情報システムプロジェクトI

第２回　FileScanner.java

～ファイル入力、文字切り出し機能を持つ
プログラムの作成～

K23言語処理系を構成する主なプログラム

FileScanner.java

ファイル入力機能、文字列から文字を切り出す機能

LexicalAnalizer.java

切り出された文字から単語（トークン）を組み立て、分類する機能

Kc.java

トークンを組み合わせ、式や構造として認識し、アセンブラの命令を次々と組み立てる機能

今日作成するプログラム

FileScanner.java

ファイル入力機能、文字列から文字を切り出す機能

実行結果

```
main() {
    int i, n=0, m=1, s, tmp, is_sorted=1, SIZE=20, data[20];
    int message[]= {'s','o','r','t'};

    outputchar('?');
```

bsort.kの内容が、改行も含め正しく
表示されていればよい

try-with-resourcesについて

教科書p. 210 ソースコード10.1

```
9    try (BufferedReader reader
                        = Files.newBufferedReader(path)){
10       String line;
11       while ((line = reader.readLine()) != null ) {
12            System.out.printf("%s¥n", line);
13       }
14   } catch (IOException e) {
```

reader がこのtry文の中でしか使えない

配布したFileScanner の雛形

```
10   private BufferedReader sourceFile;

25    FileScanner (String sourceFileName) {
26        Path path = Paths.get (sourceFileName);
28          try {
29            sourceFile = Files.newBufferedReader (path);
30          } catch (IOException err_mes) {
```

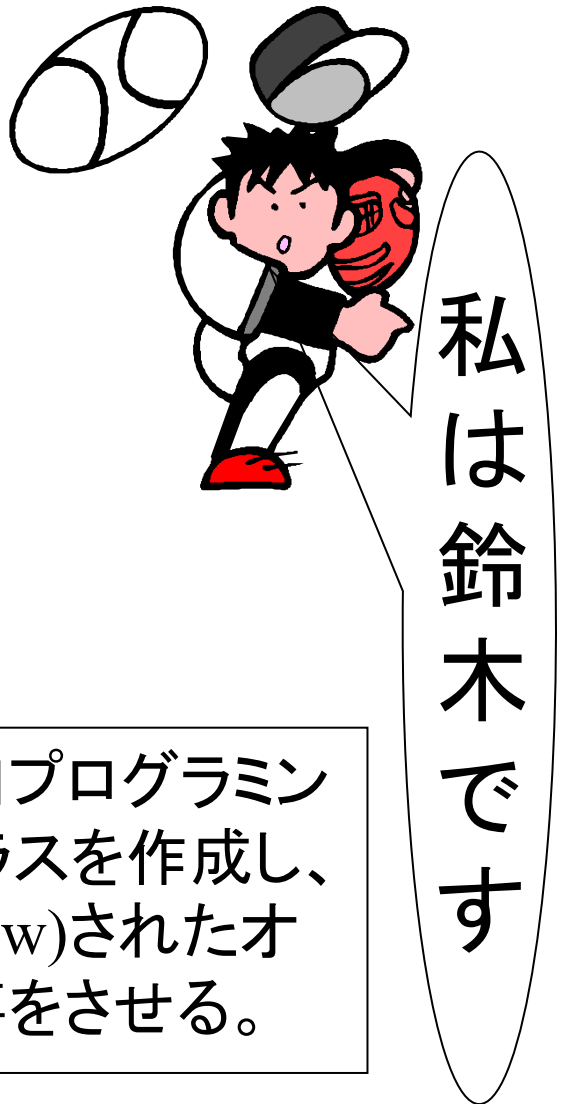
オブジェクト指向プログラミング

クラス　学生

- ●性別
- ●学年
- ●名前
- ●クラブ
- ●できること
  聞かれたら
  名前を答える

→ オブジェクト

＝男性
＝2
＝鈴木
＝野球



オブジェクト指向プログラミングでは、まずクラスを作成し、そこから生成(new)されたオブジェクトに仕事をさせる。

クラス FileScanner

- sourceFile
- line
- lineNumber
- columnNumber
- currentCharacter
- nextCharacter
- できること
  - closeFile()
  - readNextLine()
  - lookAhead()
  - getLine()
  - nextChar()
  - scanAt()

→ オブジェクト

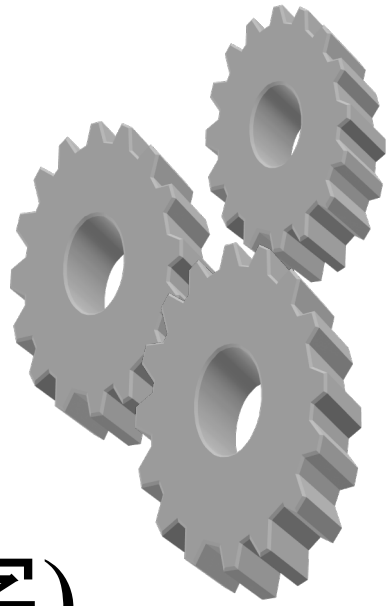
= ソースファイル

=

= 0

= -1

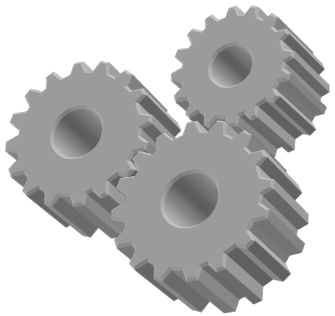
= '\n'　(改行文字)



今回作成するクラス(左)と、
コンストラクタで上記の値をフィールドに設定し生成されるオブジェクト(右)。指導書の pp. 9～11 に詳しい仕様がある。

main

{



コンストラクタでオブジェクトを生成

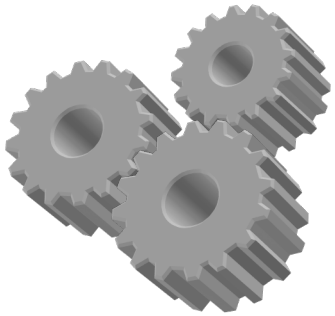
生成された直後のオブジェクトの状態

- sourceFile = bsort.kファイルを参照
- line
- lineNumberum = 0
- columnNumber = -1
- currentCharacter
- nextCharacter = '\n' (改行文字)

//nextChar(); (問2.6では、ここのコメントアウトをはずす)

main(問題 2.5)

{

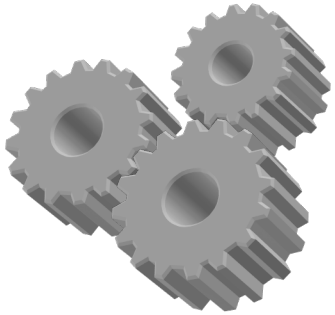


readNextLine() と getLine() を用いて、行単位で bsort.kの内容を表示する

- sourceFile = bsort.kファイルを参照
- line
- lineNumber = 0
- columnNumber = -1
- currentCharacter =
- nextCharacter = '\n'

注: 問題2.5のFileScanner.javaは提出しない

main(問題 2.5)
{



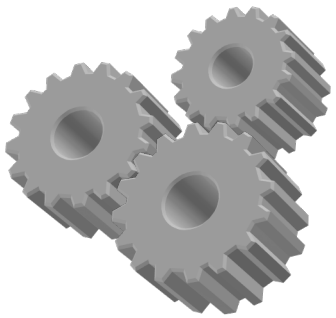
readNextLine() と getLine() を用いて、行単位で bsort.kの内容を表示する

- sourceFile = bsort.kファイルを参照
- line = "main() {\n"
- lineNumberum = 1
- columnNumber = -1
- currentCharacter =
- nextCharacter = '\n'

注: 問題2.5のFileScanner.javaは提出しない

main(問題 2.6)

{



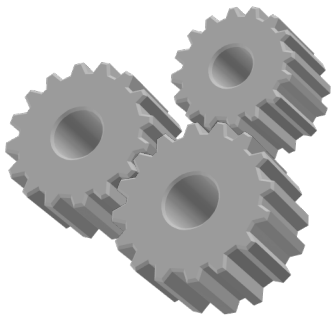
行ではなく、文字単位で
bsort.kの内容を表示する

- sourceFile ＝ bsort.kファイルを参照
- line
- lineNumberum ＝ 0
- columnNumber ＝ -1
- currentCharacter ＝
- nextCharacter ＝ '\n'

注: 問題2.6のFileScanner.javaも提出しない

main(問題 2.6)

{



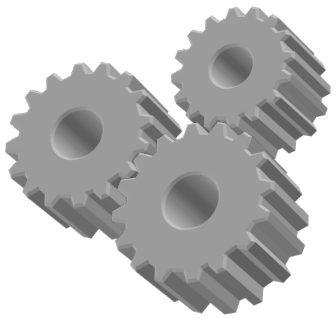
行ではなく、文字単位で
bsort.kの内容を表示する

- sourceFile ＝ bsort.kファイルを参照
- line ＝ "main() {\n"
- lineNumberum ＝ 1
- columnNumber ＝ 0
- currentCharacter ＝ '\n'
- nextCharacter ＝ 'm'

注: 問題2.6のFileScanner.javaも提出しない

main(問題 2.6)

{



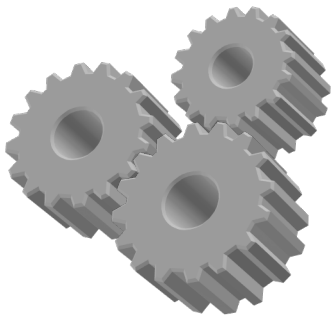
行ではなく、文字単位で
bsort.kの内容を表示する

- sourceFile ＝ bsort.kファイルを参照
- line ＝ "main() {\n"
- lineNumberum ＝ 1
- columnNumber ＝ 1
- currentCharacter ＝ 'm'
- nextCharacter ＝ 'a'

注: 問題2.6のFileScanner.javaも提出しない

main(問題 2.6)

{



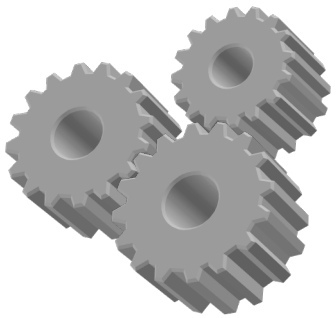
行ではなく、文字単位で
bsort.kの内容を表示する

- sourceFile ＝ bsort.kファイルを参照
- line ＝ "main() {\n"
- lineNumberum ＝ 1
- columnNumber ＝ 2
- currentCharacter ＝ 'a'
- nextCharacter ＝ 'i'

nextCharacter = line.charAt(…);など、
文字列処理を行う様々なメソッドの使い方は、オブジェクト指向Java
プログラミング入門の6章や、Web上のJava APIのページを参照

main(問題 2.7)

{



line, lineNumber, columnNumber の内容をわかりやすく表示するメソッド　scanAt()を追加

- sourceFile　＝ bsort.kファイルを参照
- line　＝ "main() {\n"
- lineNumberum　＝ 1
- columnNumber　＝ 2
- currentCharacter ＝ 'a'
- nextCharacter　＝ 'i'
- scanAt()

注: 問題2.7のFileScanner.javaを提出する

問題2.7 で作成したプログラムを公式ページから提出する。詳しい使い方は公式ページを参照。

https://www.info.kindai.ac.jp/project1/

```
/* 提出者: 21-1-037-0999　山田太郎
   問題番号:  問題2.7
   提出日:　 2023年4月19日
 */
/**
  クラスFileScannerは…（FileScannerとはどのようなクラスなのか、
  役割や使われ方について、冒頭で必ずコメントする）
*/
class FileScanner {

            ...
}
```