知らなきゃヤバイ

基礎ほど奥深〜い！ Cプログラミングの当たり前!?

第3回 goto文って使っちゃダメなの？ 構造化プログラミングのホントの意味合い

邑中 雅樹

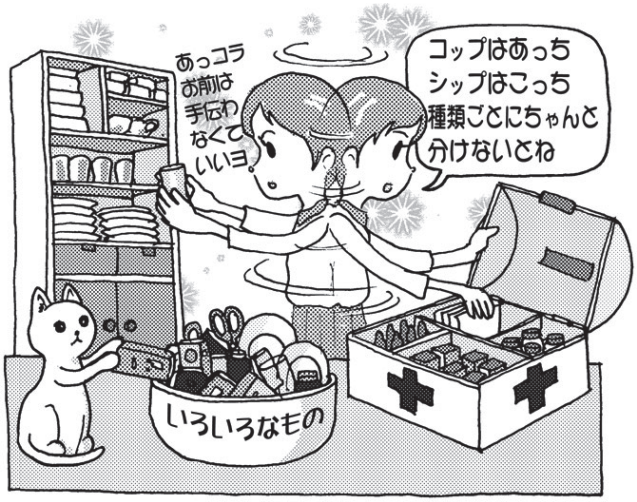

図1　棚や箱に整理すれば，パッと見やすくスッキリ！…構造化

図1のような構造化プログラミングについておさらいし，ホントの意味合いがわかればタブーのgoto文も使えます．

ソフトウェア・エンジニアリングを職業とする人であったとしても，コンピュータが理解できる機械語を直接理解する能力を持つのは困難です．そのために存在するのがプログラミング言語です．

すべてのプログラミング言語には，それぞれ独自の設計思想があり，考え方（パラダイム，**コラム1**）を持っています．

おさらい… バグの少ないCプログラムをつくるには

● 命令型プログラミング

C言語は，命令型（手続き型ともいう）プログラミング言語というパラダイムを採用しています．

命令型プログラミングは，“どのようにするのか”を積み上げることで“何をするのか”を定義します．そのソース・コードは，細かい命令の羅列となります．細かい命令はおおむね演算命令と制御構造命令の2種類に分かれます．

▶演算命令…四則演算，計算結果のメモリへの退避や復帰など

演算命令は，四則演算や比較演算，論理演算などです．これらに加えて，計算結果のメモリへの退避，メモリからの復帰なども演算とみなします．C言語ではポインタへの代入や参照でメモリ・アクセスするので，それらを演算とみなすことに違和感はないと思います．

▶制御構造命令…分岐や繰り返し，関数呼び出しなど

制御構造命令は，C言語では`if-else`，`for`，`while`，`goto`，関数呼び出しなどを指します．これらは，演算結果の論理値にしたがって次に行う演算を決定します．

`goto`や関数呼び出しには論理値判定はありませんが，「論理値が真でも偽でも指示した演算を次に行わせるもの」と考えれば，制御構造に含めることができます．

余談になりますが，プログラミング言語の中には，命令型でないパラダイムを採用しているものもあります．そのような言語では，制御構造を記述しないものもあります．静的検証など，組み込みシステムに無関係でない分野で用いられることがあるので，雑学として記憶に留めておくと良いかもしれません．本稿はC言語でのプログラミングの解説ですので，詳細は割愛します．

● 命令型プログラミングの制御構造

命令型プログラミングの本質を突き詰めてみましょう．

プログラミングは，突き詰めると抽象を操る営みです．一方，ソース・コードは具象的な存在です．ソース・コードとにらめっこしていても，本質は掴めません．

読者の皆様は，本稿をお読みになっているということは，書きかけのソース・コードが表示されているパソコンから，目を離しているでしょう．本稿もソース・コードから一旦離れて，プログラムの制御構造に目を向けてみましょう．

命令型のパラダイムに基づいたプログラムの状態遷移を**図2**に示します．演算命令を表すノード“○”と，演算終了