**#boolean의 경우 getter 메서드의 이름이 is로 시작하는 것이 관례**

boolean stop; isStop/setStop

**#싱글톤 : 유일 객체**

해당 객체를 private static 키워드를 붙여서 선언하고, getInstance와 같은 메서드로만 참조가 가능하도록 함.

**#상속**

자식 인스턴스 생성자 호출 시 부모 생성자가 자동으로 호출 됨.

자식 생성자 안에 super() 생략되어있음.

부모 클래스에 기본생성자 없으면 컴파일 에러.

부모 클래스에 매개변수필요한 생성자가 있다면 자식 생성자 내부에서 super(매개1,매개2); 로 호출해줘야함.

오버라이딩 메서드 내에서 해당 부모 메서드 호출하기

super.method();

**#final class**

최종적인 클래스이므로 상속이 불가능하다.

**#final method**

최종적인 메서드임으로 오버라이딩이 불가능하다.

**#상속과 다형성**

부모객체에 자식인스턴스를 넣어두면 부모객체에 선언된 변수와 메서드만 사용이 가능하다.

그러나 자식 인스턴스에 오버라이딩된 메서드가 있다면 자식메서드가 호출된다.(다형성)

**#instanceof 연산자**

Ex. child instanceOf parent => X child 인스턴스가 parent인가?

**#봉인된 클래스**

public sealed class A permits B,C(){} : A를 B,C만 상속이 가능하다

public final B extends A(){} : B를 다른 클래스들이 상속이 불가능하다.

public non-sealed C extends A(){} : C를 다른 클래스들이 상속이 가능하다.

파이널 -> 자식 낳지마

씰 -> 뭐뭐 허용해줄게

넌씰-> 봉인해제

**#인터페이스(상수, 추상 메서드, default 메서드(구현 메서드), 정적 메서드, private 메서드) <-> 추상클래스는 인스턴스 멤버 변수를 가질 수 있다.**

인터페이스형 객체는 구현 인스턴스를 참조할 수 있다.

이경우 추상메서드를 호출하면 상속과 마찬가지로 구현 인스턴스의 구현 메서드가 호출이 된다.

현업에서는 다형성을 상속보다 주로 인터페이스를 많이 이용한다고 한다.

멤버변수를 선언하면 public final static 키워드가 자동 생략되어 상수화됨. inter1.PI 로 바로 접근.

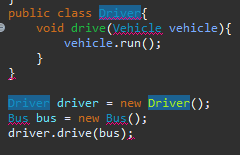
**default메서드의 경우 상수와 추상메서드를 호출하여 사용이 가능하고 구현 객체를 통해서 호출할 수 있다.**

**static정적 메서드는 구현 객체가 없어도 호출이 가능하다. private 메서드**

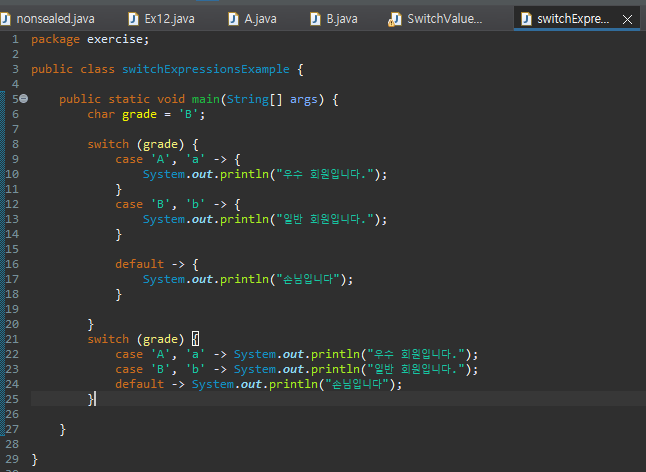
**default 메서드를 활용해서 공통 구체 메서드를 인터페이스를 구현하여 사용할 수 있다.**

**#매개변수의 다형성**

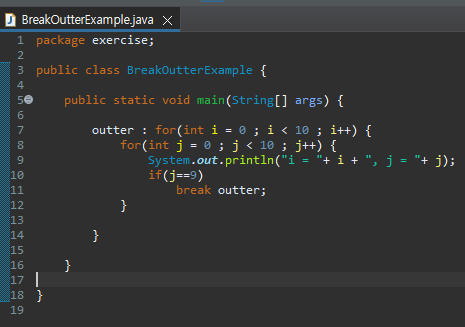
**매개변수를 상속에서 부모 타입이나 인터페이스로 선언하면 자식 인스턴스/구현 객체 대입할 수 있다.**

** 밑줄 신경 X**

**#switch문 java 12 문{  
법 개선. -> 사용하면 break를 쓰지 않아도 되게 됨.**



**#반복문에서의 Break: 일반적으로 break는 가장 가까운 반복문을 종료한다. 하지만 특정 반복문을 종료하는 방법도 있다. 특정 반복문에 라벨을 붙여서 종료시키는 방법이다.**



**변수이름과 메서드이름이 길어지더라도 직관성있도록 쓰도록 노력하자. 개발하고 수정하던가.**

**#중첩 클래스**

**중첩 클래스 : 여러 클래스와 관계 맺지 않고 특정 클래스만 관계를 맺을 때 사용한다.**

1. 멤버 클래스
   1. 인스턴스 멤버 클래스 : A 객체를 생성해야지만 B 객체 생성 가능

class A {

class B{

}

} class B의 경우 private 접근 제어자 일반적.

* 1. 정적 멤버 클래스 : A 객체를 생성하지 않아도 B 객체 생성 가능

class A{

static class B{

}

} class B의 경우 default/public 이 일반적.

**A객체를 생성하지 않아도 A.B b = new A.B();**

1. 로컬 클래스 : 메서드 내에서 클래스 선언 -> method가 실행될 때만 B객체 생성할 수 있다.

class A{

void main(){

class B{ ,,, }

}

}