# Chapter

지네릭스,

열거형,

애너테이션

Java의 정석定石 3판 - 연습문제 풀이

115

Generics, Enumeration, Annotation

## [ 연습문제 ]

[12-1] 클래스 Box가 다음과 같이 정의되어 있을 때, 다음 중 오류가 발생하는 문장은? 경고가 발생하는 문장은?

class Box<**T**> { // 지네릭 타입 T를 선언

**T** item;

void setItem(T item) { this.item = item; }

**T** getItem() { return item; }

}

* 1. Box<Object> b = new Box<String>();
  2. Box<Object> b = (Object)new Box<String>();
  3. new Box<String>().setItem(new Object());
  4. new Box<String>().setItem("ABC");

[12-2] 지네릭 메서드 makeJuice()가 아래와 같이 정의되어 있을 때, 이 메서드를 올바 르게 호출한 문장을 모두 고르시오. (Apple과 Grape는 Fruit의 자손이라고 가정하자.)

class Juicer {

static **<T extends Fruit>** String makeJuice(FruitBox**<T>** box) { String tmp = "";

for(Fruit f : box.getList()) tmp += f + " "; return tmp;

}

}

1. Juicer.<Apple>makeJuice(new FruitBox<Fruit>());
2. Juicer.<Fruit>makeJuice(new FruitBox<Grape>());
3. Juicer.<Fruit>makeJuice(new FruitBox<Fruit>());
4. Juicer.makeJuice(new FruitBox<Apple>());
5. Juicer.makeJuice(new FruitBox<Object>());

[12-3] 다음 중 올바르지 않은 문장을 모두 고르시오.

class Box<**T extends Fruit**> { // 지네릭 타입 T를 선언

**T** item;

void setItem(T item) { this.item = item; }

**T** getItem() { return item; }

}

1. Box<?> b = new Box();
2. Box<?> b = new Box<>();
3. Box<?> b = new Box<Object>();
4. Box<Object> b = new Box<Fruit>();
5. Box b = new Box<Fruit>();
6. Box<? extends Fruit> b = new Box<Apple>();
7. Box<? extends Object> b = new Box<? extends Fruit>();

[12-4] 아래의 메서드는 두 개의

ArrayList를 매개변수로 받아서,

하나의 새로운

ArrayList로 병합하는 메서드이다. 이를 지네릭 메서드로 변경하시오.

public static ArrayList**<? extends Product>** merge(

ArrayList**<? extends Product>** list, ArrayList**<? extends Product>** list2) { ArrayList**<? extends Product>** newList = new ArrayList<>(list);

newList.addAll(list2); return newList;

}

[12-5] 아래는 예제7-3에 열거형 Kind와 Number를 새로 정의하여 적용한 것이다. (1)에 알맞은 코드를 넣어 예제를 완성하시오. (Math.random()을 사용했으므로 실행결과가 달라 질 수 있다.)

int index = (int)(Math.random() \* CARD\_NUM); return pick(index);

}

void shuffle() { // 카드의 순서를 섞는다.

for(int i=0; i < cardArr.length; i++) {

int r = (int)(Math.random() \* CARD\_NUM);

// Deck에서 카드 하나를 선택한다.

Card pick() {

class DeckTest {

public static void main(String args[]) {

Deck d = new Deck(); // 카드 한 벌(Deck)을 만든다. Card c = d.pick(0); // 섞기 전에 제일 위의 카드를 뽑는다.

System.out.println(c); // System.out.println(c.toString());과 같다.

d.shuffle(); // 카드를 섞는다.

c = d.pick(0); // 섞은 후에 제일 위의 카드를 뽑는다. System.out.println(c);

}

}

class Deck {

final int CARD\_NUM = Card.Kind.values().length

\* Card.Number.values().length; // 카드의 개수

Card cardArr[] = new Card[CARD\_NUM]; // Card객체 배열을 포함

Deck () {

**/\***

**(1)** 알맞은 코드를 넣어서 완성하시오**. Deck**의 카드를 초기화한다**.**

**\*/**

}

Card pick(int index) { // 지정된 위치(index)에 있는 카드 하나를 꺼내서 반환

return cardArr[index];

}

[연습문제]**/ch12/Exercise12\_5.java**

Card temp = cardArr[i]; cardArr[i] = cardArr[r]; cardArr[r] = temp;

}

}

} // Deck클래스의 끝

// Card클래스 class Card {

enum Kind { CLOVER, HEART, DIAMOND, SPADE }

enum Number {

ACE, TWO, THREE, FOUR, FIVE, SIX, SEVEN, EIGHT, NINE, TEN, JACK, QUEEN, KING

}

Kind kind; Number num;

Card() {

this(Kind.SPADE, Number.ACE);

}

Card(Kind kind, Number num) { this.kind = kind; this.num = num;

}

public String toString() {

return "[" + kind.name() + "," + num.name() + "]";

} // toString()의 끝

} // Card클래스의 끝

[CLOVER,ACE] [HEART,TEN]

[실행결과]

[12-6] 다음 중 메타 애너테이션이 아닌 것을 모두 고르시오.

1. Documented
2. Target
3. Native
4. Inherited

[12-7] 애너테이션 TestInfo가 다음과 같이 정의되어 있을 대, 이 애너테이션이 올바르 게 적용되지 않은 것은?

@interface TestInfo {

int count() default 1; String[] value() default "aaa";

}

* 1. @TestInfo class Exercise12\_7 {}
  2. @TestInfo(1) class Exercise12\_7 {}
  3. @TestInfo("bbb") class Exercise12\_7 {}
  4. @TestInfo("bbb","ccc") class Exercise12\_7 {}