1-7 プログラミング演習

月　　日

※動画説明欄から提供プログラムをダウンロード！　→

Lesson のゴール

① FileReader(Writer)クラスや BufferedReader(Writer)クラスを用いた入出力処理のプログラムを書けるようになる

② プログラムの処理時間を計測する方法を理解する

③ try-with-resources 文を書けるようになる

［キーワード］

ストリーム、文字ストリーム、フィルタストリーム、
標準出力、標準エラー出力、try-with-resources 文

▶ 問題

1

文字ストリームを使ってテキストファイルをコピーするプログラムを作ります。既存のテキストファイル（original.txt）から 1 行の文字列を読み取り、その文字列を別のファイル（copy1.txt）に 100,000 回書き込みます。また、入力と出力の処理時間を計測して標準出力します。

CharStreamTime.java にコードを追加して、プログラムを完成させてください。なお、処理時間の計測には System クラスの currentTimeMillis()メソッド（下記 API 仕様）を用います。

currentTimeMillis

```
public static long currentTimeMillis()
```

ミリ秒で表される現在の時間を返します。 戻り値の時間単位はミリ秒ですが、値の粒度は基本となるオペレーティング・システムによって異なり、単位がより大きくなる場合があります。 たとえば、多くのオペレーティング・システムでは、時間を10ミリ秒の単位で計測します。

「コンピュータ時間」と協定世界時(UTC)との間に発生する微妙な相違については、クラスDateの説明を参照してください。

戻り値:

ミリ秒で測定した、現在時刻と協定世界時のUTC 1970年1月1日深夜零時との差。

original.txt

```
1  It always seems impossible until it's done.
```

CharStreamTime.java

```
1  /* ここを埋める */
2
3  public class CharStreamTime{
4    public static void main(String[] args){
5      FileReader fr = null;
6      FileWriter fw = null;
7      try {
8        fr = /* ここを埋める */
9        fw = /* ここを埋める */
10
11       int data;
12       String msg = "";
13
14       long t1 = System.currentTimeMillis();
15       while( (data = /* ここを埋める */) != -1) {
16         msg = msg + (char)data;
17       }
18
19       long t2 = System.currentTimeMillis();
20       for(int i = 0; i < 100000; i++){
21         /* ここを埋める */
22         fw.write("\n");
23       }
24
25       long t3 = System.currentTimeMillis();
26       System.out.println("入力 : " + (t2 - t1) + "ミリ秒");
27       System.out.println("出力 : " + (t3 - t2) + "ミリ秒");
28       System.out.println("合計 : " + (t3 - t1) + "ミリ秒");
29
```

```
30        } catch(IOException e) {
31          System.err.println("エラーが発生しました");
32        } finally {
33          try {
34            if(fr != null) /* ここを埋める */
35            if(fw != null) /* ここを埋める */
36          } catch(IOException e) {
37            System.err.println("エラーが発生しました");
38         }
39        }
40      }
41  }
```

実行例

```
>java CharStreamTime
入力：          ミリ秒
出力：    ?     ミリ秒
合計：          ミリ秒
```

※実際に実行して確認してみましょう！

copy1.txt

```
  1    It always seems impossible until it's done.
  2    It always seems impossible until it's done.
  :                          :
  :                          :
100000 It always seems impossible until it's done.
```

2

フィルタストリームを使って問題 1 と同じ動作をするプログラムを作ります。FilterStream Time.java にコードを追加して、プログラムを完成させてください。

FilterStreamTime.java

```
1  import java.io.FileReader;
2  import java.io.FileWriter;
3  /* ここを埋める */
4  import java.io.IOException;
5
6  public class FilterStreamTime{
7     public static void main(String[] args){
8        FileReader fr = null;
9        FileWriter fw = null;
10       BufferedReader br = null;
11       BufferedWriter bw = null;
12       try {
13          fr = new FileReader("original.txt");
14          br = /* ここを埋める */
15          fw = new FileWriter("copy2.txt");
16          bw = /* ここを埋める */
17
18          long t1 = System.currentTimeMillis();
19          String msg = /* ここを埋める */
20
21          long t2 = System.currentTimeMillis();
22          for(int i = 0; i < 100000; i++){
23             /* ここを埋める */
24             bw.newLine();
25          }
26          /* ここを埋める */
27
```

```
28          long t3 = System.currentTimeMillis();
29          System.out.println("入力 : " + (t2 - t1) + "ミリ秒");
30          System.out.println("出力 : " + (t3 - t2) + "ミリ秒");
31          System.out.println("合計 : " + (t3 - t1) + "ミリ秒");
32
33        } catch(IOException e) {
34          System.err.println("エラーが発生しました");
35        } finally {
36          try {
37            if(br != null) /* ここを埋める */
38            if(bw != null) /* ここを埋める */
39          } catch(IOException e) {
40            System.err.println("エラーが発生しました");
41          }
42        }
43      }
44  }
```

実行例

```
>java FilterStreamTime
入力：          ミリ秒
出力：    ?     ミリ秒
合計：          ミリ秒
```

※実際に実行して確認してみましょう！

copy2.txt

```
     1  It always seems impossible until it's done.
     2  It always seems impossible until it's done.
     :                    :
     :                    :
100000  It always seems impossible until it's done.
```

3

問題 2 と同じ動作をするプログラムを、try-with-resources 文（Java SE 7 仕様）を使って作ります。TWRTime.java にコードを追加して、プログラムを完成させてください。

TWRTime.java

```
1  import java.io.FileReader;
2  import java.io.FileWriter;
3  import java.io.BufferedReader;
4  import java.io.BufferedWriter;
5  import java.io.IOException;
6
7  public class TWRTime{
8     public static void main(String[] args){
9        try( /* ここを埋める */ ) {
10
11          long t1 = System.currentTimeMillis();
12          String msg = br.readLine();
13
14          long t2 = System.currentTimeMillis();
15          for(int i = 0; i < 100000; i++){
16             bw.write(msg);
17             bw.newLine();
18          }
19          bw.flush();
20
21          long t3 = System.currentTimeMillis();
22          System.out.println("入力 : " + (t2 - t1) + "ミリ秒");
23          System.out.println("出力 : " + (t3 - t2) + "ミリ秒");
24          System.out.println("合計 : " + (t3 - t1) + "ミリ秒");
25
26       } catch(IOException e) {
27          System.err.println("エラーが発生しました");
```

```
28        }
29      }
30    }
```

実行例

```
>java TWRTime
入力：          ミリ秒
出力：    ?     ミリ秒
合計：          ミリ秒
```

※実際に実行して確認してみましょう！

copy3.txt

```
     1    It always seems impossible until it's done.
     2    It always seems impossible until it's done.
     :                    :
     :                    :
100000    It always seems impossible until it's done.
```

＼フリーラーニング（無料で学べる場）を広げたい！／
チャンネル登録や拡散よろしくお願いします！

せかチャン