情報処理 II No.1

# ファイルの扱い方

桂田 祐史

1995 年 4 月 21 日

今回のテーマは (1) UNIX[1]の持つ便利な機能である、標準入出力のリダイレクション&パイプ、(2) FORTRAN でのファイル処理用の命令、の二つを学ぶことです。

## 1 UNIX の標準入出力

### 1.1 FORTRAN における標準入出力の扱い

情報処理・演習で FORTRAN によるプログラミングを勉強した人は、通常

- “read(*, )” によりキーボードからの入力を読み込める。
- “write(*, )” によって画面へ文字を出力（＝画面に表示）できる。

ということを学んだと思います。例えば、

```
* mean.f -- 二つの数の相加平均、相乗平均を計算する FORTRAN プログラム
      program mean
      real a,b
      read(*,*) a,b
      write(*,*) (a+b)/2, sqrt(a * b)
      end
```

は (普通に実行すると)、キーボードから 2 つの数を変数 “a”, b” に読み込んで、その相加平均 $\frac{a+b}{2}$, 相乗平均 $\sqrt{ab}$ を画面に表示する FORTRAN プログラムです。

```
waltz11% nemacs mean.f        ← nemacs で “mean.f” を作る。
waltz11% f77o mean.f          ← f77o で “mean.f” をコンパイルして “mean” を作る。
waltz11% mean                 ← “mean” を実行する。
2 3                           ← 2, 3 を入力
 2.50000000 2.44948983
waltz11%
```

実は UNIX における FORTRAN では、

[1]大半のワークステーション（もちろんこの情報処理 II で使っているものも含む）で採用されているオペレーティング・システム（コンピューター・システムの基本的な部分を制御・管理しているソフトウェアのこと）の名前。

“read(*, )” の “*” は**標準入力**と呼ばれる**入力ファイル**
“write(*, )” の “*” は**標準出力**を呼ばれる**出力ファイル**

を意味していて、通常は、

**標準入力**はキーボードからの入力
**標準出力**は画面への出力

となっているわけです。

## 1.2 リダイレクション

標準入力はキーボード入力に、標準出力は画面出力に、それぞれ結び付けられていると書きましたが、この結合は簡単に変更することができます。

- “**コマンド** > **出力ファイル名**” のようにすると、コマンドの標準出力を画面の代わりに指定したファイルに書き出します。
- “**コマンド** < **入力ファイル名**” のようにすると、キーボードの代わりに指定したファイルからコマンドの標準入力を読み込みます。

以上のことは上のサンプル・プログラムに限らず、ほとんど総てのプログラムについて当てはまることです。例えばカレンダーを表示するコマンド “cal”[2]を用いて

```
cal 1995 > cal.out
```

とすると “cal.out” という名前のファイルが出来て、その中身は 1995 年のカレンダーになります。このように標準入出力を通常とは違うものに結合させることを**リダイレクション** (redirection) と呼びます。

次のようにすると、何が起こるか分かりますか？

```
waltz11% mean > data1
2 3
waltz11% mean < data1
```

## 1.3 パイプ

あるプログラムが計算したデータを、別のプログラムに渡したい場合があります。前者のプログラム（名前を “prog1” としましょう）が計算データを標準出力に書き出し、後者のプログラム（名前を “prog2” としましょう）が標準入力から読み込むようになっていれば、

```
prog1 > data
prog2 < data
```

としてファイル “data” を橋渡し役にすれば良いわけですが、この種の状況はしばしば生じるため、ファイルを使ったりせずに、次のように済ますことが出来ます（この機能を**パイプ**と呼びます）。

```
prog1 | prog2
```

[2]余談ですが “cal 1752” が面白いです。9 月に注目。

例えば、二つの数を書いたファイル “input-data” がある時に、最初のプログラム “mean” を使って、それら数の相加平均、相乗平均を計算するには、

```
cat input-data | mean
```

とすれば OK です（やってみましょう）。前節の最後に書いた例は

```
waltz11% mean | mean
2 3
```

もう一つの例として、今年のカレンダーを印刷するには、 cal コマンドと、印刷コマンド lpr をパイプでつないで、“cal 1995 | lpr” とすれば OK です。

UNIX にはパイプでつないで使うと便利なように、標準入力から入力し、標準出力へ出力するようになっているコマンドがたくさんあります[3]。

## 2 FORTRAN のファイル処理

上記のサンプル・プログラムでは、標準入出力を用いましたが、特定のファイルを指定して入出力することも可能です。

```
* sum2.f -- 二つの整数の和を計算し、結果を myfile に出力する
      program sum2
      real a,b
      open(1,FILE='myfile')
      read(*,*) a,b
      write(1,*) a+b
      close(1)
      end
```

このプログラムでは、結果を標準出力ではなく、“myfile” という名前のファイルに書き出しています (ファイルの名前は自分で好きなものが選べます)。まず最初に “open(1,FILE='myfile')” で “myfile” という名前のファイルを開き[4]、番号 1 を対応させます。次に “write(1,*) a+b” で番号 1 のファイルに出力しています。最後に “close(1)” で番号 1 のファイルを閉じています。

同様のことは **write(,)** による書き出しだけでなく、 **read(,)** による読み込みでも出来ます。試しに “input-data” というファイルから整数を読み込むようなプログラムを書いてみましょう。

## 3 次回のための準備

0 度から 1 度おきに 90 度までの sin の表を作りなさい。各行が “$\theta$ の値 (単位は度)　$\sin\theta$ の値” という形式になっている全部で 91 行のデータ・ファイル “sintable” を作りましょう。§1 の方法、§2 の方法の両方でやってみましょう。

---

[3] ファイルの先頭部分だけを読む “head”, 末尾部分だけを読む “tail”, 特定のパターンを検索する “grep”, 並べ換えをする “sort”、文字列の置換等の編集をする “sed”, 1 ページずつ区切って表示する “less”, e.t.c.

[4] ファイルの読み書きに先だって必要な準備をすることを「ファイルを開く (open する)」と言います。

## 4　それでも時間があまってしまった人のために

次回くわしく説明しますが、次のことを試してみましょう。

```
waltz11% getsample
waltz11% ls
waltz11% f77o trans1.f
waltz11% cat rasen.data | mgraph | xplot
waltz11% cat rasen.data | trans1 | mgraph | xplot
waltz11% cat rasen.data | trans1 | trans1 | mgraph | xplot
```

何が起こっているか、分かりますか？“trans1.f”を読んでみましょう。