プログラミングIII

黒瀬　浩

kurose@neptune.kanazawa-it.ac.jp

OH: 講義の前後，火4限21・405

居室　67・121

レポート3補足　メイン側

main側のスレッド作成

　同一ループ内でnew(スレッド作成)とstart(runメソッド起動）をしている人がいましたが，それでは，最後のスレッドを作成し終える前に最初のスレッドが終わってしまうかもしれない。

並列動作にするためには作成と起動は別ループで行う必要がある。

```
MyThread[] t = new MyThread[n];
for(int i=0; i<n; i++){ // スレッド作成
    t[i] = new MyThread();
}
for(int i=0; i<n; i++){ // スレッド起動
    t[i].start();
}
for(int i=0; i<n; i++){ // この処理は終了順では無いが良しとする
    try{
        t[i].join();
    }
    catch(InterruptedException e){
        System.out.println(e);
    }
}
```

レポート3補足　スレッド側

スレッドID, 開始時刻を取得後printし，sleep後終了時刻をprintすると最初のprint 後にsleepでスレッド切替が発生し，別のスレッドが動く可能性が高く，各スレッドの出力が混ざってしまう。

開始時刻，スレッドIDは変数に保管しておき最後の1つのprintで出力する(print中に他のprintが混じらないのかという議論があるがstream処理をまだ説明していないのでここで留めておく）

出力の整形をしていない人がいましたが以下の様に数字は右揃え，文字は左揃え（桁数に-をつける）すれば良い

```
System.out.printf( "%5d start %-10s wait %3d end %-10s¥n",
                   id, d1.format(df), w, d2.format(df) );
}  // idはスレッドid, wは待ち秒数，d1は開始時刻，d2は終了時刻
```

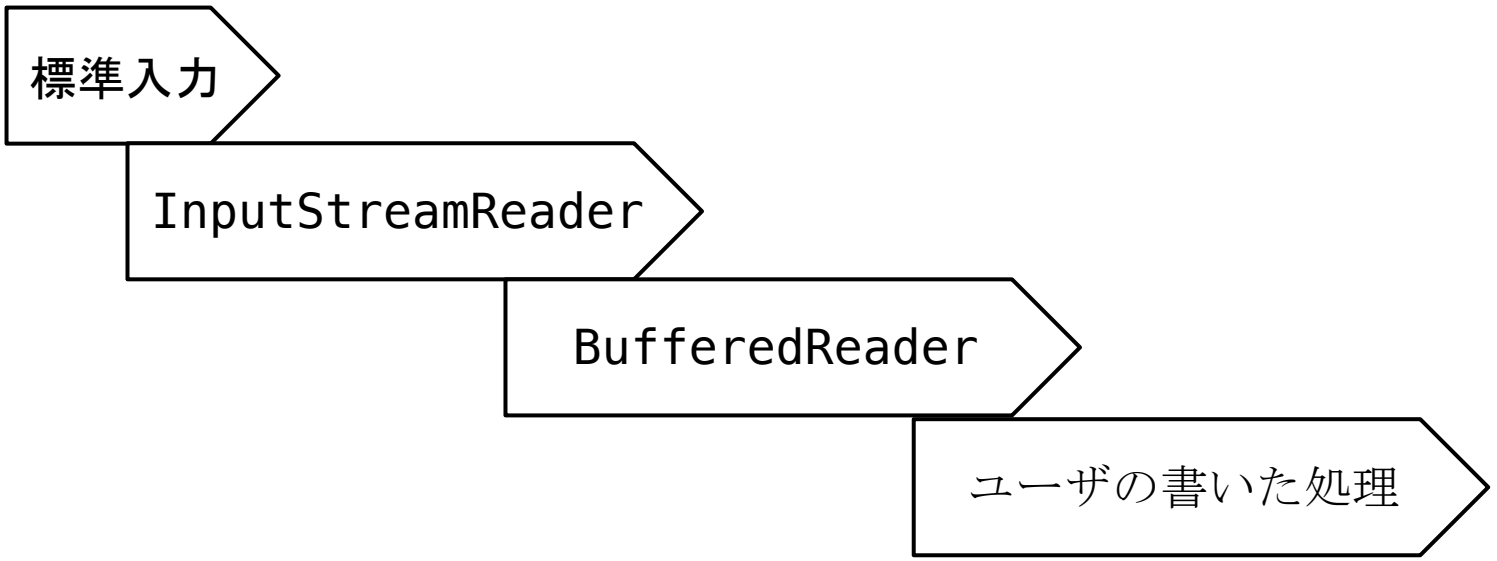
sleepの引数の単位はミリ秒

7章　ファイル入出力

文字列データ　Unicode文字として処理(テキストファイルなど印字可能)
バイナリデータ 1バイト単位で処理(プログラムファイル，圧縮データなど)

標準入力と標準出力
DOSプロンプトでは，画面のキー入力と出力
IDEでは，　コンソール画面相当のキー入力と出力部分

Javaでのデータの流れ



Streamからの入力

```
// 標準入力から1行読む
InputStreamReader in = new InputStreamReader( System.in );
BufferedReader reader = new BufferedReader( in );
String line = reader.readLine();


// 入門編で出てきた例 標準入力から数値を読む
Scanner in = new Scanner( System.in );
int num = in.nextInt();
```

Scannerクラスを使用した方がお手軽だが,
ファイルやネットワークからまとめてデータを受けたい場合は
Streamを使う

Streamは，データが入っていれば読み，無ければ待つ
(End of Fileなど，これ以上読めない場合は例外発生)

文字列と数値の変換

Streamから読んだデータは，Unicode文字列かバイト列
整数に変換するには

```
int i = Integer.parseInt( line );                    // 整数化
double d = Double.parseDouble( line );       // 実数化
```

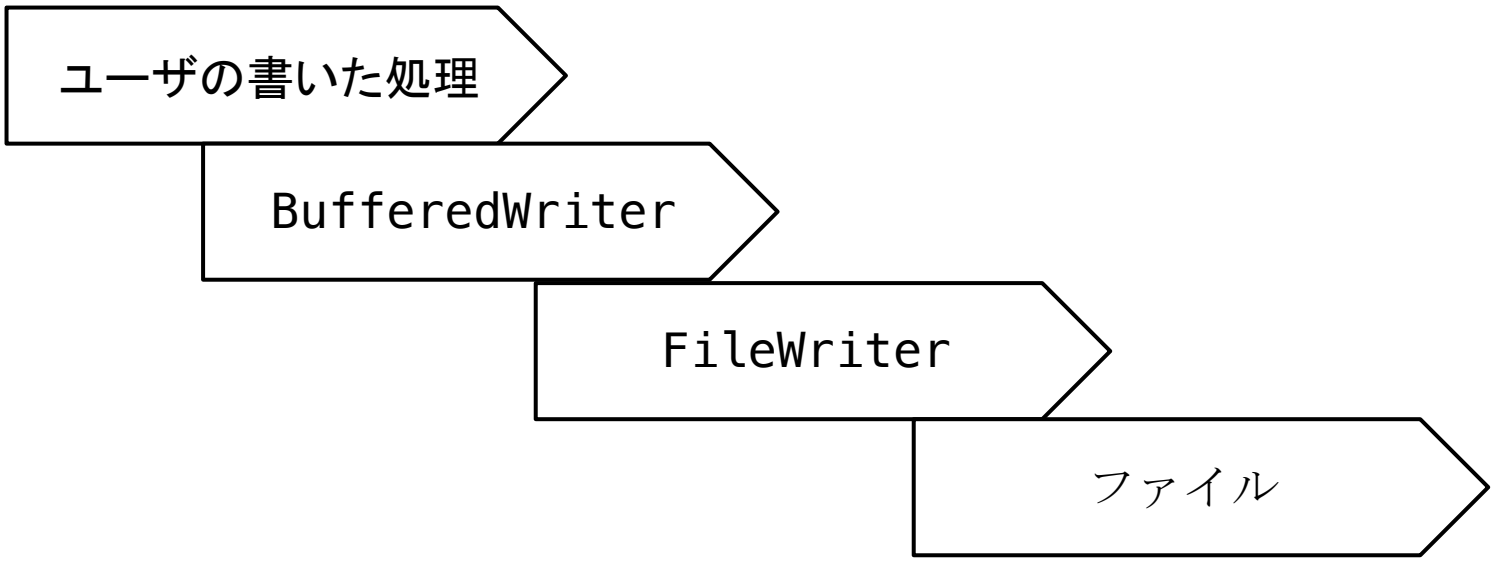
これは入門編でも使用した

前のページで　line は readLine()メソッドで読んでいるので1行分入る
1行に1つの数値なら上記で良いが，1行に複数の数値がある場合は，
前もって変換する部分を分割する必要がある
split()メソッドなどでカンマ区切り，スペース区切りを処理できる

ファイルへの出力

```
File file = new File("c:/java/test.txt");
FileWriter fw = new FileWriter( file );
BufferedWriter bw = new BufferedWriter( fw );
bw.write( 文字列 );
bw.newLine();
bw.close();
```

ファイルは標準出力では無いのでopenが必要 1行目



ストリームの連結

ストリームはデータ列を順に処理するので，複数のストリームを連結可能

```
File file = new File("c:/java/test.txt");
FileWriter fw = new FileWriter( file );
BufferedWriter bw = new BufferedWriter( fw );
PrintWriter pw = new PrintWriter( bw );
pw.println( 文字列 );
pw.close();
```

それぞれのクラスで使用できるメソッドは決められている

println()メソッドはPrintWriterクラスで使える

標準出力への出力System.outは，open(fileオブジェクト作成)やPrintWriterの準備がされていたので，いきなりprintln()が使えた

ファイルからの入力

```
File file = new File("C:/java/test.txt);
FileReader fr = new FileReader( file );
BufferedReader br = new BufferedReader( fr );
String s;
while( (s = br.readLine() ) != null ) {
        処理
}
// ファイルが終了(End of File)かエラー
br.close();
```

教科書ではさらに`try catch`で`IOException`の処理をしている

シリアライゼーション

数値の羅列であればテキストファイルに連続で書けば良いが
配列は，配列長も記録しないと復元できない
クラスのフィールドの値を保存したい場合に，数値，文字列，コレクションなどが混在する
Javaオブジェクトを保存，復元する場合にシリアライゼーションが必要

保存

```
FileOutputStream fs = new FileOutputStream( ファイルパス );
ObjectOutputStream os = new ObjectOutputStream( fs );
os.writeObject( オブジェクト );
os.close();
```

復元

```
FileInputStream fs = new FileInputStream( ファイルパス );
ObjectInputStream os = new ObjectInputStream( fs );
戻す型　変数 = (戻す型)os.readObject();　// キャスト
```

ファイルとフォルダの操作

ファイル

```
File 変数 = new File( ファイルパス );
変数.exists();                          // 存在する場合True
変数.delete();                          // 削除できればTrue
変数.renameTo( 新しいファイル名 );      // リネームできればTrue
```

フォルダ(ディレクトリ)

```
File 変数 = new File( フォルダパス );
変数.list();                            // ファイル一覧をString[]で戻す

File 変数1 = new File( ファイルパス );
変数1.mkdir();                          // ディレクトリを作る
```

レポート4 配点4点

7章 章末問題7.3を完成させ以下の指示事項に対応させる

ファイルの各行に行番号をつけて別ファイルへ出力する
行番号は1から始め4桁頭に0を詰めコロンとスペースを2つ入れる
例　0001: ......

その行にブロック開始({)があれば:の後ろに +とスペース1個を入れる
その行にブロック終了(})があれば:の後ろに – とスペース1個を入れる
その行にブロック開始と終了があれば:の後ろに*とスペース1個を入れる
例 0010:+ ... main(){ ...↵
   0011:- }↵
   0012:* ... method1(){ return 1; }↵

入力ファイルは，このプログラムのJavaソースファイル
ヒント：0001のように頭の空白を0にする出力はString.format() を使う

期限 次回開始時
A4 1枚(両面可能) クラス名列氏名を記入
出力ファイルの内容を印刷する