教科書には載っていない！

現場で役立つ プログラミングのちょい技

第11回 変数の初期化で痛い目に合わないためのちょい技

邑中 雅樹

プログラムで使う変数にはいくつかの種類があり，その種類によって実際のメモリへの割り当て方法が決められます．システムのさまざまな領域に割り当てられた変数を，どのように初期化しておくべきかについて，JIS規格での規定を参照しながら解説します．（編集部）

C言語における変数初期化の基本

● 初期値を定義/チェックする習慣でバグは減る

C言語で記述されたプログラムのコーディング・スタイルで必ず議論の的になるものの一つに，変数の初期化があります．変数の初期値について常に意識することで，未初期化変数の使用に起因するバグのほとんどは撲滅できます．もしすり抜けてしまうバグがあったとしても，その変数の値は，0やNULLポインタで初期化されています．初期化されている変数が引き起こすバグは，不定の値が引き起こすものよりも，はるかに再現性が高くなります．

最近のコンパイラはプログラマ以上（？）に賢くなりました．バグとなりそうな初期化漏れについては，コンパイル時に警告を出してくれます．しかしながら，コンパイラでの検出は万能ではありません．

C言語でプログラムを書いている方は暗黙的に理解しているはずなのですが，ISO C99仕様の和訳版であるJIS X 3010:2003規格を参照しながら，整理してみます．

図1 バグを減らすためのチェックの流れ

● 理解しておこう！ 自動変数とstatic静的変数で初期化処理が違う

C言語における変数は，いくつかの分類ができます．すぐに思いつくのはデータの型（int，floatなど）です．

また，記憶域期間という概念によって，静的変数と自動変数に分けられます．

ソース・コード上の表現でざっくり区分けするならば，関数内でstaticを付けないで宣言した変数は自動変数で，自動変数以外の変数は静的変数です．自動変数と静的変数で，その初期化の処理が異なっています．

■ 自動変数の初期化

● 油断大敵！ C言語の自動変数は初期値不定

C言語のJIS X 3010:2003規格では，自動変数の初期化について簡潔に定義されています．

> 「6.7.8 初期化（前略）
> 自動記憶域期間を持つオブジェクトを明示的に初期化しない場合，その値は不定の値を持つ．（後略）」

後ほど示すとおり，静的変数では初期化が行われますし，C言語以外の大多数の言語では自動変数であっても何らかの初期化が行われます．そのためC言語では，しばしば自動変数が初期化されるものと勘違いしたコードを書いてしまいがちです．

変数の値が不定である場合，プログラムは乱数的な挙動を示す可能性があります．これは再現の難しいバグの温床になりがちです．プログラムのレビューで自動変数の未初期化が見つかると，レビュアの表情が一気に険しくなるはずです．