2011 年度 情報科学演習　演習 3

2011 年 5 月 25 日 (水)

演習 1　以下の (1)〜(5) のプログラムを入力しなさい。**もちろん、コンパイル後、オブジェクトモジュールを実行し、本プリントの実行結果または「新訂 新 C 言語入門 ビギナー編」の実行結果と同じになることを確認すること。**

(1)　■ p.137 の「1 文字入出力プログラム 3」 (ファイル名「`exercise0301.c`」) を入力しなさい。

実行結果：

```
$ ./exercise0301.exe Enter
```

⇓

```
$ ./exercise0301.exe
abcdefg Enter
ABCDEFG
```

⇓

```
$ ./exercise0301.exe
abcdefg
ABCDEFG
hiJKLm Enter
HIJKLM
```

⇓

```
$ ./exercise0301.exe
abcdefg
ABCDEFG
hijklm
HIJKLM
Ctrl+d          ← Ctrl+z を押すとプログラムが途中終了してしまうので、
                  Cygwin 環境では Ctrl+d を押して入力の終わりを知らせる
$
```

(2) ■ p.144 の「ふたつの数値の和差積商を計算するプログラム」(ファイル名「exercise0302.c」) を入力しなさい。

実行結果：

```
$ ./exercise0302.exe
数値 1: 100 Enter
数値 2: 20 Enter
和=120
差=80
積=2000
商=5

$
```

(3) ■ p.145 の「数値の連続入力プログラム」(ファイル名「exercise0303.c」) を入力しなさい。

実行結果：

```
$ ./exercise0303.exe
100 Enter
200 Enter
33.44 Enter
Ctrl+d          ← Ctrl+z を押すとプログラムが途中終了してしまうので、
                  Cygwin 環境では Ctrl+d を押して入力の終わりを知らせる
```

⇓

```
$ ./exercise0303.exe
100
200
33.44
合計=333.440000

$
```

(4) ■ p.153 の「printf のオプション指定子の機能を確認する」(ファイル名「exercise0304.c」) を入力しなさい。

(5) ■ p.155 の「(scanf の) 変換仕様の機能を確認する」(ファイル名「exercise0305.c」) を入力しなさい。

演習 2　「アルファベットの小文字を大文字に変換して表示するプログラム」 (■ p.137) を変更し、実行結果のように「アルファベットの小文字は大文字に大文字は小文字に変換して表示するプログラム」 (ファイル名「`exercise0306.c`」) を作成しなさい。ヒント：

- `while`+`getchar` の定石記法を使用する。
- 大文字を小文字に変換するには`tolower` 関数 (例：`ch = tolower(ch)`) を、小文字を大文字に変換するには`toupper` 関数 (例：`ch = toupper(ch)`) を使用する。なお、これらの関数を使用するにはヘッダファイル「`ctype.h`」の記述が必要である。
  【参考】文字を数値として扱い、定数を加算 (大文字から小文字は +32) または減算 (小文字から大文字は −32) することで変換することもできる。
- 大文字と小文字の判別には`if` 文を使用する (`'a'`=97, `'z'`=122, `'A'`=65, `'Z'`=90)。

実行結果：

```
$ ./exercise0306.exe
AbCdEfG Enter
aBcDeFg
```

⇓

```
$ ./exercise0306.exe
AbCdEfG
aBcDeFg
ABC123xyz Enter
abc123XYZ
```

⇓

```
$ ./exercise0306.exe
AbCdEfG
aBcDeFg
ABC123xyz
abc123XYZ
Ctrl+d          ← Ctrl+z を押すとプログラムが途中終了してしまうので、
                  Cygwin 環境では Ctrl+d を押して入力の終わりを知らせる
$
```

演習 3　「3×3 行列 A, B の積を計算するプログラム」 (ファイル名「exercise0307.c」) を作成しなさい。ただし、以下の実行結果と同じ出力となるようにソースプログラムを記述すること。ヒント：

- for 文を利用 (復習)。
- 結果の出力にはprintf 関数を利用する (変換仕様は「%8.2f」)。
- double 型の配列を次のように与える。

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \iff \texttt{a[3][3] = \{\{1, 2, 3\},\ \{4, 5, 6\},\ \{7, 8, 9\}\}}$$

$$B = \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix} \iff \texttt{b[3][3] = \{\{1, 2, 3\},\ \{4, 5, 6\},\ \{7, 8, 9\}\}}$$

- 行列の積の演算は以下のように定義されている (結果は配列c[3][3] に格納する)。

$$AB(=C) \text{ の } (i,j) \text{ 成分 } c_{ij} = \sum_{k=1}^{3} a_{ik} b_{kj} \iff \texttt{c[i][j]} = \sum_{\texttt{k=0}}^{2} \texttt{a[i][k] * b[k][j]}$$

- 最初にc[i][j] を全て0 に初期化すること。

実行結果：

```
$ ./exercise0307.exe
   30.00   36.00   42.00
   66.00   81.00   96.00
  102.00  126.00  150.00

$
```

演習 5　以下の「ハノイの塔－直接的解法－ (奇数個の円盤)」(ファイル名「exercise0308.c」) は奇数個の円盤を最小回数で棒 A から棒 C に移すプログラムである。このプログラムを変更して偶数個の円盤を最小回数で棒 A から棒 C に移すプログラム「ハノイの塔－直接的解法－ (偶数個の円盤)」(ファイル名「exercise0309.c」) を作成しなさい。ヒント：

- 変数 x[][] は各棒に積まれた円盤の状態を表す。したがって、円盤が 4 枚の場合、初期値は

```
x[3][5] = {{1000, 4, 3, 2, 1},
           {1000, 0, 0, 0, 0},
           {1000, 0, 0, 0, 0}};
```

  となる。なお、変数に設定してある 1000 は各棒の地面を表し、一番大きな円盤の数より大きな値が設定してある。

- 変数 y[] は各棒のに積まれた一番上の円盤の位置または地表 0 を表す。したがって、円盤が 4 枚の場合、初期値は y[3] = {4, 0, 0}; となる。
- 棒 B と棒 C を入れ替えると比較的簡単である。

● ハノイの塔－直接的解法－ (奇数個の円盤) exercise0308.c

```
 1: #include <stdio.h>
 2:
 3: int main(void)
 4: {
 5:     char s, t;
 6:     int i = 0, j, k, l, m;
 7:     int x[3][6] = {{1000, 5, 4, 3, 2, 1},
 8:                    {1000, 0, 0, 0, 0, 0},
 9:                    {1000, 0, 0, 0, 0, 0}};
10:     int y[3] = {5, 0, 0};
11:
12:     while (x[2][5] != 1) {
13:         if (y[i] != 0) {
14:             k = y[i];
15:             m = x[i][k] % 2;
16:             if (m == 1) {
17:                 j = (i + 2) % 3;
18:                 l = y[j];
19:                 if (x[i][k] < x[j][l]) {
20:                     if (i == 0) s = 'A';
21:                     if (i == 1) s = 'B';
22:                     if (i == 2) s = 'C';
23:                     if (j == 0) t = 'A';
24:                     if (j == 1) t = 'B';
25:                     if (j == 2) t = 'C';
26:                     printf("円盤 %d を %c から %c に移す¥n", x[i][k], s, t);
27:                     x[j][l+1] = x[i][k];
28:                     x[i][k] = 0;
29:                     y[i] = y[i] - 1;
30:                     y[j] = y[j] + 1;
31:                     i = i + 2;
32:                     i = i % 3;
33:                 }
34:             }
35:             else {
36:                 j = (i + 1) % 3;
37:                 l = y[j];
38:                 if (x[i][k] < x[j][l]) {
39:                     if (i == 0) s = 'A';
```

```
40:                     if (i == 1) s = 'B';
41:                     if (i == 2) s = 'C';
42:                     if (j == 0) t = 'A';
43:                     if (j == 1) t = 'B';
44:                     if (j == 2) t = 'C';
45:                     printf("円盤 %d を %c から %c に移す¥n", x[i][k], s, t);
46:                     x[j][l+1] = x[i][k];
47:                     x[i][k] = 0;
48:                     y[i] = y[i] - 1;
49:                     y[j] = y[j] + 1;
50:                 }
51:             }
52:         }
53:         i = i + 2;
54:         i = i % 3;
55:     }
56:
57:     return 0;
58: }
```

実行結果：

```
$ ./exercise0309.exe
円盤 1 を A から B に移す
円盤 2 を A から C に移す
円盤 1 を B から C に移す
円盤 3 を A から B に移す
円盤 1 を C から A に移す
円盤 2 を C から B に移す
円盤 1 を A から B に移す
円盤 4 を A から C に移す
円盤 1 を B から C に移す
円盤 2 を B から A に移す
円盤 1 を C から A に移す
円盤 3 を B から C に移す
円盤 1 を A から B に移す
円盤 2 を A から C に移す
円盤 1 を B から C に移す

$
```