

1. 패턴이란 ?

패턴은 객체지향 언어를 배우는데 알아야 하는 중요한 방법 중 하나임 개발을 할 때 일종의 패턴을 가지고 코딩을 하는 것을 하나하나 정립화 시켜놓은 것이 패턴. 이 방식을 어떠한 틀로 정해 놓은 것이 디자인 패턴

즉 기존의 개발자분들이 객체지향 언어의 장점들을 모아 가장 효율적으로 개발을 할 수 있게 만들어 놓은 프레임

2. 싱글톤 패턴 : 어떤 클래스의 객체는 오직 하나의 객체만을 만들어 여러가지 상황에 동일한 객체에 접근하기 위해 만들어진 패턴

즉 싱글톤 패턴은 객체를 한 개만 생성해야 하기 때문에 new를 쓸 수가 없음

```
public class SingletonClass {
        private static SingletonClass SINGLETON_INSTANCE;
        private int i;


    private SingletonClass(){i = 10;}    //생성자함수가 private이므로 외부에서는 new를 이용해서
          // 생성못하고, 이 클래스 내부에서만 생성자함수 호출할 수 있다.
    public static SingletonClass getSingletonClass(){
          // 객체가 생성되기 전에 데이터영역의 클래스 상태에서 바로 접근가능한 메소드
          if(SINGLETON_INSTANCE==null)
                SINGLETON_INSTANCE = new SingletonClass();//객체생성을 이곳에서만 한다
          return SINGLETON_INSTANCE;
    }
}
public class FirstClass {
      public FirstClass(){
            SingletonClass singletonObject = SingletonClass.getSingletonClass();
            System.out.println("FirstClass 객체 ");
            System.out.println(singletonObject.getI());
            singletonObject.setI(999);
            System.out.println("i = "+singletonObject.getI());
      }
```

객체생성을 할 수가 없기 때문에 위와 같이 new가 아닌 get을 통해 받아냄

3. 스트레티지 패턴

기능 하나를 정의 하고 각각을 캡슈로하 하여 교환해서 사용할 수 있도록 만든 것

시나리오
모든 로봇은 기본적으로 걷고, 달릴 수 있어야 합니다.
로봇 모양은 팔, 다리, 머리, 몸통으로 이루어져 있습니다.
Super 로봇 : 날 수 O. 미사일을 쓸 수 O. 레이저검.
Standard로봇: 날 수 X. 미사일을 쓸 수 O. 목검
Low 로봇 : 날 수 X. 미사일을 쓸 수 X. 검 없음.

이런 클라이언트의 요구가 있다고 생각했을 때..

1. 로보트라는 추상 클래스에 flying 유무, 미사일 유무, 무기 유무 관련 함수 선언

2. 로보트가 가지고 있는 공통 특성 만들어 놓기(걸을 수 있고, 달릴 수 있음)

3. 각각의 클래스에서 구현할 수 있도록 각각의 메소드(부품) 만들어 놓기

4. 상속받은 각각의 로보트 클래스에 특징에 맞게 함수 override 하여 구현해놓기

5. 메인에서 객체 생성 후 불러오기