

第8章 「いろいろなプログラムを作ってみよう」のまとめ

8.1 用語のまとめ

再帰的 (recursive) (教 p.224)

関数の定義の中で自分自身を呼び出すこと。一般に x の定義に x 自身を使用すること。

```
factorial(4)
→ 4 * factorial(3)
→ 4 * 3 * factorial(2)
→ 4 * 3 * 2 * factorial(1)
→ 4 * 3 * 2 * 1 * factorial(0)
→ 4 * 3 * 2 * 1 * 1
```

- “自分自身”と言っても、変数（List 8-7のfactorialの場合、n）は別々に確保される。
- 繰返し（for, while）で簡単に実現できることを、再帰で書くのは（C 言語の場合）良いこととはいえない。階乗の例題プログラムは、あくまでも再帰を説明するためのものと考えること。（もちろん、再帰を使わなければ簡単に書けないプログラムも多い。）
- 再帰関数には、特別な文法も特別な実行規則も必要ない。あくまでも C 言語の普通の関数で、普通の実行規則に基づいて計算される。

getchar 関数 (教 p.228)

標準入力から を読み込んで返す関数。

EOF (教 p.228)

getchar などが、入力の終わり（ に由来）に達した場合に返す値をマクロで EOF と書く。（stdio.h に定義されている。）この値は、通常の文字とは区別される。

リダイレクト (教 p.231) (教 p.231)

標準入出力をファイルへの入出力につなぎかえることで、C 言語ではなく OS (Unix, MS-DOSなど)の機能になる。

コマンド名 < ファイル名 — ファイルの内容をコマンドの標準入力に渡す
 コマンド名 > ファイル名 — コマンドの標準出力をファイルに書込む
 コマンド名 >> ファイル名 — コマンドの標準出力をファイルの最後に追加する形で書込む

文字 (教 p.232)

C 言語では文字は、単にその文字に与えられたコード（整数値）で表す。

ASCII コード表での文字コードの抜粋:

文字 **10 進** **16 進**

'0' 48 0x30

'A' 65 0x41

'a' 97 0x61

拡張表記 (教 p.234)

「\n」の他に、「\t」、「\a」、「\b」などいくつかの特殊文字を表す表記がある。特に、バックスラッシュ（円記号）そのものを表す時には「 」と書く。

8.2 プログラム例

ハノイの塔（再帰）(hanoi.c)

```

1  #include <stdio.h>
2
3  void move(int n, int a, int b) {
4      printf("ディスク%dを棒%dから棒%dへ\n", n, a, b);
5  }
6
7  /* n枚のディスクをaからbに移動する手順 */
8  void hanoi(int n, int a, int b, int c) {
9      if (n > 0) {
10         hanoi(n - 1, a, c, b);
11         move(n, a, b);
12         hanoi(n - 1, c, b, a);
13     }
14 }
15
16 int main(void) {
17     int n;
18     printf("円盤は何枚ですか? "); scanf("%d", &n);
19     hanoi(n, 1, 2, 3);
20     return 0;
21 }
22
```

樹の描画（再帰）(tree.c)

```

1  #include <stdio.h>
2  #include <math.h>
3
4  void drawTree(int d, double x, double y, double r, double t) {
5      /* d      --- 再帰の深さ、
6         (x, y) --- 枝の根元の座標、
7         r      --- 枝の長さ、
8         t      --- 枝の伸びる向き（ラジアン）*/
9      double r1;
10     if (d == 0) return; /* 打ち切り */
11
12     printf("%6.3f %6.3f %6.3f %6.3f\n",
13           x, y, x + r * cos(t), y + r * sin(t));
14     drawTree(d - 1, x + r * cos(t),
15             y + r * sin(t), 0.5 * r, t);
16     r1 = 0.5 * r;
17     drawTree(d - 1, x + r1 * cos(t),
18             y + r1 * sin(t), 0.5 * r, t + 3.1416 / 2);
19     drawTree(d - 1, x + r1 * cos(t),

```

```
20             y + r1 * sin(t), 0.5 * r, t - 3.1416 / 2);  
21     }  
22  
23     int main(void) {  
24         drawTree(6, 128, 255, 128, -3.1416 / 2);  
25         return 0;  
26     }  
27
```
