Getter와 Setter

▣ Getter와 Setter

- 객체의 필드(데이터)를 외부에서 마음대로 읽고, 변경할 경우 객체의 무결성(결점이 없는 성질)이 깨질 수 있다. 이러한 문제점 때문에 객체 지향 프로그래밍에서는 직접적인 외부에서의 접근을 막고 대신 메소드를 통해 필드에 접근하는 것을 선호한다. 그 이유는 메소드는 데이터를 검증해서 유효한 값만 필드에 저장할 수 있기 때문이다. 이러한 역할을 하는 메소드가 Setter이다.

- 필드값이 객체 외부에서 사용하기에 부적절한 경우, 메소드로 적절한 값으로 변환해서 리턴할 수 있다. 이러한 역할을 하는 메소드가 Getter이다.

```
private double speed;

public void setSpeed(double speed) {
  if(speed < 0) {
    this.speed = 0;
    return;
  } else {
    this.speed = speed;
  }
}
```

매개값이 음수일 경우 speed 필드에 0으로 저장하고, 메소드 실행 종료

```
private double speed;     //speed의 단위는 마일

public double getSpeed() {
  double km = speed*1.6;
  return km;
}
```

필드값인 마일을 km 단위로 환산 후 외부로 리턴


private 타입 fieldName;
필드 접근 제한자: private
//Getter
public 타입 getFieldName() {
return fieldName;
}
접근 제한자: public
리턴 타입: 필드타입
메소드 이름: get + 필드이름(첫 글자 대문자)
리턴값: 필드값
//Setter
public void setFieldName(타입 fieldName) {
this.fieldName = fieldName;
}
접근 제한자: public
리턴 타입: void
메소드 이름: set + 필드이름(첫 글자 대문자)
매개변수 타입: 필드타입
private boolean stop;
필드 접근 제한자: private
//Getter
public boolean isStop() {
return stop;
}
접근 제한자: public
리턴 타입: 필드타입
메소드 이름: is + 필드이름(첫 글자 대문자)
리턴값: 필드값