8. 인터페이스(Interface)

인터페이스

- 추상클래스와 비슷하나 상수(final static)와 추상 메소드의 선언문으로만 구성됨
  - 클래스와 같이 public, default 접근 제한자 허용
  - 상수(public static final)와 추상 메소드(public abstract) 만을 가질 수 있음
- 인터페이스를 구현(implements)하는 클래스는 인터페이스에 선언된 모든 추상 메소드를 실제 구현해야 함
  - 하나의 클래스가 하나 이상의 인터페이스를 구현할 수 있음
  - 인터페이스 타입의 참조변수를 구현 클래스 인스턴스로 생성하여 다형성 기능 구현 가능
- 인터페이스 간 상속이 가능하며 다중 상속도 허용

```
interface MyInterface1 {  // public 또는 default 접근 제한자 사용 가능
    public final static int CONST = 100;  // 상수(public static final 생략 가능)
    public abstract void method1( );  // 추상 메소드 선언 (public abstract 생략 가능)
}

interface MyInterface2 {
    public final static int CONST = 500;  // 상수(public static final 생략 가능)
    public abstract void method2( );  // 추상 메소드 선언 (public abstract 생략 가능)
}

class MyClass implements MyInterface1, MyInterface2 {  // 인터페이스 구현
    .....
    public void method1( ){  // 추상 메소드에 대한 구현 (public 생략하면 안 됨)
        …
    }
    public void method2( ){  // 추상 메소드에 대한 구현 (public 생략하면 안 됨)
        …
    }
}
```

실습 - 인터페이스 구현(1)

[Ex1-1] RemoteControl.java

```
package chap08.interfaceex;

public interface RemoteControl {
    //상수
    int MAX_VOLUME = 10;
    int MIN_VOLUME = 0;

    //추상 메소드
    void turnOn();
    void turnOff();
    void setVolume(int volume);
}
```

실습 - 인터페이스 구현(2)

[Ex1-2] Televisionjava

```
package chap08.interfaceex;

public class Television implements RemoteControl {
    private int volume;

    public void turnOn() { // 추상메소드 구현
        System.out.println("TV를 켭니다.");
    }
    public void turnOff() { // 추상메소드 구현
        System.out.println("TV를 끕니다.");
    }
    public void setVolume(int volume) { // 추상메소드 구현
        if(volume>RemoteControl.MAX_VOLUME) {
            this.volume = RemoteControl.MAX_VOLUME;
        } else if(volume<RemoteControl.MIN_VOLUME) {
            this.volume = RemoteControl.MIN_VOLUME;
        } else {
            this.volume = volume;
        }
        System.out.println("현재 TV 볼륨: " + this.volume);
    }
}
```

[Ex1-3] Audio.java

```
package chap08.interfaceex;

public class Audio implements RemoteControl {
    private int volume;

    public void turnOn() { // 추상메소드 구현
        System.out.println("Audio를 켭니다.");
    }
    public void turnOff() { // 추상메소드 구현
        System.out.println("Audio를 끕니다.");
    }
    public void setVolume(int volume) { // 추상메소드 구현
        if(volume>RemoteControl.MAX_VOLUME) {
            this.volume = RemoteControl.MAX_VOLUME;
        } else if(volume<RemoteControl.MIN_VOLUME) {
            this.volume = RemoteControl.MIN_VOLUME;
        } else {
            this.volume = volume;
        }
        System.out.println("현재 Audio 볼륨: " + this.volume);
    }
}
```

실습 - 인터페이스 구현(3)

[Ex1-4] MyClass.java

```java
package chap08.interfaceex;

public class MyClass {
    // 필드
    RemoteControl rc = new Television();

    // 생성자
    MyClass() {
    }

    MyClass(RemoteControl rc) {
        this.rc = rc;
        rc.turnOn();
        rc.setVolume(5);
    }

    // 메소드
    void methodA() {
        RemoteControl rc = new Audio();
        rc.turnOn();
        rc.setVolume(5);
    }

    void methodB(RemoteControl rc) {
        rc.turnOn();
        rc.setVolume(5);
    }
}
```

[Ex1-5] MyClassExample.java

```java
package chap08.interfaceex;

public class MyClassExample {
    public static void main(String[] args) {
        System.out.println("1)---------------");

        MyClass myClass1 = new MyClass();
        myClass1.rc.turnOn();
        myClass1.rc.setVolume(5);

        System.out.println("2)---------------");

        MyClass myClass2 = new MyClass(new Audio());

        System.out.println("3)---------------");

        MyClass myClass3 = new MyClass();
        myClass3.methodA();

        System.out.println("4)---------------");

        MyClass myClass4 = new MyClass();
        myClass4.methodB(new Television());
    }
}
```

실습 – 다중 인터페이스 구현

[Ex2-1] RemoteControl.java 는 앞의 예제(Ex1-1)와 동일

[Ex2-2] Searchable.java

```
package chap08.interfaceex;

public interface Searchable {
    void search(String url);
}
```

[Ex2-3] SmartTelevision.java

```
package chap08.interfaceex;

public class SmartTelevision implements RemoteControl, Searchable {
    private int volume;

    public void turnOn() {
        System.out.println("TV를 켭니다.");
    }
    public void turnOff() {
        System.out.println("TV를 끕니다.");
    }
    public void setVolume(int volume) {
        if(volume>RemoteControl.MAX_VOLUME) {
            this.volume = RemoteControl.MAX_VOLUME;
        } else if(volume<RemoteControl.MIN_VOLUME) {
            this.volume = RemoteControl.MIN_VOLUME;
        } else {
            this.volume = volume;
        }
        System.out.println("현재 TV 볼륨: " + this.volume);
    }

    public void search(String url) {
        System.out.println(url + "을 검색합니다.");
    }
}
```

다중 인터페이스 구현

```
package chap08.interfaceex;

public class SmartTelevisionExample {
    public static void main(String[] args) {
        SmartTelevision tv = new SmartTelevision();

        RemoteControl rc = tv;
        rc.turnOn();
        rc.setVolume(10);

        Searchable searchable = tv;
        searchable.search("http://www.google.com");
    }
}
```

[Ex2-4] SmartTelevisionExample.java

실습 – 인터페이스 다형성

[Ex3-1] Tire.java

```
package chap08.interfaceex;

public interface Tire {
    public void roll();
}
```

[Ex3-2] HankookTire.java

```
package chap08.interfaceex;

public class HankookTire implements Tire {
    @Override
    public void roll() {
        System.out.println("한국 타이어가 굴러갑니다.");
    }
}
```

[Ex3-3] dog.java

```
package chap08.interfaceex;

public class KumhoTire implements Tire {
    @Override
    public void roll() {
        System.out.println("금호 타이어가 굴러갑니다.");
    }
}
```

[Ex3-4] Car.java

```
package chap08.interfaceex;

public class Car {
    Tire frontLeftTire = new HankookTire();
    Tire frontRightTire = new HankookTire();
    Tire backLeftTire = new HankookTire();
    Tire backRightTire = new HankookTire();

    void run() {
        frontLeftTire.roll();
        frontRightTire.roll();
        backLeftTire.roll();
        backRightTire.roll();
    }
}
```

[Ex3-5] CarExample.java

```
package chap08.interfaceex;

public class CarExample {
    public static void main(String[] args) {
        Car myCar = new Car();

        myCar.run();

        myCar.frontLeftTire = new KumhoTire();
        myCar.frontRightTire = new KumhoTire();

        myCar.run();
    }
}
```

인터페이스 상속

[Ex4-1] InterfaceA.java

```java
package chap08.interfaceex;

public interface InterfaceA {
    public void methodA();
}
```

[Ex4-2] InterfaceB.java

```java
package chap08.interfaceex;

public interface InterfaceB {
    public void methodB();
}
```

[Ex4-3] InterfaceC.java

```java
package chap08.interfaceex;

public interface InterfaceC extends InterfaceA, InterfaceB {
    public void methodC();
}
```

[Ex4-4] ImplementationC.java

```java
package chap08.interfaceex;

public class ImplementationC implements InterfaceC {
    public void methodA() {
        System.out.println("ImplementationC-methodA() 실행");
    }

    public void methodB() {
        System.out.println("ImplementationC-methodB() 실행");
    }

    public void methodC() {
        System.out.println("ImplementationC-methodC() 실행");
    }
}
```

[Ex4-5] Example.java

```java
package chap08.interfaceex;

public class Example {
    public static void main(String[] args) {
        ImplementationC impl = new ImplementationC();

        InterfaceA ia = impl;
        ia.methodA();
        System.out.println();

        InterfaceB ib = impl;
        ib.methodB();
        System.out.println();

        InterfaceC ic = impl;
        ic.methodA();
        ic.methodB();
        ic.methodC();
    }
}
```