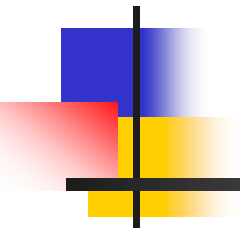


# Television

경복대학교
소프트웨어융합과
배희호 교수

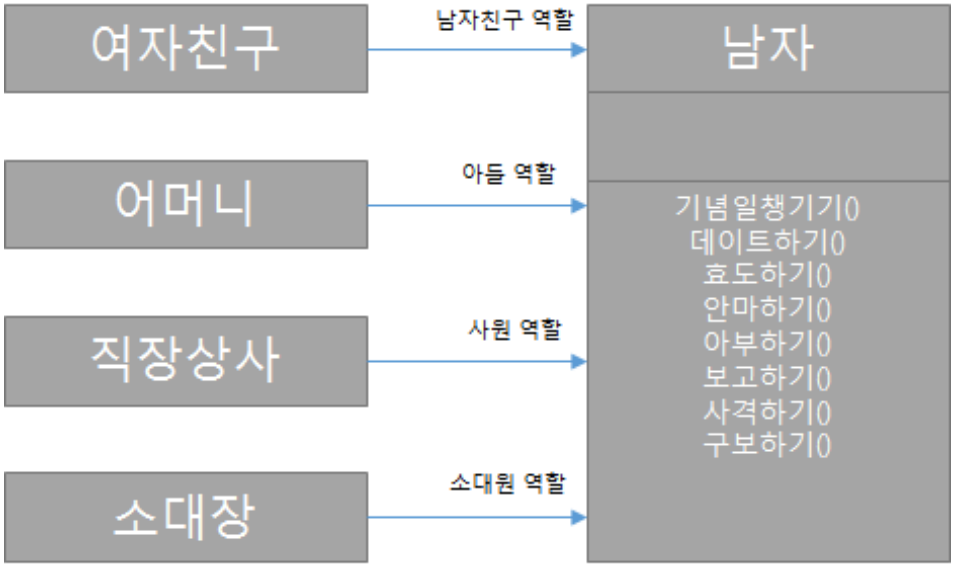
# Object Oriented의 설계 5원칙

- Object Oriented 설계의 5대 원칙(SOLID 원칙)
  - 유지 보수성과 확장성을 높이기 위해 Software 설계에서 지켜야 할 원칙들 임

  - SRP : Single Responsibility Principle (단일 책임 원칙)
  - OCP : Open-Closed Principle (개방 폐쇄 원칙)
  - LSP : Liskov Substitution Principle(리스코프 치환 원칙)
  - ISP : Interface Segregation Principle
    (인터페이스 분리 원칙)
  - DIP : Dependency Inversion Principle(의존 역전 원칙)

# Object Oriented의 설계 5원칙

- SRP(Single Responsibility Principle) : 단일 책임 원칙
  - 하나의 Class는 하나의 책임만 가져야 함
  - 변경이 필요할 때, 오직 하나의 이유만으로 수정되어야 함



# Object Oriented의 설계 5원칙

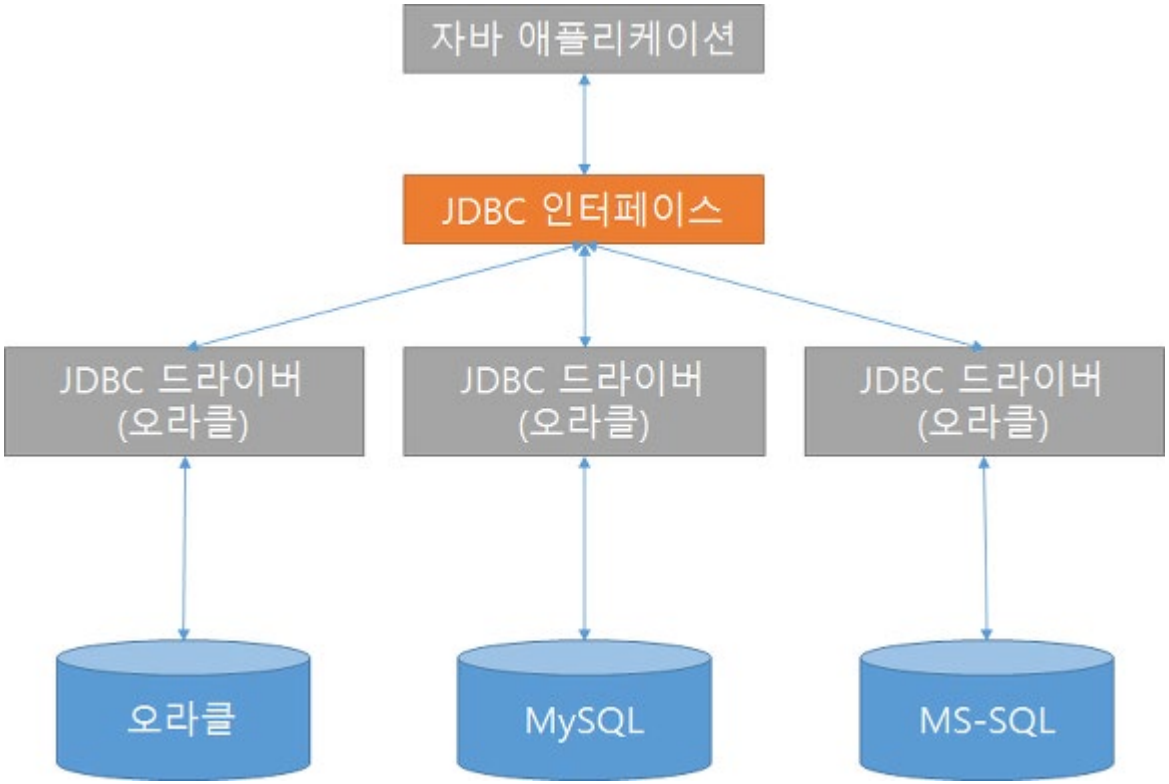
- SRP(Single Responsibility Principle) : 단일 책임 원칙

# Object Oriented의 설계 5원칙

- OCP(Open Closed Principle) : 개방 폐쇄 원칙
  - “Software Entity(Class, Module, Function 등)는 확장에 대해서는 열려 있어야 하지만 변경에 대해서는 닫혀 있어야 한다.“
  - 자신의 확장에는 열려있고, 주변의 변화에 대해서는 닫혀 있어야 한다.“
  - 새로운 기능을 추가할 때 기존 Code를 수정하지 않고 확장할 수 있도록 설계해야 함
  - 예) Shape Interface를 만들어 Circle과 Rectangle이 이를 구현하도록 하면, 새로운 도형을 추가할 때 기존 Code를 수정할 필요가 없음

# Object Oriented의 설계 5원칙

- OCP(Open Closed Principle) : 개방 폐쇄 원칙

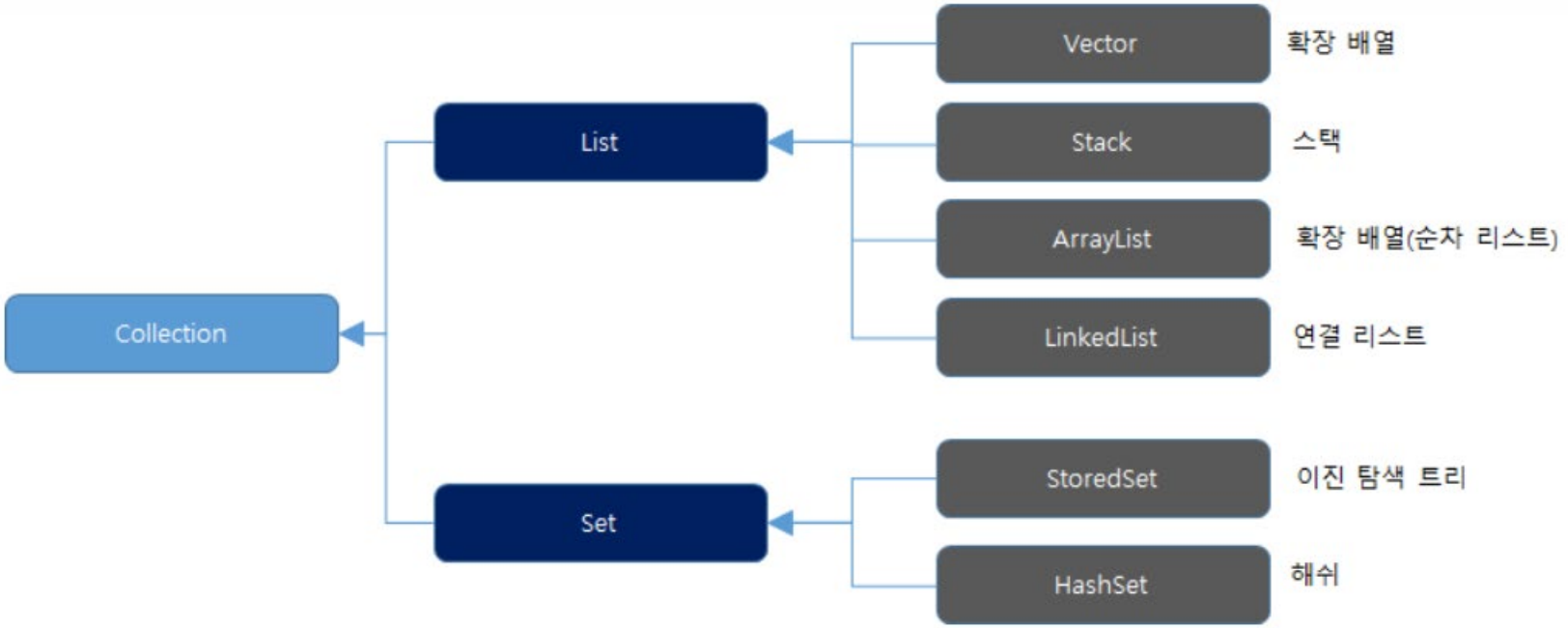


# Object Oriented의 설계 5원칙

- LSP(Liskov Substitution Principle) : 리스코프 치환 원칙
  - 1988년 Barbara Liskov(바바라 리스코프)가 올바른 상속 관계의 특징을 정의하기 위해 발표한 것으로, Sub Type은 언제나 기반 Type으로 교체할 수 있어야 한다는 것을 뜻함
  - 교체할 수 있다는 말은, 자식 Class는 최소한 자신의 부모 Class에서 가능한 행위는 수행이 보장되어야 한다는 의미
  - 즉, 부모 Class의 Instance를 사용하는 위치에 자식 Class의 Instance를 대신 사용했을 때 Code가 원래 의도대로 작동해야 한다는 의미

# Object Oriented의 설계 5원칙

- LSP(Liskov Substitution Principle) : 리스코프 치환 원칙



# Object Oriented의 설계 5원칙

- ISP(Interface Segregation Principle) : 인터페이스 분리 원칙
  - Interface 분리 원칙이란 객체는 자신이 사용하는 Method에만 의존해야 한다는 법칙
  - Object가 사용하지 않는 Method를 의존해서는 안 된다는 뜻이기도 함
  - Interface는 지나치게 광범위하거나 지나치게 많은 기능을 구현해서는 안 되고, 그 Interface를 사용하는 Object를 기준으로 잘게 분리되어야 한다는 의미
  - 예) Print Interface와 Scan Interface스를 분리하면, Print 기능만 필요한 Class가 불필요한 scan() Method를 구현할 필요가 없음

# Object Oriented의 설계 5원칙

- DIP(Dependency Inversion Principle) : 의존 역전 원칙
  - 고수준 Module(비즈니스 로직)이 저수준 Module(구현 세부사항)에 의존하는 것이 아니라, 추상화(인터페이스)에 의존해야 함
  - 예) Database Class 대신 IDatabase Interface를 사용하면, 나중에 MySQL, PostgreSQL 등의 구현체를 자유롭게 교체할 수 있음

# Television 문제 0

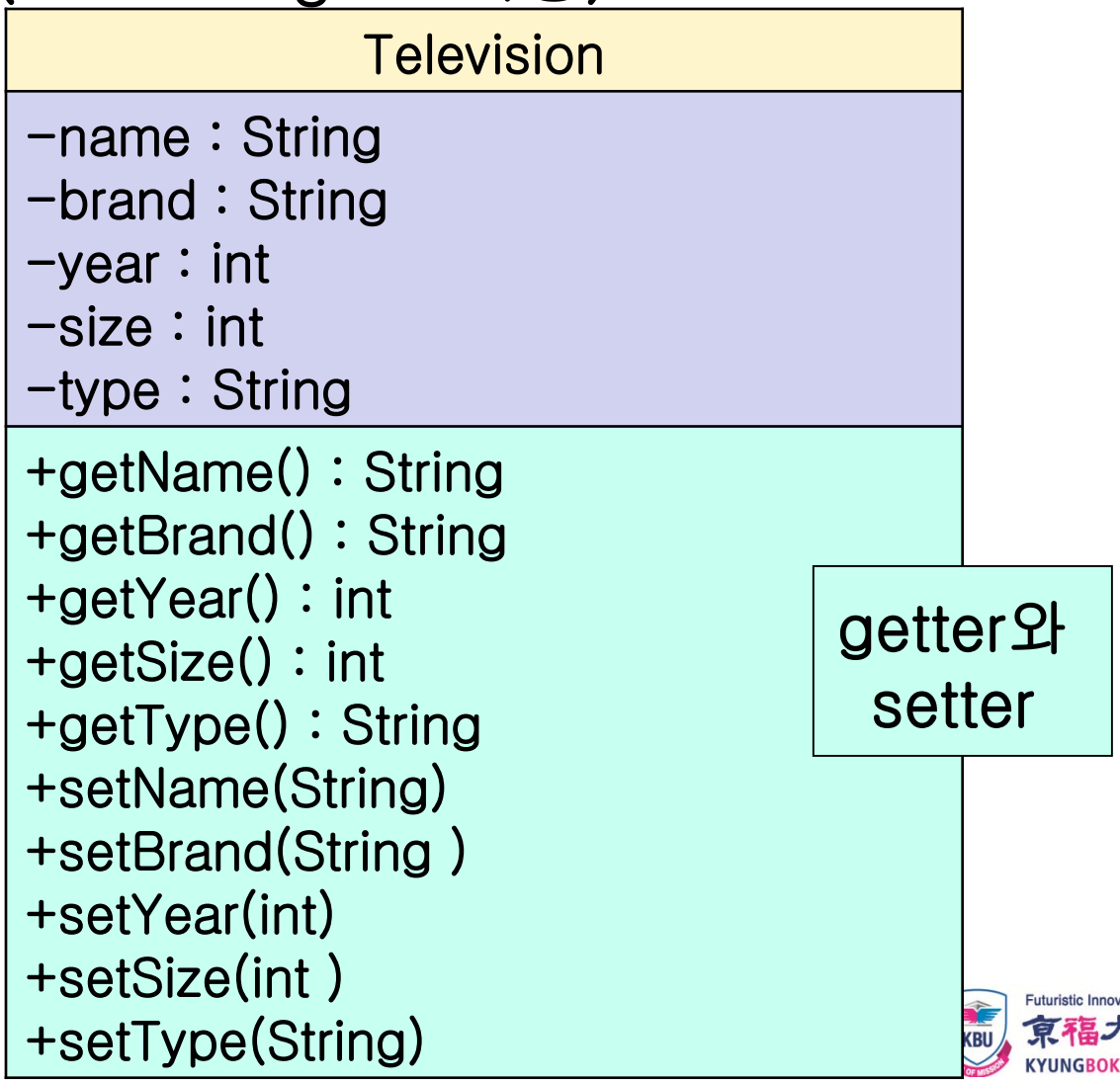
- 다음과 같이 출력하는 Program을 만들어보자

*우리집 TV는 Samsung에서 만든 2017년형 55 인치 LED TV 입니다*

- 관심 사항을 파악하자 (Data)
  - 명사 (속성)
- Class를 만들어보자
  - Member 변수

# Television 문제 0

- Class 설계 (Class Diagram 작성)



# Television 문제 0

- Television.JAVA

```
public class Television {
    private String name;
    private String brand;
    private int year;
    private int size;
    private String type;

    public Television(String name, String brand, int year, int size, String type) {
        this.name = name;
        this.brand = brand;
        this.year = year;
        this.size = size;
        this.type = type;
    }

    public String getName() {
        return name;
    }
```

# Television 문제 0

- Television.JAVA

```
    public String getBrand() {
        return brand;
    }
    public int getYear() {
        return year;
    }

    public int getSize() {
        return size;
    }

    public String getType() {
        return type;
    }
}
```

# Television 문제 0

- Main.JAVA

```
public static void main(String[] args) {
    Television myTV = new Television("우리집 TV", "Samsung", 2017, 55, "LED");

    System.out.printf("%s는 %s에서 만든 %d년형 %d인치 %s TV 입니다₩n",
            myTV.getName(), myTV.getBrand(), myTV.getYear(), myTV.getSize(),
                                                    myTV.getType());
}
```

# Television 예제 1

- Television의 필드(Field)와 메소드(Method)

TV 객체

상태

- 채널번호
- 볼륨
- 전원상태

동작

- 켜기
- 끄기
- 채널변경하기
- 볼륨변경하기

# Television 예제 1

- Television 클래스

```
public class Television {
    private int channel;
    private int volume;
    private boolean onOff;    // powerButton으로 동작

    public Television() {
    }

    public Television(int channel, int volume) {
        this.channel = channel;
        this.volume = volume;
        this.onOff = false;
    }

    public int getChannel() {
        return channel;
    }
```

# Television 예제 1

- Television 클래스

```
public void setChannel(int channel) {
    if (onOff)
        this.channel = channel;
}

public int getVolume() {
    return volume;
}

public void setVolume(int volume) {
    if (onOff)
        this.volume = volume;
}

public boolean getOnOff() {
    return onOff;
}
```

# Television 예제 1

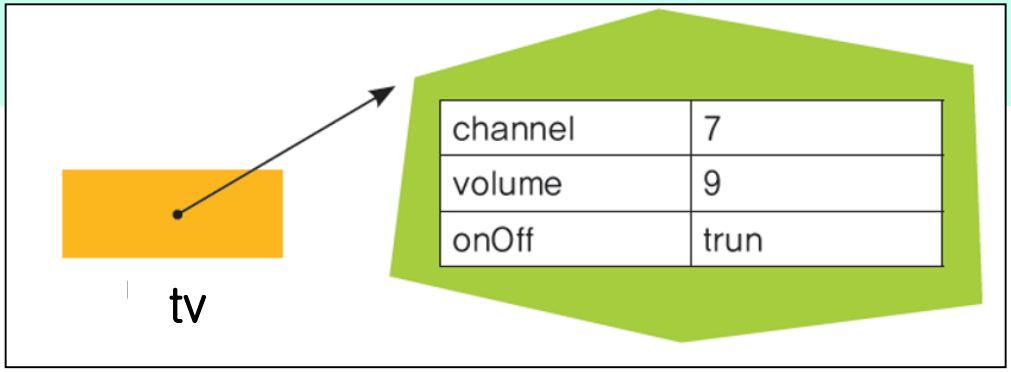
- Television 클래스

```
    public void powerButton() {
        if (onOff)
            onOff = false;
        else
            onOff = true;
    }

    @Override
    public String toString() {
        return "채널 : " + channel +
                ", 볼륨 : " + volume +
                ", onOff = " + onOff;
    }
}
```

# Television 예제 1

## Object 생성

```
public class Main {
    public static void main(String[] args) {
        Television tv = new Television(6, 6);    // 객체 생성
        tv.powerButton();
        tv.setChannel(7);                        // 객체 멤버변수 접근
        tv.setVolume(9);

        if (tv.getOnOff())
            System.out.println("TV 채널은 " +tv.getChannel() +
                                                    "볼륨은 "+tv.getVolume());
        else
            System.out.println("TV가 꺼져있습니다");
    }
}
```



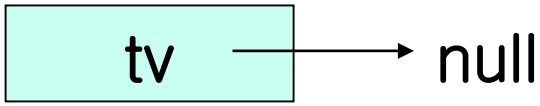
# Television 예제 1

- Object의 선언과 할당
  - Object 변수를 선언하면, Object의 Address를 저장할 수 있는 Memory 공간이 확보됨
  - 실제 Object Data를 저장할 수 있는 공간은 new()에 의해서 실행 시간에 할당됨

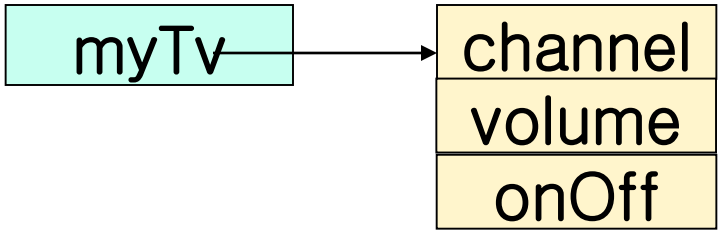
# Television 예제 1

- Television Object tv와 myTv를 선언하는 문장과 이 문장으로 확보되는 Memory 공간을 도식화 하면

Television tv;

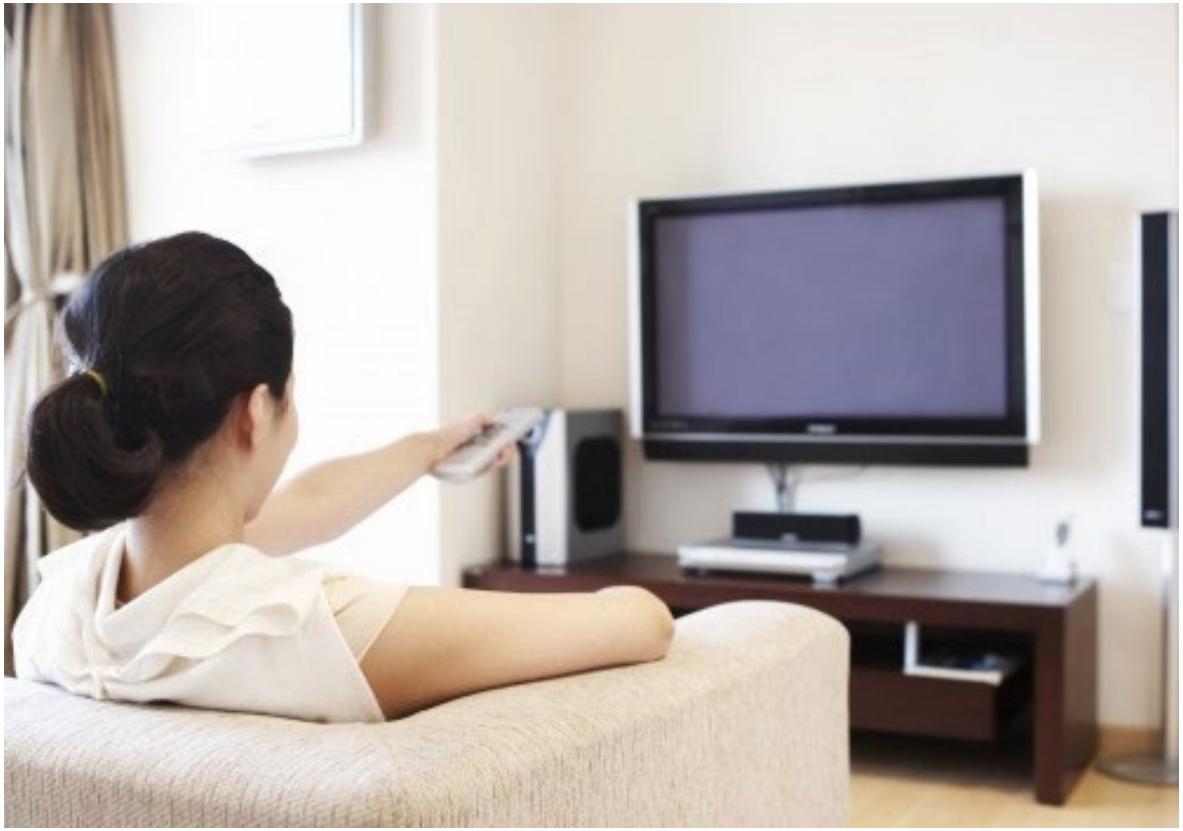



Television myTv = new Television();



# Television 예제 2

- Television을 보자 (Class와의 상호 작용)



# Television 예제 2

- Television Class
  - Channel
  - Volume
  - Power

- Remocon Class
  - Television
  - Battery (0 ~ 100)

  - Power을 켜고/끈다
  - Volume을 조절한다 (0에서 10까지)
  - Channel을 조정한다 (1에서 350개 채널)

# Television 예제 2

- Television Class

```
public class Television {
    private int channel;
    private int volume;
    private boolean onOff;

    public Television(int channel, int volume) {
        this.channel = channel;
        this.volume = volume;
        this.onOff = false;
    }

    public int getChannel() {
        return channel;
    }

    public void setChannel(int channel) {
        this.channel = channel;
    }
```

# Television 예제 2

- Television Class

```
public int getVolume() {
    return volume;
}

public void setVolume(int volume) {
    this.volume = volume;
}

public boolean isOnOff() {
    return onOff;
}

public void setOnOff(boolean onOff) {
    this.onOff = onOff;
}
```

# Television 예제 2

- Television Class

```
    @Override
    public String toString() {
        return  "채널 : " + channel +
                ", 볼륨 : " + volume +
                ", onOff = " + onOff;
    }
}
```

# Television 예제 2

- Remocon Class

```
public class Remocon {
    private Television television;

    public Remocon(Television television) {
        this.television = television;
    }

    public void powerOn() {
        if (television.isOnOff())
            television.setOnOff(false);
        else
            television.setOnOff(true);
    }
```

# Television 예제 2

- Remocon Class

```
public void volumeUp() {
    if (television.isOnOff()) {
        television.setVolume(television.getVolume() + 1);
        if (television.getVolume() > 10)
            television.setVolume(10);
    }
}

public void volumeDown() {
    if (television.isOnOff()) {
        television.setVolume(television.getVolume() - 1);
        if (television.getVolume() < 0)
            television.setVolume(0);
    }
}
```

# Television 예제 2

- Remocon Class

```
    public void channelUp() {
        if (television.isOnOff()) {
            television.setChannel(television.getChannel() + 1);
            if (television.getChannel() > 350)
                television.setChannel(television.getChannel() % 350);
        }
    }

    public void channelDown() {
        if (television.isOnOff()) {
            television.setChannel(television.getChannel() - 1);
            if (television.getChannel() < 1)
                television.setChannel(television.getChannel() + 350);
        }
    }
}
```

# Television 예제 2

- Main Class

```
public static void main(String[] args) {
    Television myTv = new Television(350, 5);
    Remocon remocon = new Remocon(myTv);

    remocon.channelUp();
    remocon.channelDown();

    if (myTv.getOnOff())
        System.out.println(myTv);
    else
        System.out.println("TV가 꺼져있습니다");
}
```

# Television 예제 3

- Remocon의 Battery 상태 관리

# Television 예제 3

- Remocon은 Battery 상태에 따라 동작을 함 (1에서 10)

```
public class Remocon {
    private Television television;
    private int battery;

    public Remocon(Television television) {
        this.television = television;
        battery = 10;
    }

    public int getBattery() {
        return battery;
    }

    public void setBattery(int battery) {
        this.battery = battery;
    }
```

# Television 예제 3

- Remocon은 Battery 상태에 따라 동작을 함 (1에서 10)

```
public void powerOn() {
    if (battery > 0) {
        if (television.getOnOff())
            television.setOnOff(false);
        else
            television.setOnOff(true);
    }
}

public void volumeUp() {
    if (battery > 0) {
        if (television.getOnOff()) {
            television.setVolume(television.getVolume() + 1);
            if (television.getVolume() > 10)
                television.setVolume(10);
        }
    }
}
```

# Television 예제 3

- Remocon은 Battery 상태에 따라 동작을 함 (1에서 10)

```
public void volumeDown() {
    if (battery > 0) {
        if (television.getOnOff()) {
            television.setVolume(television.getVolume() - 1);
            if (television.getVolume() < 0)
                television.setVolume(0);
        }
    }
}

public void channelUp() {
    if (battery > 0) {
        if (television.getOnOff()) {
            television.setChannel(television.getChannel() + 1);
            if (television.getChannel() > 350)
                television.setChannel(television.getChannel() % 350);
        }
    }
}
```

# Television 예제 3

- Remocon은 Battery 상태에 따라 동작을 함 (1에서 10)

```
    public void channelDown() {
        if (battery > 0) {
            if (television.getOnOff()) {
                television.setChannel(television.getChannel() - 1);
                if (television.getChannel() < 1)
                    television.setChannel(television.getChannel() + 350);
            }
        }
    }
}
```