KB금융그룹 | 국민의 평생 금융파트너

2024년 상반기 K-디지털 트레이닝

상속

[KB] IT's Your Life

KB국민은행

상속

- **다음의 Phone 클래스를 상속한 SmartPhone 클래스를 정의하세요.**
  - model, color는 SmartPhone 클래스의 생성자 매개변수로 초기화 함

```
package ch07.sec03.exam02;

public class Phone {
  public String model;
  public String color;

  public Phone(String model, String color) {
    this.model = model;
    this.color = color;
    System.out.println("Phone(String model, String color) 생성자 실행");
  }
}
```

- **SmartPhoneExample 클래스로 SmartPhone의 인스턴스를 생성하여 초기화가 올바른지 확인하세요.**

SmartPhone.java

```
package ch07.sec03.exam02;

public class SmartPhone extends Phone {
  //자식 생성자 선언
  public SmartPhone(String model, String color) {
    super(model, color);
    System.out.println("SmartPhone(String model, String color) 생성자 실행됨");
  }
}
```

SmartPhoneExample.java

```
package ch07.sec03.exam02;

public class SmartPhoneExample {

  public static void main(String[] args) {
    //SmartPhone 객체 생성
    SmartPhone myPhone = new SmartPhone("갤럭시", "은색");

    //Phone으로부터 상속 받은 필드 읽기
    System.out.println("모델: " + myPhone.model);
    System.out.println("색상: " + myPhone.color);
  }
}
```

```
Phone(String model, String color) 생성자 실행
SmartPhone(String model, String color) 생성자 실행됨
모델: 갤럭시
색상: 은색
```

상속

- **Calculator를 상속한 Computer 클래스를 정의하고, areaCircle() 메서드를 재정의하세요.**
  - Computer 클래스의 areaCircle()은 Math.PI 상수를 이용해 계산함

```
package ch07.sec04.exam01;

public class Calculator {
  //메소드 선언
  public double areaCircle(double r) {
    System.out.println("Calculator 객체의 areaCircle() 실행");
    return 3.14159 * r * r;
  }
}
```

- **ComputerExample 클래스를 정의하여, Calculator와 Computer 클래스의 areaCircle() 메서드의 계산값을 모두 출력하세요.**

Calculator.java

```
package ch07.sec04.exam01;

public class Calculator {
  //메소드 선언
  public double areaCircle(double r) {
    System.out.println("Calculator 객체의 areaCircle() 실행");
    return 3.1459 * r * r;
  }
}
```

Computer.java

```
package ch07.sec04.exam01;

public class Computer extends Calculator {
  //메소드 오버라이딩
  @Override  // 컴파일 시 정확히 오버라이딩이 되었는지 체크 해줌
  public double areaCircle(double r) {
    System.out.println("Computer 객체의 areaCircle() 실행");
    return Math.PI * r * r;
  }
}
```

ComputerExample.java

```
package ch07.sec04.exam01;

public class ComputerExample {
  public static void main(String[] args) {
    int r = 10;

    Calculator calculator = new Calculator();
    System.out.println("원 면적: " + calculator.areaCircle(r));
    System.out.println();

    Computer computer = new Computer();
    System.out.println("원 면적: " + computer.areaCircle(r));
  }
}
```

```
Calculator 객체의 areaCircle() 실행
원 면적: 314.159

Computer 객체의 areaCircle() 실행
원 면적: 314.1592653589793
```

상속

다음 클래스들을 정의하세요.

```
package ch07.sec04.exam02;

public class Airplane {

  public void land() {
    System.out.println("착륙합니다.");
  }

  public void fly() {
    System.out.println("일반 비행합니다.");
  }

  public void takeOff() {
    System.out.println("이륙합니다.");
  }
}
```

```
package ch07.sec04.exam02;

public class SupersonicAirplane extends Airplane {
  public static final int NORMAL = 1;
  public static final int SUPERSONIC = 2;

  public int flyMode = NORMAL;

  @Override
  public void fly() {
    if(flyMode == SUPERSONIC) {
      System.out.println("초음속 비행합니다.");
    } else {
      super.fly();
    }
  }
}
```

SupersonicAirplaneExample 클래스의 실행 결과를 적어보고, 실제 결과와 비교해보세요

```
package ch07.sec04.exam02;

public class SupersonicAirplaneExample {
  public static void main(String[] args) {
    SupersonicAirplane sa = new SupersonicAirplane();
    sa.takeOff();
    sa.fly();
    sa.flyMode = SupersonicAirplane.SUPERSONIC;
    sa.fly();
    sa.flyMode = SupersonicAirplane.NORMAL;
    sa.fly();
    sa.land();
  }
}
```

```
이륙합니다.
일반 비행합니다.
초음속 비행합니다.
일반 비행합니다.
착륙합니다.
```

상속

다음 코드에서 잘못된 코드를 찾고, 그 이유를 설명하세요.

```
package ch07.sec07.exam01;

class A {
}

class B extends A {
}

class C extends A {
}

class D extends B {
}

class E extends C {
}
```

```
public class PromotionExample {
  public static void main(String[] args) {
    B b = new B();
    C c = new C();
    D d = new D();
    E e = new E();

    A a1 = b;
    A a2 = c;
    A a3 = d;
    A a4 = e;

    B b1 = d;
    C c1 = e;

    // B b3 = e;    상속 관계가 없으므로 컴파일 에러
    // C c2 = d;
  }
}
```

ChildExample에서 잘못된 코드를 찾고, 그 이유를 설명하세요.

```
package ch07.sec07.exam02;

public class Parent {
  public void method1() {
    System.out.println("Parent-method1()");
  }

  public void method2() {
    System.out.println("Parent-method2()");
  }
}
```

```
package ch07.sec07.exam02;

public class Child extends Parent {
  @Override
  public void method2() {
    System.out.println("Child-method2()");
  }

  public void method3() {
    System.out.println("Child-method3()");
  }
}
```

```
package ch07.sec07.exam02;

public class ChildExample {
  public static void main(String[] args) {
    Child child = new Child();
    Parent parent = child;

    parent.method1();
    parent.method2();
    // parent.method3(); -- Parent 클래스에서는 method3이 없음
  }
}
```

상속

다음 클래스를 정의하세요.

```
package ch07.sec07.exam03;

public class Parent {
  public String field1;

  public void method1() {
    System.out.println("Parent-method1()");
  }

  public void method2() {
    System.out.println("Parent-method2()");
  }
}
```

```
package ch07.sec07.exam03;

public class Child extends Parent {
  public String field2;

  public void method3() {
    System.out.println("Child-method3()");
  }
}
```

상속

- 앞의 두 클래스를 다음과 같이 운영했을 때, 잘못된 부분을 찾아 수정하세요.

```
package ch07.sec07.exam03;

public class ChildExample {
  public static void main(String[] args) {

    Parent parent = new Child();

    parent.field1 = "data1";
    parent.method1();
    parent.method2();
    parent.field2 = "data2";
    parent.method3();

    Child child = (Child) parent;

    child.field2 = "data2";
    child.method3();
  }
}
```

상속

- 앞의 두 클래스를 다음과 같이 운영했을 때, 잘못된 부분을 찾아 수정하세요.

```
package ch07.sec07.exam03;

public class ChildExample {
  public static void main(String[] args) {

    Parent parent = new Child();

    parent.field1 = "data1";
    parent.method1();
    parent.method2();
    // parent.field2 = "data2";  -- field2는 Child 클래스의 멤버임, Parent 타입으로 사용 불가
    // parent.method3();         -- method3는 Child 클래스의 멤버임, Parent 타입으로 사용 불가

    Child child = (Child) parent;

    child.field2 = "data2";
    child.method3();
  }
}
```

상속

다음 클래스들을 정의하세요.

```
package ch07.sec08.exam01;

public class Tire {
  public void roll() {
    System.out.println("회전합니다.");
  }
}
```

```
package ch07.sec08.exam01;

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

```
package ch07.sec08.exam01;

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

상속

다음 클래스를 정의하고, CarExample의 실행결과가 다음과 같도록 수정하세요.

```
package ch07.sec08.exam01;

public class Car {
  public Tire tire;

  public void run() {
    tire.roll();
  }
}
```

```
package ch07.sec08.exam01;

public class CarExample {
  public static void main(String[] args) {

    Car myCar = new Car();


  }
}
```

```
회전합니다.
한국 타이어가 회전합니다.
금호 타이어가 회전합니다.
```

CarExample.java

```
package ch07.sec08.exam01;

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

    //Tire 객체 장착
    myCar.tire = new Tire();
    myCar.run();

    //HankookTire 객체 장착
    myCar.tire = new HankookTire();
    myCar.run();

    //KumhoTire 객체 장착
    myCar.tire = new KumhoTire();
    myCar.run();
  }
}
```

```
회전합니다.
한국 타이어가 회전합니다.
금호 타이어가 회전합니다.
```

상속

다음 클래스들을 정의하세요.

```
package ch07.sec08.exam02;

public class Vehicle {
  public void run() {
    System.out.println("차량이 달립니다.");
  }
}
```

```
package ch07.sec08.exam02;

public class Bus extends Vehicle {
  @Override
  public void run() {
    System.out.println("버스가 달립니다.");
  }
}
```

```
package ch07.sec08.exam02;

public class Taxi extends Vehicle {
  @Override
  public void run() {
    System.out.println("택시가 달립니다.");
  }
}
```

상속

다음 클래스들을 정의하고, 출력결과가 다음과 같이 나오도록 DriverExample을 완성하세요.

```
package ch07.sec08.exam02;

public class Driver {
  public void drive(Vehicle vehicle) {
    vehicle.run();
  }
}
```

```
package ch07.sec08.exam02;

public class DriverExample {
  public static void main(String[] args) {
    Driver driver = new Driver();

    Bus bus = new Bus();
    _____________________;

    Taxi taxi = new Taxi();
    _____________________;

  }
}
```

```
버스가 달립니다.
택시가 달립니다.
```

DriverExample

```
package ch07.sec08.exam02;

public class DriverExample {
  public static void main(String[] args) {
    //Driver 객체 생성
    Driver driver = new Driver();

    //매개값으로 Bus 객체를 제공하고 driver() 메소드 호출
    Bus bus = new Bus();
    driver.drive(bus);       // driver.drive(new Bus()); 와 동일

    //매개값으로 Taxi 객체를 제공하고 driver() 메소드 호출
    Taxi taxi = new Taxi();
    driver.drive(taxi);      // driver.drive(new Taxi()); 와 동일
  }
}
```

```
버스가 달립니다.
택시가 달립니다.
```

상속

다음 클래스를 추상 클래스로 변경하세요.

```
package ch07.sec10.exam01;

public class Phone {
  String owner;

  Phone(String owner) {
    this.owner = owner;
  }

  void turnOn() {
    System.out.println("폰 전원을 켭니다.");
  }

  void turnOff() {
    System.out.println("폰 전원을 끕니다.");
  }
}
```

Phone.java

```
package ch07.sec10.exam01;

public abstract class Phone {
  //필드 선언
  String owner;

  //생성자 선언
  Phone(String owner) {
    this.owner = owner;
  }

  //메소드 선언
  void turnOn() {
    System.out.println("폰 전원을 켭니다.");
  }

  void turnOff() {
    System.out.println("폰 전원을 끕니다.");
  }
}
```

상속

앞에서 정의한 Phone 클래스를 상속받는 SmartPhone 클래스를 완성하세요.

```
package ch07.sec10.exam01;

public class SmartPhone _____ {
  SmartPhone(String owner) {
    _____;
  }

  void internetSearch() {
    System.out.println("인터넷 검색을 합니다.");
  }
}
```

SmartPhone.java

```
package ch07.sec10.exam01;

public class SmartPhone extends Phone {
  //생성자 선언
  SmartPhone(String owner) {
    //Phone 생성자 호출
    super(owner);
  }

  //메소드 선언
  void internetSearch() {
    System.out.println("인터넷 검색을 합니다.");
  }
}
```

상속

- 앞에서 정의한 클래스를 다음과 같이 운영하였다. 잘못된 부분을 찾아 수정하세요.

```
package ch07.sec10.exam01;

public class PhoneExample {
  public static void main(String[] args) {
    Phone phone = new Phone();

    SmartPhone smartPhone = new SmartPhone("홍길동");

    smartPhone.turnOn();
    smartPhone.internetSearch();
    smartPhone.turnOff();
  }
}
```

PhoneExample.java

```
package ch07.sec10.exam01;

public class PhoneExample {
  public static void main(String[] args) {
    //Phone phone = new Phone();

    SmartPhone smartPhone = new SmartPhone("홍길동");

    smartPhone.turnOn();
    smartPhone.internetSearch();
    smartPhone.turnOff();
  }
}
```

```
폰 전원을 켭니다.
인터넷 검색을 합니다.
폰 전원을 끕니다.
```

상속

- 다음 클래스에 리턴값이 없는 추상 메서드 sound()를 추가하세요.

```
package ch07.sec10.exam02;

public abstract class Animal {
  public void breathe() {
    System.out.println("숨을 쉽니다.");
  }

}
```

Animal.java

```
package ch07.sec10.exam02;

public abstract class Animal {
  //메소드 선언
  public void breathe() {
    System.out.println("숨을 쉽니다.");
  }

  //추상 메소드 선언
  public abstract void sound();
}
```

상속

- Animal 클래스를 상속받아 완전한 Dog 클래스를 정의하세요.

```
package ch07.sec10.exam02;

public class Dog {

}
```

- Animal 클래스를 상속받아 완전한 Cat 클래스를 정의하세요.

```
package ch07.sec10.exam02;

public class Cat {

}
```

Dog.java

```
package ch07.sec10.exam02;

public class Dog extends Animal {
  //추상 메소드 재정의
  @Override
  public void sound() {
    System.out.println("멍멍");
  }
}
```

Cat.java

```
package ch07.sec10.exam02;

public class Cat extends Animal {
  //추상 메소드 재정의
  @Override
  public void sound() {
    System.out.println("야옹");
  }
}
```

상속

- 앞에서 정의한 클래스를 이용하여,다음 출력이 나오도록 코드를 완성하세요.

```
package ch07.sec10.exam02;

public class AbstractMethodExample {
  public static void main(String[] args) {
    Dog dog = new Dog();
    dog.sound();

    Cat cat = new Cat();
    cat.sound();

    animalSound(new Dog());
    animalSound(new Cat());
  }


}
```

```
멍멍
야옹
멍멍
야옹
```

AbstractMethodExample.java

```
package ch07.sec10.exam02;

public class AbstractMethodExample {
  public static void main(String[] args) {
    Dog dog = new Dog();
    dog.sound();

    Cat cat = new Cat();
    cat.sound();

    //매개변수의 다형성
    animalSound(new Dog());
    animalSound(new Cat());
  }

  public static void animalSound( Animal animal ) {
    animal.sound(); // 재정의된 메소드 호출
  }
}
```

자동 타입 변환

```
멍멍
야옹
멍멍
야옹
```