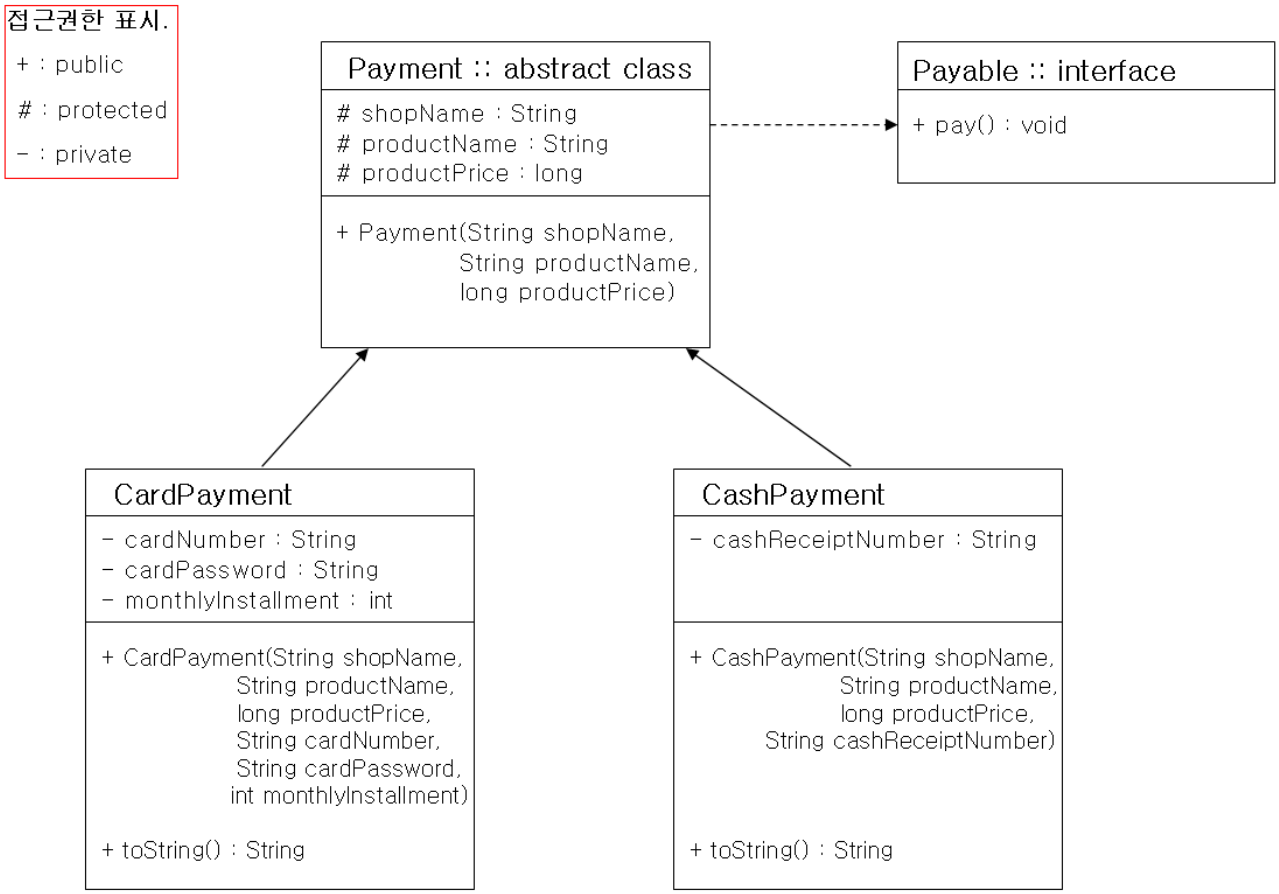
문제1. 그림과 같은 상속관계를 나타내고 있을 때, 아래 질문에 대한 클래스를 작성하십시오.

PayMentTest 의 main() 의 내용을 참고하시고, PayMentTest 클래스는 는 수정하지 마세요.

주어진 Payable, PayException 은 그대로 사용하세요~!



I. 클래스 다이어그램의 내용을 토대로 각 클래스 및 인터페이스를 작성하십시오

A. Payment 추상 클래스를 작성하십시오. 멤버 변수로는 상점명(shopName), 상품명(productName), 상품가격(productPrice) 가 존재합니다. 멤버변수를 초기화하는 생성자를 가지고 있으며, Payable 인터페이스를 사용토록 기술하세요. (구현은 Payment 를 상속받은 클래스에서 하도록 합니다)

B. CardPayment 클래스를 작성하십시오. 멤버 변수로는 신용카드번호(cardNumber), 카드비밀번호(cardPassword), 할부개월(monthlyInstallment) 이 존재하며, 멤버변수를 초기화하는 생성자를 가지고 있습니다. 또한 toString 메서드를 아래 실행 예시를 확인하여 오버라이딩 하세요.

C. CashPayment 클래스를 작성하십시오. 멤버 변수로는 현금영수증번호(cashReceiptNumber)가 존재하며, 멤버변수를 초기화하는 생성자를 가지고 있습니다. 또한 toString 메서드를 아래 실행 예시를 확인하여 오버라이딩 하세요.

D. pay 메소드는 상품가격이 0 이하이거나, 신용카드 할부개월수가 음수인 경우에는 PayException 을 던집니다. 그렇지 않은 경우에는 정상적으로 지불됨을 출력합니다. (출력포맷은 아래 내용 참고)

[PaymentTest 클래스 실행 예시]

```
신용카드가 정상적으로 지불되었습니다.
[ 신용카드 결제 정보 ]
상점명 : 11번가
상품명 : Java책
상품가격 : 17000
신용카드번호 : 123-432-111
할부개월 : 0
---------------------------------------
현금이 정상적으로 지불되었습니다.
[ 현금 결제 정보 ]
상점명 : 인터파크
상품명 : 에어컨
상품가격 : 2400000
현금영수증번호 : 198-32
---------------------------------------
```

[Exception 예제 1 : 주석을 풀고 실행 시 결과의 예]

```
Exception in thread "main" PayException: 가격이나 할부개월수가 잘못되었습니다.
        at CardPayment.pay(CardPayment.java:25)
        at PayMentTest.payProcess(PayMentTest.java:30)
        at PayMentTest.main(PayMentTest.java:20)
```

[Exception 예제 2 : 주석을 풀고 실행 시 결과의 예]

```
Exception in thread "main" PayException: 가격이 잘못되었습니다.
        at CashPayment.pay(CashPayment.java:19)
        at PayMentTest.payProcess(PayMentTest.java:30)
        at PayMentTest.main(PayMentTest.java:24)
```

문제2. 응용 프로그램의 환경설정을 하기 위한 목적으로 myserver.ini 파일에 서버 및 포트, 이메일 정보를 저장하였다. 이 환경설정 파일을 읽어서 출력할 수 있도록 아래의 ConfigurationReader 클래스를 완성하시오.

```
<< myserver.ini 파일 내용 >>

# 다음은 포트 번호입니다.
port=>7690
# 다음은 서버 이름입니다.
server=>sds.prestc.com
# 다음은 관리자 메일입니다.
admin=>admin@sds.prestc.com
```

```
import java.io.*;
import java.util.*;

public class ConfigurationReader {
    String file;
    char comment;
    String  delm;
    Hashtable ht;

    public ConfigurationReader(String file) {
        this.file = file;
        this.comment  = '#';   // 주석 구분자 # 기호.
        this.delm = "=>";      // 서버 정보 구분자 = 기호.
        ht = new Hashtable();  // 정보를 파싱하여 key, value 쌍으로 저장하기 위한 공간
    }

    public String getValue(String name) {
        // 프로그램 구현 부분 ------------------------
        // 해쉬테이블의 키에 해당하는 값을 리턴시키도록 구현.

        // ----------------------------------------
    }
```

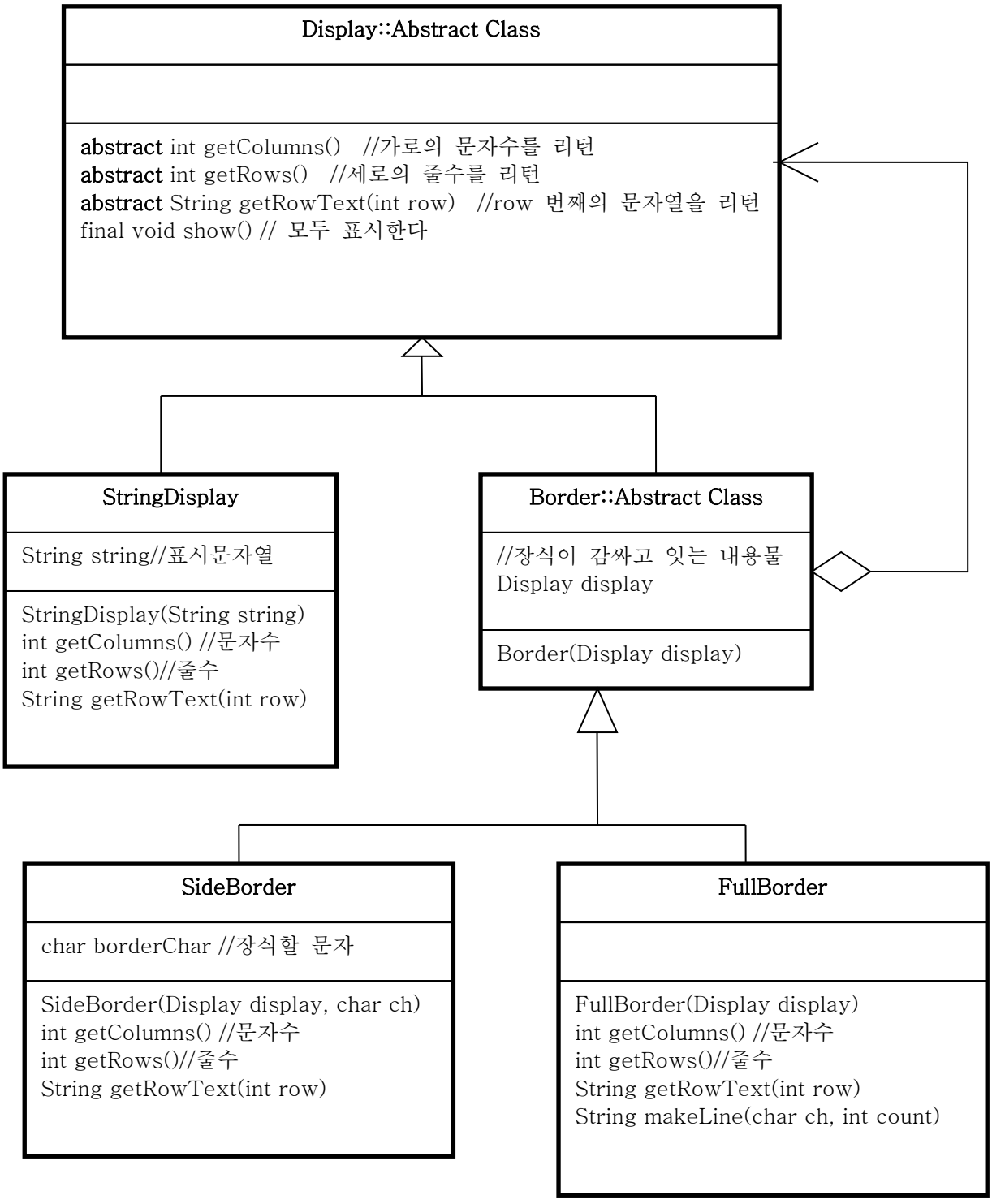
```
    public void parse() throws IOException {

        // 프로그램 구현 부분 -------------------------

        // 1. 파일로부터 내용을 읽어내어  파싱 하는 부분이다.
        // 2. 읽은 첫 글자가 '#' 인 경우는 주석이므로 다음라인을 읽고,
        // 3. 주석이 아닌 경우는 서버 정보이므로.. 서버정보 구분자인 delm ( "=>" )을
        //    이용하여 key, value 를 추출하여 해쉬 테이블에 저장하는 기능을 완성하라.


        //-----------------------------------------
    }

    public static void main(String args[]) {  // main 메쏘드는 변경하지 마세요.

        ConfigurationReader cr =
                new ConfigurationReader("myserver.ini");

        try {
            cr.parse();
            System.out.println(cr.getValue("port"));
            System.out.println(cr.getValue("server"));
            System.out.println(cr.getValue("admin"));
        } catch (Exception e) { }

    }
}
```

[ConfigurationReader 실행결과]

```
    7690
    sds.prestc.com
    admin@sds.prestc.com
```

문제3. Display 클래스와 Display 클래스를 상속받는 StringDisplay, Border, 그리고 Border 클래스를 상속받는 SideBorder, FullBorder 클래스를 주어진 요구사항에 맞도록 구현하시오.

O 클래스다이어그램



다음은 제공되는 BorderMain클래스를 제외한 Display, StringDisplay, Border, SideBorder, FullBorder 클래스의 구현에 필요한 요구사항들입니다. (25점 - 세부 점수로 분할)

A. Display 클래스 구현 (2점)

- 추상 클래스로 작성합니다.
- getColumns() 와 getRows() 메소드는 각각 문자 수와 세로의 줄의 수를 얻기 위한 메소드입니다. 이것은 추상 메소드로 하위 클래스에서 적절하게 구현되어야 합니다.
- getRowText() 메소드는 지정한 줄의 문자열을 얻기 위한 추상 메소드 입니다.
- show() 메소드는 모든 줄을 표시하는 메소드입니다. 이 안에서는 getRows() 메소드로 표현해야하는 문자열의 줄 수를 얻고 getRowText() 메소드로 표시해야 하는 문자열을 얻은 후 for 루프를 사용해서 모든 줄을 화면에 출력합니다.
  또한 show() 메소드는 자식 클래스에서 재정의(overriding)하지 못하도록 final로 작성합니다.

B. StringDisplay 클래스 구현 (5점)

- Display클래스를 상속받는 클래스로서 Display 클래스의 추상메소드를 모두 구현합니다.
- 멤버변수로 화면에 출력할 문자열을 가지고 있습니다.
- 생성자 메소드를 통해서 화면에 출력할 문자열을 초기화 합니다.
- getColumns() 메소드는 string.getBytes().length 구문을 통해서 출력할 문자열이 몇 개의 문자로 구성되어 있는지 리턴하는 메소드입니다.
- getRows() 메소드는 무조건 1을 return 하도록 구현합니다.
- getRowText(int row) 메소드는 매개변수로 받은 row 값이 0이면 멤버변수인 string 문자열을 리턴하고 아닌 경우에는 null을 리턴하도록 구현합니다.

C. Border 클래스 (3점)

- Border 클래스는 장식을 나타내는 추상클래스로서 StringDisplay 클래스와 마찬가지로 Display 클래스의 자식 클래스로 작성합니다.
- 멤버변수로 Display 클래스 타입의 display 변수를 가지고 있습니다.
- 생성자 메소드를 통해 display 변수의 값을 초기화 합니다.

D. SideBorder 클래스 (5점)

- Border 클래스를 상속받으며 문자열의 양 Side를 특정 문자(char)로 장식하는 클래스입니다.
- char 타입의 장식 문자를 멤버변수로 가지고 있습니다.
- 매개변수로 Display 타입의 변수와 char 타입의 문자를 받아서 부모클래스인 Border 클래스의 Display 변수를 초기화하고 맴버변수로 선언된 borderChar를 초기화 하는 생성자 메소드를 가지고 있습니다. SideBorder(Display display, char ch)
- getColumns() 메소드는 장식할 내용물 양쪽에 장식문자를 더한 숫자를 리턴합니다.
  return 1 + display.getColumns() + 1; 혹은 return display.getColumns() + 2;
- getRows() 메소드는 장식할 문자열 줄 수와 같은 줄 수를 리턴합니다.
- getRowText(int row) 메소드는 장식해야 할 문자열의 줄 번호를 매개변수로 받아 해당 줄의 양 Side에 장식할 장식문자(borderChar)를 붙여 리턴합니다.

E. FullBorder 클래스 구현 (10점)

- Border 클래스를 상속받으며 장식 문자열의 테두리를 장식하는 클래스입니다.
  장식할 내용물의 첫번째와 마지막 줄에 장식할 내용물의 단어수만큼”----------”로 장식을 하고 나머지 줄에는 ”|”로 장식합니다.
- 매개변수로 Display 타입의 display 변수를 받아 부모의 생성자 메소드를 호출하여 부모의 display 변수값을 초기화 하는 생성자 메소드가 있습니다.
- getColumns() 메소드는 장식할 내용물 양쪽에 장식문자를 더한 숫자를 리턴합니다.
- getRows() 메소드는 장식할 문자열 줄 수 위 아래에 장식할 문자 수를 1씩 더하는 메소드로 구현합니다.
- getRowText(int row) 메소드는 장식할 내용물의 상단과 하단에 ”----------”로 장식하고 나머지 줄에는 ”|”로 장식하는 메소드입니다. 매개변수로 넘어온 숫자가 0이거나 display.getRows() + 1 과 같은 값인 경우 장식할 내용의 첫번째와 마지막 라인을 의미하므로 makeLine() 메소드를 통해 ”----------”로 장식을 추가합니다.
- makeLine(char ch, int count) 메소드는 매개변수로 지정된 문자 ch를 count 만큼(즉, 장식할 내용물의 column 수만큼) 반복해서 상하 장식 문자열인 ”----------”을 리턴하는 메소드입니다.

[BorderMain 실행결과]

```
1. 문자열을 출력합니다.
Hello, world.

2. '#'으로 Side를 장식합니다.
#Hello, world.#

3. '&'으로 Side를 장식합니다.
&#Hello Gurum#&

4. 모든 테두리를 장식합니다.
+----------------+
|&#Hello Gurum#&|
+----------------+
```

<<참고 및 제한 사항>>

- BoarderMain.java 외 별도로 제시되는 소스 파일이 없습니다.
- 이번 문제에서 작성해야 할 소스 파일은 총 5개입니다. (Display.java, StringDisplay.java, Border.java, SideBorder.java, FullBorder.java)

문제4. StudentManagerTest.java 파일을 수행하였을 때 아래와 같은 화면출력 결과가 나오도록, Student.java 파일과 StudentManager.java 파일을 완성하십시오.

I. (10점) Student.java

- 학생이름을 매개변수로 하는 생성자를 구현한다.
- 학생이름, 중간점수, 기말점수, 과제점수를 매개변수로 하는 생성자를 구현한다.
- 학생의 등급을 설정하는 calcGrade() 메쏘드를 구현한다. 등급에 대한 판정기준은 아래와 같다.
  - 중간, 기말, 숙제 점수를 각각 40%, 40%, 20%로 반영
  - 100점 이하 ~ 90점 이상 : A
  - 90점 미만 ~ 80점 이상: B
  - 80점 미만 ~ 70점 이상: C
  - 70점 미만 ~ 60점 이상: D
  - 60점 미만 : F

II. (15점) StudentManager.java

- 학생이름(name)을 인자로 받아서, 동일한 이름에 해당하는 학생(Student)을 리턴하는 getStudent() 메쏘드를 구현한다.
- 학생의 등급(grade)을 인자로 받아서 검색된 학생 목록을 ArrayList에 담아서 리턴하는 search() 메쏘드를 구현한다.

[StudentManagerTest 클래스 실행 결과]

```
getStudent() Test --------------
하대치, C
search() Test --------------
염상진, B
소화, B
심재모, B
```

문제5. Keyboard로부터 입력 받은 16진수를 10진수로 변환하는 Hex2Decimal 클래스를 작성하십시오.

다음은 Hex2Decimal 클래스의 구현에 필요한 요구사항들입니다.

A. 16진수 문자열값을 Keyboard로부터 받을 수 있어야 합니다. (3점)

B. 입력 받은 문자열이 ' quit' 인 경우, " Bye !!!" 메시지를 보여주고 종료합니다. (2점)

C. 입력 받은 문자열이 16진수가 아닌 경우, " Invalid hex !!!" 메시지를 보여주고 다시 입력 받습니다. (5점)

D. 아래의 예와 같이 16진수 문자열을 10진수로 변환합니다. (15점)

예) 16진수 : ABCD
= 10 * 16^3 + 11 * 16^2 + 12 * 16^1 + 13 * 16^0
= 40960 + 2816 + 192 + 13 = 43981

예) 16진수 : 325
= 3 * 16^2 + 2 * 16^1 + 5 * 16^0
= 768 + 32 + 5 = 805

주) ^ 은 거듭제곱을 의미합니다. 즉, 16^2 = 16 * 16 = 256

[Hex2Decimal 클래스 실행 예시]

```
Enter the hex value ('quit' for exit) : ABCD
hex input : ABCD
decimal output : 43981
==========================================
Enter the hex value ('quit' for exit) : 325
hex input : 325
decimal output : 805
==========================================
Enter the hex value ('quit' for exit) : 3H3
hex input : 3H3
Invalid hex !!!
==========================================
Enter the hex value ('quit' for exit) : quit
Bye !!!
```

<<참고 및 제한 사항>>

- Math.pow(double, double) 메소드를 사용하여 거듭제곱 계산을 할 수 있습니다.
- 문자는 모두 대문자 알파벳이 입력된다고 가정합니다.

문제6. Keyboard로부터 구매하려는 상품코드와 상품팩의 가격, 구매하려는 상품팩의 개수 등을 입력받아서 지불해야 할 금액을 계산하는 Purchase 프로그램을 작성하십시오. (25점)

1) 상품코드(productCode), 상품 가격(productCost), 구매할 상품팩의 개수(numOfPack)와 하나의 상품팩에 포함된 상품의 개수(numOfProduct)를 멤버 변수로 갖는 Product 클래스를 구현합니다. 각 멤버 변수에 대하여 getter, setter 메소드를 구현합니다.(5점)

2) 다음은 Purchase 클래스의 구현에 필요한 요구사항들입니다. (15점)

A. 아래와 같이 상품코드와 상품팩의 가격, 구매하려는 상품팩의 개수 등을 Keyboard로부터 입력받을 수 있어야 합니다. (5점)

구매하려는 상품코드를 선택하세요. [1:사과,2:오렌지,3:포도,4:딸기] ? 4
상품팩에 포함된 상품의 개수, 상품팩의 가격을 순서대로 입력하세요 (구분자:,) ?
2,4000
구매하려는 상품팩의 개수를 입력하세요 ? 3

B. 입력받은 값을 Product 클래스에 담아서 반환하는 getProduct 메소드를 작성해야 합니다. (5점)

```
public Product getProduct();
```

C. Product 클래스를 받아서 해당 상품팩의 재고량을 체크하고, 구매 가능한 경우 지불해야 할 총 금액을 계산하여 반환하는 calcTotalPrice 메소드를 작성해야 합니다. 재고량이 부족할 경우에는 -1을 반환합니다. (5점)

```
public double calcTotalPrice(Product product);
```

D. 아래와 같이 결과를 출력해야 합니다. (2점)

3개의 딸기 상품팩을 구매하셨습니다.
딸기 한 개의 가격은 2000원이며, 지불해야 할 총 금액은 12000원입니다.

해당 상품팩에 대한 재고량이 부족할 경우 결과는 다음과 같이 출력해야 합니다. (3점)

재고가 없어 3개의 딸기 상품팩에 대한 구매가 불가합니다. 죄송합니다.

<<처리 결과>>

구매하려는 상품코드를 선택하세요. [1:사과,2:오렌지,3:포도,4:딸기] ? **4**
상품팩에 포함된 상품의 개수, 상품팩의 가격을 순서대로 입력하세요 (구분자:,) ? **2, 4000**
구매하려는 상품팩의 개수를 입력하세요 ? **3**
3개의 딸기 상품팩을 구매하셨습니다.
딸기 한 개의 가격은 2000원이며, 지불해야 할 총 금액은 12000원입니다.

<<참고 및 제한 사항>>

- 상품별 상품명을 정의한 nameOfProduct 배열을 사용합니다.

```
private static String nameOfProduct[] = { "사과" ," 오렌지" ," 포도" ," 딸기" };
```

- 상품별 재고 수량을 정의한 totalStockOfPack 배열을 사용합니다.

```
private static int totalStockOfPack[] = {3, 5, 20, 1};
```

- 상품 한 개의 가격 : 상품팩의 가격 / 상품팩에 포함된 상품의 개수, 나누어 떨어지지 않는

경우 Math.round() 를 이용하여 반올림합니다.

- 지불 금액 : 상품팩 한개의 가격 * 구매 상품팩의 개수
- **Product 클래스는 새로 구현**하고, **Purchase 클래스는 주어진 소스의 주석 부분을 채워**서 완성합니다.