情報処理 II 第 6 回

情報の電子化 (1) 文字コードとテキスト・ファイル

かつらだ まさし
桂田 祐史

2001 年 5 月 31 日

1 連絡事項

- 次回の授業までに、 tcsh を使う設定をすませておくことを強く勧める (キーボード入力が楽になる)。情報処理 II ホームページの『tcsh のすすめ』の『csh の代りにいつでも tcsh を利用する』参照。

2 コンピューターのファイル — 万物はビットである

2.1 デジタル、ビット、バイト

現在の一般的なコンピューター[1]の処理するデータはデジタル [2]である。
より具体的に言うと、

コンピューター上のすべてのデータは、ビット (bit)[3]の有限列である。

実際には、複数のビットを適当な大きさにまとめて処理することが多い。例えば

- 表示する際には、 3 桁, 4 桁ごとに区切って 8 進数, 16 進数とする[4]。
- データの格納や移動の際には、バイト[5] やワード[6]という単位が使われることが多い。

コンピューター上の情報はすべてファイルとして保存できるが、もちろん中身は数列である。特に、パソコンの OS (MS-DOS, Windows) や、ワークステーションの OS である UNIX では、

ファイルはバイトの有限列である (ファイルは数列である)

[1]コンピューターの古い呼び方は — electric digital computer (電子計数型計算機) — という。アナログ計算機というものもあった。

[2]デジタル (digital) — もともとは「指の」という意味だが、ここでは「(離散的な) 数字の」という意味。

[3]0 または 1 の二つの状態のいずれであるかを示す、情報量の単位をビットと言う。電子回路との相性がいい。

[4]コンピューターの世界では、 16 進法の数字として '0' ~ '9', 'A', 'B', 'C', 'D', 'E', 'F' を用いるのが普通。

[5]バイト (byte) — 通常は 8 ビットのこと。 2^8 通りの状態を区別出来るので、例えば 0 ~ 255 までの数値を割り当てて読むことがよく行なわれる。

[6]ワード (word) — その計算機にとって都合のよいデータの大きさで、普通はバイトの整数倍の大きさ。 16 bits, 32 bits, 64bits などの場合が多い。

と言って構わない。

色々なデータ (文書、数値データ、静止画、動画、音声, etc.) を記録したファイルがあるが[7]、バイトの列であることに変わりはない。ファイルの保存、コピー、通信による転送など、共通の操作で扱うことが出来る。

2.2 ファイルの大きさを調べる

UNIX 環境では、ファイルの大きさは、 `ls -l` または `wc ファイル名` とすれば分かる (前者はバイト数、後者は文字数、単語数、行数を調べる)。自分がどれだけディスクを消費しているかは “`du -k ~`” とすれば分かる (実習に使っているワークステーションでは、単位は KB である)。

K, M, G, T, .. 「メートル法」では、大きな量を表すために、 k, M, G, $\cdots$ を補助的に用いる。コンピューターの世界でも、 K$= 1024 = 2^{10}$, $\overset{\text{メガ}}{\mathrm{M}} = K^2 =$ 約 100 万, $\overset{\text{ギガ}}{\mathrm{G}} = K^3$, $\overset{\text{テラ}}{\mathrm{T}} = K^4$, $\overset{\text{ペタ}}{\mathrm{P}} = K^5$ などを用いる。身の回りにある記憶装置、記憶媒体の容量を調べてみよう。 (メイン・メモリー、フロッピー・ディスク、ハードディスク、 CD-ROM, MO, DVD など。)

2.3 レポート課題 X

以下のことを調べよ (〆切は 6 月末日とするので、少しずつ調べていこう)。 (ファイルのサイズについての感覚を身につけるのが主旨。)

(1) WS 上のファイルのサイズについて調べよ。バイト数以外に、フロッピーや CD-ROM にどれくらい入るかを記せ。

  (a) 文書ファイル。
  レポート、メイル、 C プログラム、 TeX のソース (`.tex`) など。
  (印刷して何ページくらいの文書が何バイトになるか。)

  (b) 実行可能プログラム
  例えば C プログラムはコンパイル前と後でどうサイズが変わるか。
  自分が普段使っているプログラム[8]をいくつか選び、そのプログラム・ファイルのサイズを調べよ?

  (c) (もし出来れば) 画像ファイル、音声ファイルなど。
  (図の大きさ、色数も分かれば調べる。)

  (d) 現在、自分が持っているファイルの総容量。

---

[7] 余談だが、ファイルの種類をある程度まで自動的に判別するコマンド `file` がある。 `file *` としてみよう。

[8] コマンドの実体がどこにあるか (パス名) は、 which コマンドを使って調べられる。例えば、 `which netscape.v47j` とすると、netscape コマンドのパス名は `/usr/meiji/X11/bin/netscape.v47j` であることが分かる。ただし、これは実はシェル・スクリプトであり、その中身を読むと、netscape のプログラム本体のパス名は `/usr/meiji/X11/netscape.v47j/netscape` であることが分かる。同様に mule を調べると、最初に `/usr/meiji/gnu/bin/mule` というパス名が得られるが、これはシンボリック・リンクというもの (Windows でいうショートカット) で、本体は `/usr/meiji/gnu/bin/emacs` である。

```
du -k ~
```

(結果の単位は KB である。)

(2) 自分が持っている本を一冊選び、その文字情報を記憶するファイルを作った場合、サイズはどれくらいになるか計算せよ。フロッピー・ディスクに記憶する場合、何枚必要か？また CD-ROM (容量 650MB 程度) には何冊分記憶できるか。

3　テキスト・ファイル

3.1　最初に

テキスト・ファイルとは　コンピューター上のファイルのうち、普通の意味で「読める」字で出来ているファイルのことをテキスト・ファイルという。今回は実験しながら、テキスト・ファイルの構造について学ぶ。

文字コードとは　コンピューターでは、文字も数で表す。このためには「この数はこの文字」とルールを決める必要がある。文字を表現するため、表現したい文字の集合と、数の集合を選び、両者の間の 1 対 1 の写像を定めて用いる。このとき、ある文字に対応する数を、その文字の文字コードと呼ぶ。

3.2　英数字の文字コード (ASCII)

まず mule を使って、

ascii.txt
```
012
ABC abc
```

という 2 行からなるファイルを作成し、以下のコマンドを実行してみよう。

```
waltz% od -cx ascii.txt
```

(od -cx　ファイル名とすると、ファイルの文字コードが 16 進数で表示される。)

次の事が分かる。

- 数字 ‘0’, ‘1’, ··· の文字コードは 0x30, 0x31, ···
- 英字 ‘A’, ‘B’, ··· の文字コードは 0x41, 0x42, ···
- 英字 ‘a’, ‘b’, ··· の文字コードは 0x61, 0x62, ···
- 行の終りは、‘\n’ という一つの文字 (文字コードは 0x0a) で表される。
- 空白 ‘ ’ も一つの文字 (文字コードは 0x20) で表される。

(相当に不正確な言い方だが) このような「半角」英数字の文字コードは ASCII (アスキー) というアメリカの規格で定められ、それが色々な規格に「輸入」されている。

3.3　ASCII コード表を作ろう

print_code_table.c

```
/*
 * print_code_table.c --- ASCII コード表（印字可能な文字のみ）を表示する。
 */

#include <stdio.h>

main()
{
    int code;
    /* 0x20 (==32) から 0x7e (==126) までの数をコードとする文字を扱う */
    for (code = 0x20; code <= 0x7e; code++) {
        /* 4 つおきに改行 */
        if (code % 4 == 0)
            printf("\n");
        /* 16 進数，10 進数，文字 として表示 */
        printf("0x%02x (%3d): %c    ", code, code, code);
    }
    printf("\n");
}
```

というプログラムを実行してみよう (WWW ページに載せてあります — 入手法は前回説明しました)。

C 言語の知識として

- `putchar(整数式);` あるいは `printf("%c", 整数式);` でその整数を文字コードとする文字が表示される。
- `printf()` で 16 進数を表示するには `%x` という書式を用いる。

3.4　日本語の文字コード

日本語の文字コードは JIS (日本工業規格) で決められている。日本語における文字数が多いため[9]、1 文字を表すのに 16 ビットを用いる。

例えば「桂」という文字の JIS コードは 0x374b である。 mule で C-^とすると、

```
記号入力:  0.JIS 入力  1.記号  2.英数字  3.ひらがな  4.カタカナ  5.ギリシャ文字
```

というメニューが現れる。 0 を選択し “374b” と入力すると「桂」という文字が入力できる。

しかし、ファイルの中に 0x37, 0x4b というバイト列をそのまま入れたのでは、 ASCII の ’7’, ’K’ と区別がつかない。両者を混在させるには何らかの工夫が必要になる。

情報科学センターのワークステーションでは日本語 **EUC**という文字コードを用いている。

kanji.txt

```
桂田 祐史
```

というファイルを `od -cx` してみると、「桂」が 0xb7, 0xcb という 2 バイトで表現されていることが分かる。これは JIS コードを上下 8 ビットにそれぞれ 0x80 を加えたものになっている:

$$\texttt{0x37} + \texttt{0x80} = \texttt{0xb7}, \quad \texttt{0x4b} + \texttt{0x80} = \texttt{0xcb}$$

[9] ちなみに JIS の情報交換用漢字符号系コードの第一、第二水準の漢字は約 7000 個弱ある。

(もともと ASCII で用いている数値は 0x7f までで、 0x80 以上の数値は使われずに空いている。その部分に JIS コード表の文字を埋め込んだことになっている。)

日本語 EUC 以外にも、 ASCII の文字と日本語の文字を混在を可能にした文字コードはいくつかある。

通称「JIS 漢字」 通信により情報を交換するために作られた国際的な規格に従ったもの。文字コードを切り替えるために、目印となる特別な文字列を用いる。電子メイルなどで通信をするときに最もよく使われる。実は kterm では JIS 漢字のデータも普通に表示できる。

```
tango21% cat kanji-jis.txt
桂田 祐史
tango21% od -cx kanji-jis.txt
0000000 033   $   B   7   K   E   D 033   (   B     033   $   B   M   4
            1b24    4237    4b45    441b    2842    201b    2442    4d34
0000020   ;   K 033   (   B  \n
            3b4b    1b28    420a
0000026
```

通称「MS 漢字 (シフトジス)」 パソコンの世界のために作られた規格 (Windows, Mac などで採用されている)。「半角カタカナ」も効率的に埋め込まれている。情報科学センターのワークステーションの通常の設定では直接表示できない。

```
tango21% cat kanji-ms.txt
jc S■ (いわゆる文字化け)
tango21% od -cx kanji-ms.txt
0000000 214   j 223   c     227   S 216   j  \r  \n
            8c6a    9363    2097    538e    6a0d    0a00
0000013
tango21%
```

3.5 レポート課題 3

自分の名前を構成する文字 (桂田祐史なら「桂」,「田」,「祐」,「史」の 4 文字) の JIS コードを調べよ。 (過去に同じ課題を出したとき、間違えて EUC コードを答えた人が少なくなかった。)

3.6 文字コードをいじってみよう

WWW ページに `mycat.c` というプログラムを載せてあります (繰り返しになりますが、入手法は前回説明しました)。これは `cat` コマンドの真似をして、ファイルの内容を標準出力 (通常は画面) に出力するだけのプログラムです。

```
waltz12% cc -o mycat mycat.c
waltz12% mycat mycat.c
```

このプログラム中の `print_file()` という関数を書き換えると、色々なことができます。例えば

日本語 EUC に対応した dump

```
print_file(FILE * fp)
{
    int c, c2;
    /* ファイルの終りまで一文字ずつ c に読み込み、標準出力に書き出す */
    while ((c = getc(fp)) != EOF) {
        if (c >= 0x80) {
            /* 0x80 以上だったら漢字の 1 バイト目だと判断して、
               もう 1 バイト読んで、まとめて出力する。 */
            c2 = getc(fp);
            printf(``0x%02x 0x%02x: %c%c\n'', c, c2, c, c2);
        } else if (c < 0x20 || c == 0x7f)
            /* 0x20 未満または 0x7f の場合は文字コードのみ表示 */
            printf(``0x%02x\n'', c);
        else
            /* それ以外の場合は文字コードと、その文字自身を出力 */
            printf(``0x%02x: %c\n'', c, c);
    }
}
```

```
tango21% cat ascii_and_kanji.txt
I am 桂田祐史.
tango21% mydump ascii_and_kanji.txt
0x49: I
0x20:
0x61: a
0x6d: m
0x20:
0xb7 0xcb: 桂
0xc5 0xc4: 田
0xcd 0xb4: 祐
0xbb 0xcb: 史
0x2e: .
0x0a
tango21%
```

腕試し用プログラミング課題 1　ファイルの中の英小文字を英大文字に変換するプログラムを作りなさい。(メイルでレポートを送るときは Subject は “prog 1” として下さい。)

腕試し用プログラミング課題 2　ファイルのバイト数、行数を数えるプログラムを作りなさい。(‘\n’ (文字コード 0x0a) の個数を行数と考えることにする。) (メイルでレポートを送るときは Subject は “prog 2” として下さい。)

(以下は付録)

A　日本語の文字コードの変換

UNIX 上のコマンドには自動的に文字コードを判別して必要な処理をしてくれるものがあるが (mule, less 等)、時にはユーザーが意識的に変換することが必要になる。

それほど難しい作業でもないのでフリーソフトがある。二つほど紹介する。

nkf (Network Kanji code conversion Filter) UNIX では定番。

nkf -e ファイル名 で日本語 EUC に変換したものを標準出力に書き出す。
nkf -j ファイル名 で JIS 漢字コードに変換したものを標準出力に書き出す。
nkf -s ファイル名 で MS 漢字コードに変換したものを標準出力に書き出す。
電子メールで使われる MIME のデコードもできる。
nkf -v でオプションの一覧が表示される。

```
waltz12% nkf -j kanji.txt > kanji-jis.txt
```

qkc (Quick KANJI code Converter) Windows 版もある。行末の変換もしてくれる。

qkc -eu ファイル名 で日本語 EUC, 行末を UNIX 形式に変換する。
qkc -ms ファイル名 でシフトジス, 行末を MS-DOS 形式に変換する。

```
waltz12% cp kanji.txt kanji-ms.txt
waltz12% qkc -ms kanji-ms.txt
```

B　インターネットで使って良い文字悪い文字 (日本語)

コンピューター上で色々な日本語文字が使えるようになっていて、テキスト・ファイルを作ることが出来るが、インターネット (電子メイル、 WWW ページ、ネットニュースなど) で用いる場合、すべての文字を使えるわけではない。

- 電子メイル、ネットニュースでは、**ISO-2022-JP** と呼ばれる文字コードが使える。
  インターネットの作法である RFC で定義されている。
  → http://www.noge.com/koba/network/RFC/rfc1468.html
  元々は日本のインターネットのルーツである JUNET で利用されて来たもの。
  ASCII, JIS X 0201-1976, JIS X 0208-1978, JIS X 0208-1983 という 4 種類の文字コードをエスケープ・シーケンスというバイト列でスイッチする。
  - ASCII はアメリカの規格 (キーボードから直接打ち込める英数字・記号)。
  - JIS X 0201 はその日本版 (円記号とバックスラッシュなど ASCII と異なる)。
  - JIS X 0208 はいわゆる JIS 第一、第二水準の漢字、ひらがな、カタカナ、その他記号。
- 「外字」は使えない (論外)。
  (外字については後述。)
- 機種依存文字も使ってはいけない。
  (コンピューターやソフトウェアのメーカーが作った文字。普通の人には、 JIS で正式に定義された文字と見分けがつきにくいかも知れない。丸つき数字などが有名。)

- JIS X201 の右半面 (俗称「半角カナ[10]」) は RFC に違反しているので、メイルやネットニュースでは使用してはいけない。 WWW ページでは、ルールがまとまる前になしくずしに使われてしまった (そのおかげで化けるページを根絶できないなどの弊害が残った)。

参考　「日本語フォントや文字コードについて」

http://www.ryukyu.ad.jp/%7Eshin/jdoc/

[10]昔の MS-DOS パソコンやワープロ専用機では、これらの文字が他の日本語文字の半分の幅で表示されたため、こう呼ばれた (←→全角)。元々は印刷業界の言葉であったとか。しかし、そもそも規格には「半角」という言葉はない。文字がどう表示されるかは、利用する環境によるので規格の範囲外である。