9. 인터페이스(Interface) //src: ch12_interface 참고

1) 인터페이스의 이해

(1) **작업 명세서(작업 지시서)**; 실제 구현된 것이 없는 기본 설계도

- 미리 정해진 규칙에 맞게 구현하도록 표준을 제시(표준화 가능)
- 인스턴스(객체) 생성 불가. 추상 메소드와 상수만을 가질 수 있다 cf) default method

(2) **다형성**을 가능하게 함; 하나의 객체를 다양한 type으로 만들 수 있음

(3) 객체를 **부속품화**; 다양한 객체를 부속품처럼 마음대로 변경 가능(독립적 프로그래밍可)

(4) 서로 관계 없는 클래스들에게 하나의 인터페이스를 공통 구현하도록 관계 맺기 가능

2) 인터페이스의 문법

(1) 실제 구현 된 기능 없이 추상 메소드와 상수만 존재

(2) 구현은 Implemnets 된 클래스에서 구현; private 접근 제한자 불가

```
public interface 인터페이스명{
        public static final 상수명 = 값;     //final 변수명(=상수)
        public abstract 메소드 이름(매개변수 목록); //추상 메소드만 가능
}
```

※모든 멤버 변수는 public static final이어야 하며 생략 가능

※모든 메소드는 public abstract여야 하며 생략 가능

3) 인터페이스와 다중 상속

(1) 단일 상속(O); class, interface 둘다 가능

```
Public interface InterfaceName extends IFunction {          }
public class ChildCalss extends ParentClass {          }
```

(2) 다중 구현(O); 한 클래스에서 하나 이상의 인터페이스 implements(구현) 가능

```
public class ChildCalss implements IFunction1, IFunction2, IFunction3 {          }
```

(3) 다중 상속(O); interface cf) Java 클래스는 다중 상속(X)

```
Public interface InterfaceName extends IFunction1, IFunction2, IFunction3 {     }
```

(4) 단일 상속 및 다중 구현(O); 클래스는 단일 상속, 인터페이스는 다중 구현 가능

```
public class CalssName extends SClass implements I1, I2, I3 {          }
```

※오버로딩 vs. 오버라이딩

오버로딩(overloading): 다중 정의; 동일 클래스에 같은 메소드가 매개 변수를 달리 여러 개 존재

오버라이딩(overriding): 재정의; 상속받은 메소드 내용을 변경