≫ 文法の曖昧さを理解して確実性と再利用性を高める

マイコンC言語 転ばぬ先のつえ

第19回　静的変数領域の削減②…動的変数と静的変数の性質をプログラムで確認する

鹿取 祐二

表1　C言語の変数の分類…静的変数と動的変数（再掲）
局所変数と大域変数とは異なり，記憶クラスと密接に関わっている

宣言場所 / 記憶クラス	局所変数（関数，複文の先頭）	大域変数（関数外部）
なし	動的変数①	静的変数②
auto	動的変数⑤	文法違反
static	静的変数③	静的変数④
register	動的変数⑥	文法違反

本連載では，C言語の言語仕様（文法）の曖昧な部分と，それにより起こる問題を解説します．再利用性と効率が高く，安全かつ安心して使えるソフトウェアが開発できるようになることを目指します．

前回（本誌2022年12月号）から，静的変数領域の削減について紹介しています．「静的変数」の一言で，どの変数であるかが理解できていればよいのですが，宣言場所や記憶クラスも絡んでおり，多少複雑です．そこで，今後数回にわたり，静的変数の文法的な説明も含めて，その削減方法を紹介します．

（編集部）

動的変数と静的変数の違いをプログラムで見てみる

今回は，前回紹介した動的変数と静的変数の性質を実際にプログラム例で確認します．

前回と同様に，動的変数は**表1**の①，静的変数は**表1**の③で説明します．なお，**表1**の静的変数②，③，④はどれも同じ性質です．

● 確認1…番地は変わる？変わらない？

動的変数はプログラムの実行に応じて番地が変わることと，静的変数はプログラムを実行しても番地が変わらないことを確認します．

リスト1のプログラムでは，記憶クラスの指定がない動的変数aとstatic記憶クラスを指定した静的変数sを持つabc関数を異なる経路で呼び出しています．1つはmain関数から直接呼び出しています．もう1つはxyz関数を経由して呼び出しています．

リスト1　動的変数と静的変数の番地を確認するプログラム
動的変数はプログラムの実行に応じて番地が変わるが，静的変数はプログラムを実行しても番地が変わらないことを確認する

```
void abc(void)
{
        int a;
static int s;

    printf("&a = %p, &s = %p¥n", &a, &s );
}

void xyz(void)
{
    abc( );
}

void main(void)
{
    abc( );
    xyz( );
}
```

（a）ソースコード

```
&a = ac4, &s = 45c
&a = ac8, &s = 45c
```

（b）実行結果

プログラムの実行結果を**リスト1**（b）に示します．

動的変数aの番地は呼び出し経路によって変わりますが，静的変数sの番地は呼び出し経路に関係なく変わらないことが分かります．

もちろん，abc関数が表示している値は局所変数の番地なので，結果は対象のマイコンや処理系により異なります．あくまで1つの例と考えてください．

● 確認2…初期値の設定タイミング

動的変数と静的変数では，初期値の設定タイミングが異なります．動的変数は，それが宣言された複文や関数の実行時に初期化されます．静的変数はシステムの実行時に一度だけ初期化されます．このことを**リスト2**のプログラムで確認します．

abc関数で動的変数aと静的変数sを10で初期化して宣言し，main関数から4回呼び出します．abc関数は，呼び出されると動的変数aと静的変数sを共にインクリメントし，その値を表示します．

第13回　演算子④…算術演算子の正しい使い方（2022年3月号）
第14回　演算子⑤…加法演算子とシフト演算子（2022年4月号）
第15回　演算子⑥…関係演算子と等値演算子（2022年5月号）