キーボードからの入力，画面への出力，さらにリダイレクトの機能

C言語の入出力の書き方

岸 哲夫

UNIX系のOSには，キーボードや画面などの標準入出力のほかに，入出力をほかのデバイスに付け替えるリダイレクトという機能がある．ここでは，この機能とファイル・システムへのデータの読み込み/書き込みの方法について解説する．（編集部）

C言語には標準入力（キーボードからの入力），標準出力（画面への出力）また，1文字ずつ入出力ができる入出力関数，1行専用の入出力関数が用意されています．

C言語の標準入出力

C言語での標準入力はキーボードからの入力で，標準出力は画面への出力です．とは言っても，実際にキーボードからの入力で標準入力を行うことは少ないと思います．

UNIX系OSのシェルには「リダイレクト」という機能があり，標準入出力をほかのデバイスに付け替えることもできます．

標準入出力や標準エラー出力，ファイル入出力のためのライブラリや宣言を記述してあるヘッダが，標準ヘッダに含まれるstdio.hです．ファイル・オープン/ファイル・クローズ/リネーム/削除などを行う関数も用意されています．

つぎに，標準入出力の関数について説明します．

● getchar ── 標準入力が示す入力列の次の文字を返す

getcharは標準ライブラリにある関数です．標準入力が示す入力列の次の文字を返します．入力列でファイルの終わりが検出された場合，そのストリームに対するエラー表示子をセットしてEOFを返すものです．

リスト1に示すように，押下した文字が表示されないように，処理の前後で端末属性を変え，非カノニカル・モードにするため，termios構造体をtcgetattr，tcsetattr関数で操作しています．

リスト1をコンパイル実行した結果が**図1**です．

● putchar ── 標準出力に指定した文字を一文字出力

putcharはやはり標準ライブラリにある関数です．標準出力に指定した文字を一文字出力します．

リスト2に示すように，引き数に出力したい文字をセットし，返り値はその文字の整数値です．

リスト2をコンパイル実行した結果が**図2**です．

● gets ── 使わない

getsは標準ライブラリにある関数で，昔から使われているのですが，仕様に大きな問題があるので，最近は「使わないほうがよい」と言われています．それは，単純に用意したバッファ以上のデータが与えられた場合に，動作の保障ができないからです．

この場合，いわゆる「バッファ・オーバフロー」攻撃にさらされることになります．

リスト1　getcharを使ったソース・リスト（test24.c）

```
/*
 *getchar
 */
#include <stdio.h>
#include <termios.h>

int main(void)
{
    struct termios term1, term0;
    int ch;

    tcgetattr(0, &term1);                   //現在の画面環境の取得
    term0 = term1;                          //現在の画面環境の保存
    term1.c_lflag &= ~(ICANON | ECHO);
              //現在の画面環境をエコーオフおよび非カノニカル・モードに設定
    term1.c_cc[VMIN] = 1;                   //そのためのパラメータ
    term1.c_cc[VTIME] = 0;                  //そのためのパラメータ
    tcsetattr(0, TCSANOW, &term1);          //現在の画面環境を更新

         ch   =    getchar();
         printf("押下したキーは %c です¥n",ch);

    tcsetattr(0, TCSANOW, &term0);
                    //保存しておいた環境で現在の画面環境を更新（元に戻す）
    return 0;
}
```

```
$ ./test24
押下したキーはｅです
```

図1　test24.cをコンパイルして実行した結果

KEYWORD ──標準入出力，リダイレクト，getchar，putchar，gets，puts，fgets，printf，sprintf，scanf，fscanf，Ext2，Ext3，ReiserFS，JFS，XFS，低水準入出力関数

リスト2　putcharを使ったプログラム・ソース・リスト(test25.c)

```
/*
 *putchar
 */
#include <stdio.h>

int main(void)
{
        int             ch   =    'c';
        int             res;
        res       =     putchar(ch);
        res       =     putchar('¥n');
        printf("戻った整数値は%dです¥n",res);

    return 0;
}
```

図2
test25.cをコンパイルして実行した結果

```
$ ./test25
c
戻った整数値は10です
```

リスト3　putsを使ったソース・リスト(test26.c)

```
/*
 *putchar
 */
#include <stdio.h>

int main(void)
{
        char            str[]     =    "試験";
        puts(str);
    return 0;
}
```

図3
test26.cをコンパイルして実行した結果

```
$ ./test26
試験
```

リスト4　fgetsを使ったソース・リスト(test27.c)

```
/*
 *getchar
 */
#include <stdio.h>
#include <termios.h>

int main(void)
{
    struct termios term1, term0;
    char buf[11];

    tcgetattr(0, &term1);                   //現在の画面環境の取得
    term0 = term1;                          //現在の画面環境の保存
    term1.c_lflag &= ~(ICANON | ECHO);
            //現在の画面環境をエコー・オフおよび非カノニカル・モードに設定
    term1.c_cc[VMIN] = 1;                   //そのためのパラメータ
    term1.c_cc[VTIME] = 0;                  //そのためのパラメータ
    tcsetattr(0, TCSANOW, &term1);          //現在の画面環境を更新

        printf("押下した文字は%sです
                        ¥n",fgets(buf,sizeof(buf),stdin));

    tcsetattr(0, TCSANOW, &term0);
                    //保存しておいた環境で現在の画面環境を更新（元に戻す）
    return 0;
}
```

図4
test27.cをコンパイルして実行した結果

```
$ ./test27
押下した文字は0123456789です
```

● puts ── 指定した文字列ポインタを渡すとその文字列を標準出力に表示

putsは標準ライブラリにある関数です．指定した文字列ポインタを渡すとその文字列を標準出力に表示し，復帰改行を行います．

しかし，このようにして日本語文字列を出す場合は，漢字コードに注意しなくてはいけません．具体的に言うと，0x5c「¥」を内部に含むShift-JISコード，たとえば「表」などは化けたりコンパイルが通らなかったりします．

リスト3はputsを使ったソース・リスト，**図3**はコンパイル実行した結果です．

● fgets ── 指定した容量まで文字列を入力

getsは危険ということで，fgetsがよく使われます．これはバッファの大きさが256バイトの場合には255文字までしか入力できません．

リスト4の例では11バイト分のバッファを取って入れています．最後のNull文字が文字列に必要なので，10バイト分の文字が入ります．getsを使うと何文字でも入りますが，fgetsだと指定しただけしか入らないので，バッファ・オーバフロー攻撃でも問題ありません．

図4は**リスト4**をコンパイル実行した結果です．

● printf ── 書式付きで標準出力に表示

printfはよく使われる関数です．

書式つきで標準出力に表示するものです．GDBなどのデバッガを使わない場合は，これを使ってデバッグするとよいでしょう．使い方は簡単ですが「フォーマット文字列」を覚えるのがたいへんかもしれません．まぁ覚えなくても，オンライン・マニュアルを見ればよいでしょう．

この関数の特殊な点は引き数の数が可変であるということです．フォーマット文字列内に出力対象のフォーマットをパラメータとして埋め込む方法なのでそのようになります．

リスト5にprintfを使ったソース・リストを，**図5**にそのコンパイル実行結果を示します．

リスト5のソースではフィールド幅，精度，フラグ文字「0」，10進表現，指数表現，文字や数値や文字列の表示を行っています．

● sprintf ── 標準出力のprintfの機能をそのまま指定した領域に出力

この関数は標準出力に出力するprintfの機能をそのまま指定した領域に出力する関数です．これを使えば右詰め左詰め表示などが簡単にできます．

なお数値を123,456,789などと表示する機能はprintfのフォーマット文字ではできないので，くふうして関数を作ったほうがよいでしょう．