

プログラミング演習 I 第 11 回

文字と文字列の処理 (2)

今日の目標：文字および文字列処理を使いこなす。

1. 文字処理関数 (教科書 p.202)

前回は文字列処理標準関数を紹介したが、**文字単位**の処理関数も多数標準関数として用意されている。

例：

tolower(文字)：文字の小文字変換

toupper(文字)：文字の大文字変換

isalpha(文字)：文字がアルファベット（大文字、小文字は問わない）であれば真を返す

isdigit(文字)：文字が数字なら真

islower(文字)：文字が小文字なら真

isupper(文字)：文字が大文字なら真

など。これらは ctype.h に定義されているので、インクルードを忘れずに。

以下、基本的な使用例を示す。入力して動作確認せよ。

(1) 入力の全ての文字を大文字変換

リスト 1:

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#define STRLEN 20

int main(void)
{
    char a[STRLEN];
    int i;

    printf("String:");
    scanf("%s", a);
    for (i = 0; i < strlen(a); i++)
        a[i] = toupper(a[i]); /* 一文字ずつ大文字に変換 */
    printf("%s\n", a);
    return 0;
}
```

表示例:

```
$ example11_1.exe[Enter]
String: abc4565DE[Enter]
ABC4565DE
$
```

2. 動きのある文字列表示

以下のプログラムを入力して動作確認せよ。printf の文字列にある '\r' は改行なしで同じ行に上書きする制御文字である。また、fflush(stdout) は printf を強制出力するものである。これがないと何行かまとめて出力してしまうので、動きがスムーズにならない (fflush を削除してみよ)。

```

リスト2:
#include <stdio.h>
#include <string.h>

/* タイマー周期 */
#define TINTERVAL 50000000

#define MAXLINE 80
#define COUNT 5

void forward(char a); /* 文字列を順に右に動かす関数のプロトタイプ */

int main(void)
{
    char a;
    int i;
    a='*'; /* 動かす文字列 */
    for (i = 0; i < COUNT; i++) /* COUNT 回右端まで動かす */
        forward(a); /* 一回の呼び出しで右端まで移動 */

    return 0;
}

/* 文字列を順に右に動かす関数 */
void forward(char a)
{
    char alocal[MAXLINE]; /* 表示用文字配列 */
    char blocal[MAXLINE]; /* 途中作業用文字配列 */
    int i, j;

    /* move character forward */
    alocal[0] = a; /* 最初の文字をセット */
    alocal[1] = '\0'; /* 終端文字をセット */
    for (i=0; i < MAXLINE - 2; i++){
        printf("%s\r", alocal);
        fflush(stdout); /* 強制出力 これがないとスムーズに動かない */

        strcpy(blocal, " "); /* 先頭に空白追加 */
        strcat(blocal, alocal); /* その後に今までの行追加 */
        strcpy(alocal, blocal); /* 全行を alocal に戻す */

        /* 少し時間を空ける → 表示が速すぎないように */
        for (j=0; j < TINTERVAL; j++){
        }

    }
    /* 最後に一行消す */
    printf("                                \r");
}

```

3. 本日の演習

- (1) リスト1のプログラムを入力し、その動作を確認せよ。
- (2) 文字型の配列を宣言して、**自分の名前と苗字**（全て小文字）で初期化せよ。
 例: `char name[] = "kazuhiko kondo";`
名前の一文字目と、**苗字全て**を `toupper` を用いて大文字にし、表示せよ。苗字に相当する文字列はプログラムで判定せよ（スペースの後の文字列）。ヒント: スペースの検出は `for` ループで `if (name[i] == ' ')` と判定すればよい。
 表示例:

```
$ lab1_2_1.exe[Enter]
Name: Kazuhiko KONDO
$
```
- (3) キーボードから任意の文字列を入力し、入力文字列中のアルファベット大文字を全てプラス（`'+'`）、小文字をハイフン（`'-'`）、数字を0（ゼロ）、区切り文字を:（コロン）で置き換えて表示せよ。ヒント: 大文字の検出は `for` ループで `if (isupper(a))`, 小文字は `islower(a)`, 数字は `isdigit(a)`, 区切り文字は `ispunct(a)` を使って判定すればよい。
 表示例:

```
$ lab12_2[Enter]
Enter a string: abcABC123,."[Enter]
Modified string: ---+++000:::
$
```
- (4) リスト2のプログラム（動きのある文字列表示の例）を入力し、動作を確認せよ。なお、この課題の動きのビデオを下記にアップロードしておくので、参考に。
<http://spandaudiolab.yz.yamagata-u.ac.jp/kkondo/programming1/lesson11/lab4.mp4>
- (5) 文字型の配列 `a` を以下のように宣言し、初期化せよ。
`char a[] = "abcdefghijklmnopqrstuvwxyz";`
 左端の `a` から順に**一文字ずつ**順に大文字にせよ。リスト2を参考にせよ。
 この課題の動きのビデオを下記にアップロードしておく。
<http://spandaudiolab.yz.yamagata-u.ac.jp/kkondo/programming1/lesson11/lab5.mp4>
- (6) (5)のプログラムを改良し、**1文字だけ**左から順に大文字になり、その後小文字に戻る用に改良せよ。すなわち、同時に大文字になるのは1文字だけとなるようにせよ。これを `COUNT` で指定される回数繰り返せ。この課題の動きのビデオを下記にアップロードしておく。
<http://spandaudiolab.yz.yamagata-u.ac.jp/kkondo/programming1/lesson11/lab6.mp4>
- (7) （発展）余裕がある人は挑戦してみてください。
 リスト2を改良して、自分の名前（ローマ字）をフルネームで動かすようにしてください。
 この課題の動きのビデオを下記にアップロードしておく。
<http://spandaudiolab.yz.yamagata-u.ac.jp/kkondo/programming1/lesson11/lab7.mp4>
- (8) （発展の発展）本当に余裕がある人は挑戦してみてください。
 (7)のプログラムを改良して、動きが往復するようにしてください。この課題の動きのビデオを下記にアップロードしておく。
<http://spandaudiolab.yz.yamagata-u.ac.jp/kkondo/programming1/lesson11/lab8.mp4>

本日は(3)のプログラムを提出せよ。

提出期限は今週木曜（7/9）16時までとします。WebClass を使って提出してください。

ファイル名は(学籍番号)_(演習の回)_(課題番号).c と名前を付けて提出してください。例えば学籍番号191234の人が第2回の課題(4)のプログラムを提出する場合は19123_2_4.cと名前を付けて WebClass で提出してください。必ずコンパイルしてデバッグし動作を確認したプログラムリストを提出すること。コメント、インデントも忘れずに。