Chapter 12

지네릭스, 열거형, 애너테이션
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; }
}
```

a. Box<Object> b = new Box<String>();
b. Box<Object> b = (Object)new Box<String>();
c. new Box<String>().setItem(new Object());
d. new Box<String>().setItem("ABC");

[12-2] 지네릭 메서드 makeJuice()가 아래와 같이 정의되어 있을 때, 이 메서드를 올바르게 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;
    }
}
```

a. Juicer.<Apple>makeJuice(new FruitBox<Fruit>());
b. Juicer.<Fruit>makeJuice(new FruitBox<Grape>());
c. Juicer.<Fruit>makeJuice(new FruitBox<Fruit>());
d. Juicer.makeJuice(new FruitBox<Apple>());
e. 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; }
}
```

a. Box<?> b = new Box();
b. Box<?> b = new Box<>();
c. Box<?> b = new Box<Object>();
d. Box<Object> b = new Box<Fruit>();
e. Box   b = new Box<Fruit>();
f. Box<? extends Fruit> b = new Box<Apple>();
g. 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 달라질 수 있다.)

[연습문제]/ch12/Exercise12_5.java

```
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.
                Deck의 카드를 초기화한다.
        */
   }

   Card pick(int index) {     // 지정된 위치(index)에 있는 카드 하나를 꺼내서 반환
        return cardArr[index];
   }

   Card pick() {          // Deck에서 카드 하나를 선택한다.
        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);
```

```
            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
```

[실행결과]

```
[CLOVER,ACE]
[HEART,TEN]
```

[12-6] 다음 중 메타 애너테이션이 아닌 것을 모두 고르시오.

a. Documented
b. Target
c. Native
d. Inherited

[12-7] 애너테이션 TestInfo가 다음과 같이 정의되어 있을 때, 이 애너테이션이 올바르게 적용되지 않은 것은?

```
@interface TestInfo {
    int count() default 1;
    String[] value() default "aaa";
}
```

a. @TestInfo class Exercise12_7 {}
b. @TestInfo(1) class Exercise12_7 {}
c. @TestInfo("bbb") class Exercise12_7 {}
d. @TestInfo("bbb","ccc") class Exercise12_7 {}

Chapter 13

쓰레드

thread

[연습문제]

[13-1] 쓰레드를 구현하는 방법에는 Thread클래스로부터 상속받는 것과 Runnable인터페이스를 구현하는 것 두 가지가 있는데, 다음의 코드는 Thread클래스를 상속받아서 쓰레드를 구현한 것이다. 이 코드를 Runnable인터페이스를 구현하도록 변경하시오.

[연습문제]/ch13/Exercise13_1.java

```
class Exercise13_1 {
   public static void main(String args[]) {
        Thread1 th1 = new Thread1();

        th1.start();
   }
}

class Thread1 extends Thread {
   public void run() {
        for(int i=0; i < 300; i++) {
            System.out.print('-');
        }
   }
}
```

[13-2] 다음 코드의 실행결과로 옳은 것은?

[연습문제]/ch13/Exercise13_2.java

```
class Exercise13_2 {
   public static void main(String[] args) {
        Thread2 t1 = new Thread2();
        t1.run();

        for(int i=0; i < 10; i++)
            System.out.print(i);
   }
}

class Thread2 extends Thread {
   public void run() {
        for(int i=0; i < 10; i++)
            System.out.print(i);
   }
}
```

a. 0102123345456768978 9처럼 0부터 9까지의 숫자가 섞여서 출력된다.
b. 0123456789 0123456789처럼 0부터 9까지의 숫자가 순서대로 출력된다.
c. IllegalThreadStateException이 발생한다.

[13-3] 다음 중 쓰레드를 일시정지 상태(WAITING)로 만드는 것이 아닌 것은? (모두 고르시오)

a. suspend()
b. resume()
c. join()
d. sleep()
e. wait()
f. notify()

[13-4] 다음 중 interrupt()에 의해서 실행대기 상태(RUNNABLE)가 되지 않는 경우는? (모두 고르시오)

a. sleep()에 의해서 일시정지 상태인 쓰레드
b. join()에 의해서 일시정지 상태인 쓰레드
c. wait()에 의해서 일시정지 상태인 쓰레드
d. suspend()에 의해서 일시정지 상태인 쓰레드

[13-5] 다음의 코드를 실행한 결과를 예측하고, 직접 실행한 결과와 비교하라. 만일 예측한 결과와 실행한 결과의 차이가 있다면 그 이유를 설명하라.

[연습문제]/ch13/Exercise13_5.java

```
class Exercise13_5
{
   public static void main(String[] args) throws Exception
   {
        Thread3 th1 = new Thread3();
        th1.start();

        try {
            Thread.sleep(5*1000);
        } catch(Exception e) {}

        throw new Exception("꽝~!!!");
   }
}

class Thread3 extends Thread {
   public void run() {
        for(int i=0; i < 10; i++) {
            System.out.println(i);

            try {
                Thread.sleep(1000);
            } catch(Exception e) {}
        }
   } // run()
}
```

[13-6] 다음의 코드를 실행한 결과를 예측하고, 직접 실행한 결과와 비교하라. 만일 예측한 결과와 실행한 결과의 차이가 있다면 그 이유를 설명하라.

[연습문제]/ch13/Exercise13_6.java

```
class Exercise13_6
{
   public static void main(String[] args) throws Exception
   {
        Thread4 th1 = new Thread4();
        th1.setDaemon(true);
        th1.start();

        try {
            th1.sleep(5*1000);
        } catch(Exception e) {}

        throw new Exception("꽝~!!!");
   }
}

class Thread4 extends Thread {
   public void run() {
        for(int i=0; i < 10; i++) {
            System.out.println(i);

            try {
                Thread.sleep(1000);
            } catch(Exception e) {}
        }
   } // run()
}
```

[13-7] 다음의 코드는 쓰레드 th1을 생성해서 실행시킨 다음 6초 후에 정지시키는 코드이다. 그러나 실제로 실행시켜보면 쓰레드를 정지시킨 다음에도 몇 초가 지난 후에서야 멈춘다. 그 이유를 설명하고, 쓰레드를 정지시키면 지체없이 바로 정지되도록 코드를 개선하시오.

[연습문제]/ch13/Exercise13_7.java

```
class Exercise13_7
{
   static boolean stopped = false;

   public static void main(String[] args)
   {
        Thread5 th1 = new Thread5();
        th1.start();

        try {
            Thread.sleep(6*1000);
        } catch(Exception e) {}

        stopped = true;   // 쓰레드를 정지시킨다.
        System.out.println("stopped");
   }
}

class Thread5 extends Thread {
   public void run() {
        // Exercise13_7.stopped의 값이 false인 동안 반복한다.
        for(int i=0; !Exercise13_7.stopped; i++) {
            System.out.println(i);

            try {
                Thread.sleep(3*1000);
            } catch(Exception e) {}
        }
   } // run()
}
```

[실행결과]

```
0
1
2
stopped
```

[13-8] 다음의 코드는 텍스트기반의 타자연습게임인데 WordGenerator라는 쓰레드가 Vector에 2초마다 단어를 하나씩 추가하고, 사용자가 단어를 입력하면 Vector에서 일치하는 단어를 삭제하도록 되어 있다. WordGenerator의 run()을 완성하시오.

[연습문제]/ch13/Exercise13_8.java

```
import java.util.*;

class Exercise13_8
{
   Vector words = new Vector();
   String[] data = {"태연","유리","윤아","효연","수영","서현","티파니","써니","제시카"};

   int interval = 2 * 1000;  // 2초

   WordGenerator wg = new WordGenerator();

   public static void main(String args[])
   {
        Exercise13_9 game = new Exercise13_9();

        game.wg.start();

        Vector words = game.words;

        while(true) {
            System.out.println(words);

            String prompt = ">>";
            System.out.print(prompt);

            // 화면으로부터 라인단위로 입력받는다.
            Scanner s = new Scanner(System.in);
            String input = s.nextLine().trim();

            int index = words.indexOf(input);

            if(index!=-1) {
                words.remove(index);
            }
        }
   } // main

   class WordGenerator extends Thread {
        public void run() {
            /*
                (1) 아래의 로직에 맞게 코드를 작성하시오.
                1. interval(2초)마다 배열 data의 값 중 하나를 임의로 선택해서
                2. words에 저장한다.
            */
        } // end of run()
   } // class WordGenerator
} // Exercise13_9
```

[실행결과]

```
[]
>>
[서현]
>>서현
[수영, 윤아]
>>수영
[윤아, 유리]
>>유리
[윤아, 티파니]
>>티파니
[윤아, 윤아, 유리]
>>윤아
[윤아, 유리]
>>유리
[윤아, 효연]
>>효연
[윤아, 티파니]
>>윤아
[티파니, 윤아]
>>티파니
[윤아, 수영, 써니]
>>
```

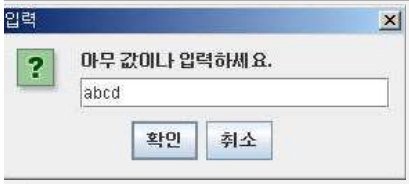
[13-9] 다음은 사용자의 입력을 출력하고 종료하는 프로그램을 작성한 것으로, 10초 동안 입력이 없으면 자동종료되어야 한다. 그러나 실행결과를 보면, 사용자의 입력이 10초 안에 이루어졌음에도 불구하고 프로그램이 즉시 종료되지 않는다. 사용자로부터 입력받는 즉시 프로그램이 종료되도록 수정하시오.

[연습문제]/ch13/Exercise13_9.java

```
import javax.swing.JOptionPane;

class Exercise13_9 {
   public static void main(String[] args) throws Exception     {
        Exercise13_9_1 th1 = new Exercise13_9_1();
        th1.start();

        String input = JOptionPane.showInputDialog("아무 값이나 입력하세요.");
        System.out.println("입력하신 값은 " + input + "입니다.");
        th1.interrupt();  // 쓰레드에게 작업을 멈추라고 요청한다.
   }
}

class Exercise13_9_1 extends Thread {
   public void run() {
        int i = 10;

        while(i!=0 && !isInterrupted()) {
```

```
            System.out.println(i--);

            try {
                Thread.sleep(1000);  // 1초 지연
            } catch(InterruptedException e) {}
        }

        System.out.println("카운트가 종료되었습니다.");
    } // main
}
```

[실행결과]




```
10
9
8
입력하신 값은 abcd입니다.
7
6
5
4
3
2
1
카운트가 종료되었습니다.
```