9주차 과제 및 실습

과제 및 실습

- **과제는 문제 및 프로그램 실습으로 나뉘어 있습니다.**
  - 반드시 실습 예제를 작성해보고 실행해봅니다.

- **제출 파일 : 보고서+소스코드**
  - 보고서 : 한글 또는 word 파일
  - 소스코드 : java 파일만 제출
  - 위 두 파일을 학번_이름_실습주차.zip으로 압축하여 제출

  (3개의 실습문제가 있다면 exec_2019111998_1.java, … exec_2019111998_3.java, 2019111998_이선호_2주차.hwp를 zip파일로 압축해서 제출)

- **보고서 내용**
  - 과제 문제 : 해답, 해답의 이유
  - 실습 문제 : 해답소스코드, 프로그램 설명, 결과 화면(결과 캡처)
    - 프로그램 설명 : 작성한 소스코드의 내용, 소스코드 내 주석으로 대체 가능
  - 보고서 파일명 : 학번_이름_실습주차.hwp
    - ex) 2023111010_홍길동_2주차.hwp

- **소스코드**
  - 파일명 : exec_학번_문제번호.java(2번 문제의 경우 exec_2019111998_2.java)
  - 클래스명이 문제에 지정된 경우 파일명은 문제에 지정된 클래스명으로 설정

- **제출 : eclass**
  - 과제 연장 제출을 희망할 경우 직접 또는 이메일을 통해 조교에게 요청

프로그램 예제1. 상속에서 슈퍼클래스의 생성자와 멤버 메서드 예제

```
class Parent{
    String name;
    public Parent() { name="허균"; }
    public Parent(String name) { this.name = name; }
    public void getName() { System.out.println("부모 : " + name); }
}

class Child extends Parent{
    String name;
    public Child() { name = "홍길동"; }
    public Child(String name, String parentName) {
        super(parentName);              //슈퍼클래스의 생성자 임의 호출
        this.name = name;
    }
    public void getName() { System.out.println("자식 : " + name); }
    public void getFamily() {
        super.getName();                //슈퍼클래스의 메소드 임의 호출
        getName();
    }
}

public class Exam1 {
    public static void main(String[] args) {
        Child c1 = new Child();
        c1.getFamily();

        Child c2 = new Child("둘리", "김길동");
        c2.getFamily();
    }
}
```

```
부모 : 허균
자식 : 홍길동
부모 : 김길동
자식 : 둘리
```

super는 슈퍼클래스를 접근하는 키워드
- super() : 슈퍼클래스의 생성자
- super : 슈퍼클래스의 객체 레퍼런스

프로그램 예제2. 상속과 추상 클래스/추상 메서드

Exam9_2. 추상클래스의 예

```
class Animal{
    String name = "동물";
    public void Eat() {System.out.println("먹이를 먹습니다.");};
}

class Person extends Animal{
    public String name;
    public void Eat() {System.out.println("밥을 먹습니다.");};
}

public class Exam2 {
    public static void main(String[] args) {
        Animal a = new Person();          //업캐스팅
        a.Eat();                          //동적 바인딩
        Person b = (Person)a;             //다운캐스팅
        b.Eat();
    }
}
```

```
밥을 먹습니다.
밥을 먹습니다.
```

```
abstract class Animal{
    String name = "동물";
    public abstract void Eat();      //추상 메소드
}

class Person extends Animal{
    public String name;
    public void Eat() {System.out.println("밥을 먹습니다.");};
}

public class Exam2 {
    public static void main(String[] args) {
        Animal a = new Person();          //업캐스팅
        a.Eat();                          //동적 바인딩
        Person b = (Person)a;             //다운캐스팅
        b.Eat();
    }
}
```

프로그램 예제3. 상속과 인터페이스

Exam9_3. 인터페이스와 다중상속의 예

- 인터페이스는 다중상속 허용
- 인터페이스와 클래스를 각각 상속받기 허용

```
갤럭시폰
전화 걸기
전화 받기
충전 하기
음악 연주
음악 정지
java로 개발
```

```java
interface phone{
    public void call();
    public void receive();
}

interface mobilephone extends phone{
    public void charge();
}

interface musicPlayer{
    public void playMusic();
    public void stopMusic();
}

class SmartPhone implements  mobilephone, musicPlayer{
    public void call() {         System.out.println("전화 걸기"); }
    public void receive() {      System.out.println("전화 받기"); }
    public void charge() {       System.out.println("충전 하기"); }
    public void playMusic() {    System.out.println("음악 연주"); }
    public void stopMusic() {    System.out.println("음악 정지"); }
}

interface develop{
    public void lang();
}

class DevAndroid extends SmartPhone implements develop{
    public String name;
    public DevAndroid(String name) { this.name = name; }
    public void lang() {     System.out.println("java로 개발"); }
}

public class Exam3 {
    public static void main(String[] args) {
        DevAndroid a = new DevAndroid("갤럭시폰");
        System.out.println(a.name);
        a.call();
        a.receive();
        a.charge();
        a.playMusic();
        a.stopMusic();
        a.lang();
    }
}
```

과제문제 : 다음의 문제를 풀고 정답의 이유도 함께 작성하세요.

1. 상속에서 슈퍼클래스와 서브클래스에 대한 설명으로 옳지 않은 것은?

① 서브 클래스의 객체를 슈퍼 클래스의 타입으로 타입 변환하는 것을 업캐스팅이라고 한다.

② 슈퍼 클래스의 레퍼런스로 서브 클래스의 멤버를 이용할 수 있다.

③ 서브 클래스의 레퍼런스가 슈퍼 클래스의 객체를 가지는 것을 다운캐스팅이라고 한다.

④ 레퍼런스가 가리키는 객체의 타입을 알아내기 위해 instanceof 연산자를 사용할 수 있다.

⑤ 다운캐스팅 할 때 슈퍼클래스의 객체를 서브클래스 레퍼런스에 대입하면 자동으로 타입 변환된다.

2. 상속에 대한 설명으로 옳지 않은 것은?

① 상속하는 클래스를 슈퍼 클래스라고 한다.

② extends 키워드를 사용하여 상속함을 나타낸다

③ 상속받은 클래스로 객체 생성 시 서브 클래스의 생성자, 슈퍼 클래스의 생성자가 순으로 실행된다.

④ 상속 구조에서 슈퍼 클래스의 여러 개의 생성자 중 선택하여 실행하기 위해서 super 키워드를 사용한다.

⑤ 상속에서 슈퍼 클래스의 생성자들 중에서 선택을 하지 않는 경우 기본 생성자가 실행된다.

3. 다형성을 위한 설명으로 옳지 않은 것은?

① 메소드 오버라이딩은 슈퍼 클래스의 메서드를 재정의하여 사용하는 것이다.

② 메소드 오버라이딩 시 메서드의 이름은 반드시 동일해야 하며, 매개변수 타입이나 개수가 달라야 성립한다.

③ 메소드 오버로딩은 상속관계 또는 동일한 클래스 내에서 사용할 수 있다.

④ 메소드 오버라이딩의 호출은 실행시간에 메서드 결정되는 동적바인딩(런타임 다형성)으로 선택된다.

⑤ 메소드 오버로딩의 컴파일하는 동안 호출될 메서드를 정적바인딩(정적 다형성)으로 구현된다.

과제문제 : 다음의 문제를 풀고 정답의 이유도 함께 작성하세요.

4. 추상 클래스에 대한 설명으로 옳지 않은 것은?

① 추상 클래스는 선언은 되어있으나 객체를 만들지 못한다.

② 클래스에 추상 메서드가 있으면 추상클래스가 된다.

③ 추상 클래스는 선언 시 abstract 키워드를 반드시 넣어야 한다.

④ 추상 클래스를 상속받는 서브 클래스는 여러 개의 추상 클래스를 상속 받을 수 있다.

⑤ 추상 클래스를 상속받으면 반드시 추상 메서드는 구현해야 한다.

5. 인터페이스에 대한 설명으로 옳지 않은 것은?

① 인터페이스는 일반 클래스와 같이 필드와 메서드를 포함한다.

② 인터페이스 내부에서 선언된 멤버 변수는 public static final이며, 멤버 메서드는 추상 메서드가 된다.

③ 인터페이스 간에 상속을 받을 수 있고, 한 인터페이스는 여러 개의 인터페이스를 상속 받을 수 있다.

④ 인터페이스에 추상 메서드가 사용된 경우 상속받은 일반 서브 클래스는 반드시 구현해야 한다.

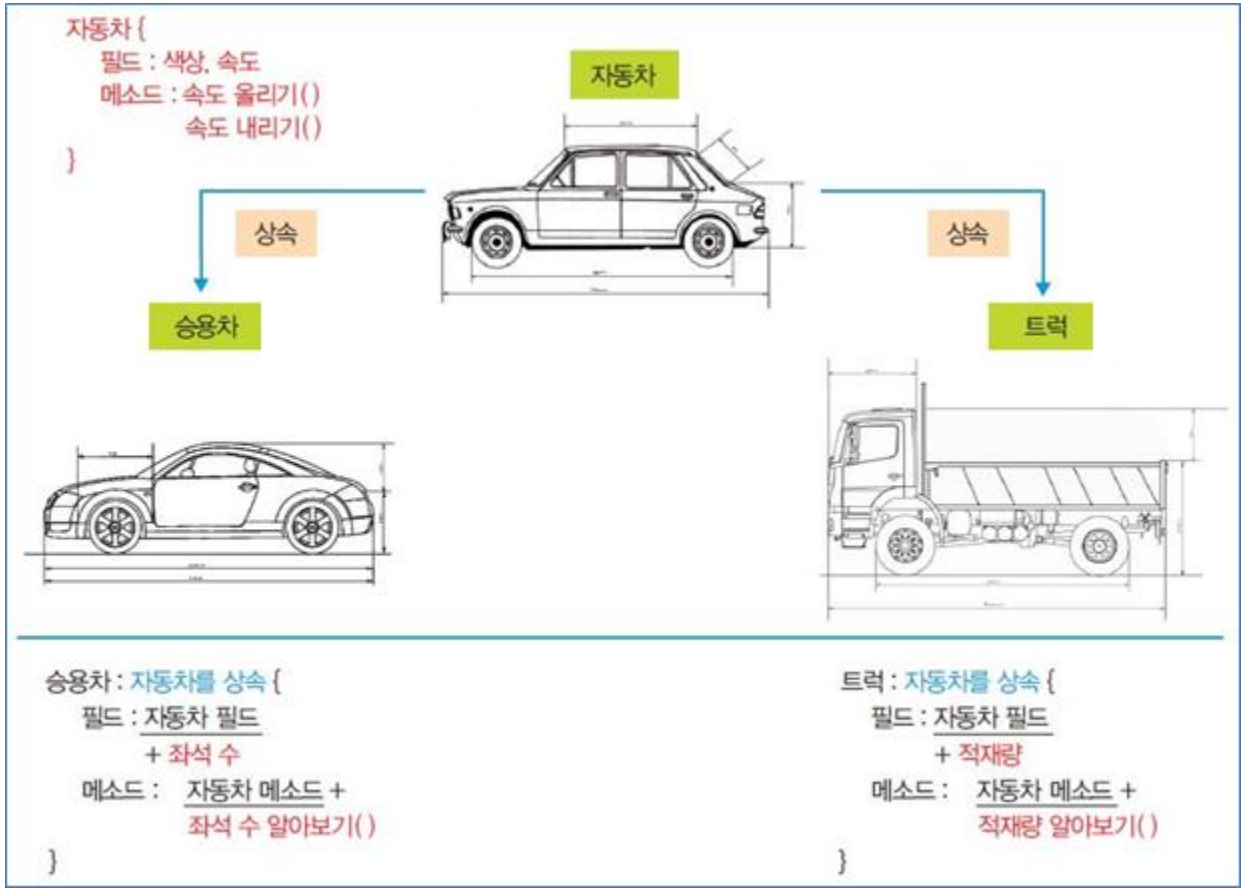
⑤ 인터페이스 간의 상속에는 extends를 사용하고, 일반 클래스로 상속 시에는 implements 키워드를 사용한다.

⑥ 클래스에서 인터페이스의 상속은 다중 상속이 허용된다.

실습 문제1

exec1) 다음 그림과 같은 상속 구조를 가지는 프로그램을 작성하라.

- 자동차와 승용차, 트럭의 상속구조에서 일반 클래스, 추상클래스, 인터페이스를 적절히 선택하여 구현하라.
- 각 메소드 호출 시 해당 동작이 수행함을 콘솔에 출력한다. ex) "자동차의 속도가 xx로 올라갑니다."
- 승용차의 객체와 트럭의 객체를 생성하여 각 메소드를 실행한 결과를 가지는 main함수도 작성하라.



실습 문제2

Exec2. 앞의 문제에서 그림에서 대포를 상속하여 탱크를 만들어라.

- 대포에는 동작한다는 메소드가 있다.
- 대포에는 발사한다는 메소드가 있다.
- 동작한다는 메소드는 “탱크가 앞으로 굴러간다”를 출력한다.
- 발사한다는 메소드는 “탱크에서 대포를 발사한다“를 출력한다.