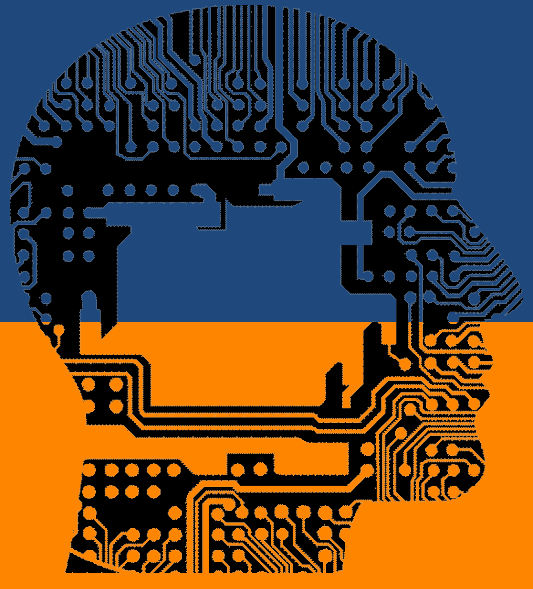
한국IT진흥부설 정보보호교육학원 아이섹



자바 프로그래밍 기초

OOP - 확인문제

By SoonGu Hong(Kokono)

1번

1. 자바의 상속에 대한 설명 중 틀린 것은?

① 자바는 다중 상속을 허용한다.
② 부모의 메서드를 자식 클래스에서 오버라이딩 할 수 있다.
③ 부모의 private 멤버는 상속대상이 아니다.
④ final클래스는 상속할 수 없고, final메서드는 오버라이딩할 수 없다.

2. 클래스 타입 변환에 대한 설명중 틀린 것은?

① 자식 객체는 부모 타입으로 자동 타입 변환될 수 있다.
② 부모 객체는 항상 자식 타입으로 강제 변환 될 수 있다.
③ 자동 타입 변환을 이용해서 필드와 매개변수에 다형성을 구현한다.
④ 강제 타입 변환 전에 instanceof로 변환 가능성을 검사하는 것이 좋다.

3. final키워드에 대한 설명으로 틀린 것은?

① final클래스는 부모 클래스로 사용할 수 있다.
② final필드는 값이 저장된 후에는 변경 불가능하다.
③ final메서드는 오버라이딩 할 수 없다.
④ static final 필드는 상수를 말한다.

4. 오버라이딩에 대한 설명으로 틀린 것은?

① 부모 메서드의 시그니처(리턴 타입, 메서드명, 매개 변수)와 동일해야 한다.
② 부모 메서드보다 접근제한이 같거나 엄격해야 한다.
(ex: public -> private)
③ @Override 어노테이션을 사용하면 재정의가 맞는지 컴파일러가 검증해준다.
④ 추상클래스의 추상메서드는 자식클래스에서 반드시 오버라이딩해야 한다.

5. 메서드 오버로딩에 대한 설명으로 틀린 것은?

① 오버로딩은 동일한 이름의 메서드를 여러개 선언하는 것을 말한다.
② 반드시 리턴 타입이 달라야 한다.
③ 매개 변수의 타입, 수, 순서를 다르게 선언해야 한다.
④ 매개값의 타입 및 수에 따라 호출될 메서드가 선택된다.

6. 인스턴스 멤버와 정적 멤버에 대한 설명으로 틀린 것은?

① 정적 멤버는 static으로 선언된 필드와 메서드를 말한다.
② 인스턴스 필드는 생성자 및 정적 블록에서 초기화될 수 있다.
③ 정적 필드와 정적 메서드는 객체 생성 없이 클래스를 통해 직접 접근할 수 있다.
④ 인스턴스 필드와 메서드는 객체를 생성하고 사용해야 한다.

7. 접근 제한에 대한 설명으로 틀린 것은?

① 접근 제한자는 클래스, 필드, 메서드, 생성자의 사용을 제한한다.
② public 접근 제한은 아무런 제한 없이 해당 요소를 사용할 수 있게 한다.
③ default 접근 제한은 해당 클래스 내부에서만 사용을 허가한다.
④ 외부에서 접근하지 못하도록 하려면 private 접근 제한을 해야한다.

8. Parent클래스를 상속해서 Child클래스를 작성했는데, 컴파일 에러가 발생했습니다. 그 이유를 쓰세요.

```
public class Parent {

   public String name;

   public Parent(String name) {
      this.name = name;
   }
}
```

```
public class Child extends Parent {

   public Child(String name) {
      this.name = name;
   }
}
```

9. 다음 코드의 실행 결과 출력될 내용을 쓰세요.

```
public class Parent {

  public String nation;

  public Parent() {
     this("대한민국");
     System.out.println("Parent() call");
  }

  public Parent(String nation) {
     this.nation = nation;
     System.out.println("Parent(String) call");
  }
}
```

```
public class Child extends Parent {
   private String name;

   public Child() {
     this("홍길동");
     System.out.println("Child() call");
   }
   public Child(String name) {
     this.name = name;
     System.out.println("Child(String) call");
   }
}
```

```
public class Main {
   public static void main(String[] args) {
      Child c = new Child();
   }
}
```

10. 다음 코드의 실행 결과 출력될 내용을 쓰세요.

```
public class Tire {

  public void roll() {
    System.out.println("일반 타이어가 회전합
니다.");
  }
}
```

```
public class SnowTire extends Tire {

  @Override
  public void roll() {
    System.out.println("스노우 타이어가 회전합
니다.");
  }
}
```

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

     SnowTire snowTire = new SnowTire();
     Tire tire = snowTire;

     snowTire.roll();
     tire.roll();
   }
}
```

감사합니다
THANK YOU