≫ 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27回　スタック・サイズの算出③…実際に計算してみる

鹿取 祐二

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．

第25回（2023年10月号）からスタック領域の算出方法を紹介しています．スタック領域は，プログラム作成時に確保したサイズ以上に使用することが許されません．もし確保したサイズを超えると，プログラムが正常に動作せず，思わぬ事故を招くことがあります．そのため，確保したサイズ内に収まっているのかを事前に検証する必要があります．

今回は，具体的なスタック・サイズの計算方法を紹介します．（編集部）

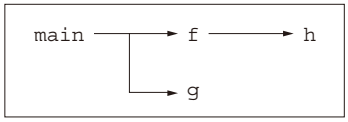

図1　スタック・サイズ計算の必須アイテム…関数呼び出し経路図
この例では，main関数はf関数とg関数を呼び出し，f関数はh関数を呼び出すことを意味している

リスト1　図1の関数呼び出し関係のみをC言語のソースコードで示すとこうなる

```
void main(void)
{
    f( );
    g( );
}

void f(void)
{
    h( );
}

void g(void)
{

}

void h(void)
{

}
```

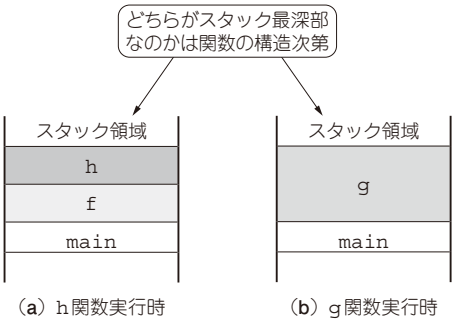

図2　スタック最深部（スタック領域を最も使用する関数呼び出し経路）**は関数のネストが最大の経路とは限らない**

45 スタック・サイズの計算方法

● 必須アイテム①…関数呼び出し経路図

▶どの関数がどの関数を呼び出すのかを表現

スタック・サイズを計算するには，自身が作成したシステムの関数呼び出し経路図が必要です．

図1に示すのは，どの関数がどの関数を呼び出すのかを表現した関数呼び出し経路です．**図1**の場合，main関数はf関数とg関数を呼び出し，f関数はh関数を呼び出すことを意味しています．この経路図における関数の呼び出し関係だけを示すと，**リスト1**のようになります．

▶全経路のスタック使用量を調べて比較する

関数呼び出し経路図を作成する理由は，関数のネストが最大の経路が必ずしもスタックの最深部になるとは限らないからです．

前述した通り，スタック領域は関数を呼び出すと使用されます．ただし，そのサイズは関数ごとに異なります．当然ながら，引数や局所変数が多かったり，関数の処理内容が複雑だったりすると，サイズは大きくなります．**図2**のように，main関数からf関数→h関数を呼び出した経路と，main関数からg関数を呼び出した経路では，どちらの最深部がより多くスタック領域を使っているかは分かりません．

もちろん，関数呼び出しネストが最大の経路がスタックの最深部となる可能性は非常に高いです．しかし，絶対ではありません．こればかりは，全ての関数

第15回　演算子⑥…関係演算子と等値演算子（2022年5月号）
第16回　演算子⑦…論理演算子（2022年6月号）
第17回　演算子⑧…条件演算子とカンマ演算子（2022年10月号）