Java 練習問題. 13 ファイル入出力

1．ファイルの種類

ファイルには大きく分けてテキストファイルとバイナリファイルの 2 種類があります。
両者の見分け方は「人間が読めるか読めないか」です。

テキストファイルは、人間が分かるようなルールに基づいて記録されているのに対し、バイナリファイルは意味不明なデータの羅列にしか見えません。
Java ではもちろん両者のファイルを扱うことができます。

2．ストリーム

ファイルの読み込みや書き出しをするときのデータの流れをストリームと言います。
Java では、ファイルの読み書きにストリーム専用のクラスから生成したオブジェクトを介して、ファイルとデータのやり取りを行います。
ストリームは扱うデータによって、文字ストリームとバイトストリームに分かれます。

ストリーム	機能
文字ストリーム	16 ビットの Unicode 文字のデータを扱います。
バイトストリーム	8 ビットのデータを扱います。

3．ストリームを扱うクラス

	バイトストリームクラス	文字ストリームクラス
基本クラス	FileInputStream FileOutputStream	FileReader FileWriter

Java には、用途に応じて様々なタイプのストリームクラスが java.io パッケージに用意されています(上表参照)。これらのクラスを利用するには次の文を最初に記述します。

import java.io.*;

４．文字の読み込みの基本

テキストファイルから読み込むには、以下の手順で行います。

① ファイルを開く

テキストファイルを読み込むには、FileReader クラスのオブジェクトを使います。オブジェクトを生成すると、ファイルを開くことができます。

② データを読み込む

データを読み込むには read メソッドを使用します。read メソッドは読み込んだ文字を int 型の整数で返します。読み込むデータがなくなると-1 を返すので、ファイルを最後まで読み込むには read メソッドが-1 を返すまで読み込みを繰り返します。

③ ファイルを閉じる

ファイルを閉じるには、close メソッドを使用します。

この手順でプログラムを作ると、Sample1 のようになります。Sample1 はあらかじめ用意しておいた file1.txt を読み込み、その内容を表示します。

Sample1.java

```
import java.io.*;

public class Sample1 {
   public static void main(String[] args) {
      try {
         FileReader fr = new FileReader("file1.txt"); //①ファイルを開く
         int c ;
         String s = new String();
         while((c = fr.read()) != -1) { //②データを読み込む
            s += (char)c;
         }
         System.out.println(s);
         fr.close(); //③ファイルを閉じる
      }catch(FileNotFoundException e) {
         System.out.println("ファイルが存在しません。");
      }catch(IOException e) {
         e.printStackTrace();
      }
   }
}
```

実行結果

```
あいうえお
かきくけこ
```

５．文字の書き出しの基本

“Hello”という文字列を file2.txt に書き出す手順を説明します。

①　ファイルを開く

テキストファイルを書き出すには、FileWriter クラスのオブジェクトを使います。
引数に指定したファイルが新規に作成されます。同じ名前のファイルが既にあった場合には上書きします。

②　データを書き出す

データを書き出すには、write メソッドを使用します。write メソッドは、引数として与えたデータをファイルに書き出します。

③　ファイルを閉じる

ファイルを閉じるには、close メソッドを使用します。

この手順でプログラムを作ると、Sample2 のようになります。

Sample2.java

```
import java.io.*;

public class Sample2 {
  public static void main(String[] args) {
    try {
      FileWriter fr = new FileWriter("file2.txt"); //①データを開く
      fr.write("Hello¥r¥n"); //②データを書き出す
      fr.close(); //③ファイルを閉じる
    }catch(Exception e) {
       e.printStackTrace();
    }
  }
}
```

６．バイナリファイルの読み込み/書き出し

バイナリファイルの読み込みには FileInputStream クラス、書き出しには FileOutputStream クラスを使います。テキストファイルの場合とクラスが違うだけで仕組みはほとんど同じなので、ここでは割愛します。

７．1 行分のテキストを読み込む

テキストファイルの 1 行分のデータを 1 度に読み込むには BufferedReader クラスを使います。BufferedReader クラスは、文字、配列、行をバッファリングすることによって FileReader クラスよりも効率良く読み込むことができます。

file3.txt から 1 行分のデータを読み込む手順を説明します。

① ファイルを開く

まずはテキストファイルを読み込むための、FileReader クラスのオブジェクトを生成し、ファイルを開きます。

② BufferedReader クラスを生成する

BufferedReader クラスのオブジェクトを生成し、引数に FileReader のオブジェクトを渡します。これによって、FileReader から読み込まれたデータが一時的にバッファに蓄積されるようなストリームを作ることができます。

③ データを読み込む

データを読み込むには、BufferedReader が持つ readLine メソッドを使うことができます。readLine メソッドはファイルから 1 行読み込み、String 型で返します。ファイルの終端に達すると null を返します。

④ ファイルを閉じる

ファイルを閉じるには、close メソッドを使用します。

この手順でプログラムを作ると、Sample3 のようになります。Sample3 はあらかじめ用意しておいた file3.txt を読み込み、その内容を表示します。

Sample3.java

```
import java.io.*;

public class Sample3 {
public static void main(String[] args) {
   try {
      FileReader fr = new FileReader("file3.txt"); //①ファイルをひらく
      BufferedReader br = new BufferedReader(fr); //②BufferedReader クラスを生成
      String str;
      while((str = br.readLine()) != null) { //③データを読み込む
            System.out.println(str);
      }
      br.close(); //④ファイルを閉じる
      }catch(FileNotFoundException e) {
         System.out.println("ファイルが存在しません");
      }catch(IOException e) {
         e.printStackTrace();
      }
   }
}
```

実行結果

```
あいうえおかきくけこ
さしすせそたちつてと
```

設問1.　exdata.txt というファイルに次の文字列を書き込むプログラム ExportData.java を作成しなさい。プログラム実行後、exdata.txt に正しく文字列が書き込まれていることを確認しなさい。

```
雨にも負けず
風にも負けず
```

※ヒント
文字列を 1 文字ずつ分解して FileWriter で書き込む方法もありますが、1 行分をまとめて書き込む BufferedWriter を使うこともできます。
BufferedWriter は次のように使います。

```
…
FileWriter fw = new FileWriter("exdata.txt");
BufferedWriter bw = new BufferedWriter(fw);
bw.write("書き込む文字列¥n");
bw.close();
…
```

例外処理の記述も忘れずに行ってください。

設問2.　練習問題 1 で作成した exdata.txt から文字列を読み込み、内容を表示するプログラム ImportData.java を作成しなさい。

※ヒント
1 行分の文字列を読み込むには BufferedReader を使うと簡単です。
例外処理の記述も忘れずに行ってください。