ご購入はこちら

≫ 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2回　整数型②…表現できない値を代入したときの規則

鹿取 祐二

C言語は，誕生から50年近く経つ今でも，分野を問わず，さまざまな場面で使われるプログラミング言語です．言語仕様（文法）は標準規格化されていますが，曖昧な部分が多く存在します．

本連載では，C言語の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．移植性と効率が高く，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1回～第3回では，整数型の文法に対する問題点や内容を解説します．〈編集部〉

3 代入の規則は代入先の型で変わる

● 文法

▶代入先の型では表現できない値を代入すると結果が変わる

整数型の代入の規則は，代入先の変数が符号付き型なのか符号なし型なのかで異なります．

代入元の値が代入先の変数でも表現できる値であれば，どちらでも値は変化しません．一方，代入元の値が代入先の変数では表現できない場合，代入先が符号付き型なのか符号なし型なのかで結果が変化します．文法では，代入先が符号付き型の場合は不定，代入先が符号なし型の場合は代入先の型の最大値+1のモジュロ（剰余）となっています．

▶符号付き型だと不定，符号なし型だと最大値+1の剰余になる

これだけでは意味が分かりにくいと思うので補足します．本連載の第1回（本誌2021年1月号）でも解説しましたが，C言語は符号付き整数型の負の内部表現を決めていません．このことから，代入先が符号付き型の場合，表現できない値を代入するとどうなるか分からないため，結果は不定という扱いになります．

これに対し，代入先が符号なしの型の場合，内部表現は2のべき乗と決まっているので，結果は確定します．もし，表現できない値を代入すると，代入先の型の最大値+1によるモジュロ（剰余演算）が結果となります．簡単に言えば，上位ビット切り捨てと同じです．

リスト1　整数型の代入規則（int型を16ビット，long型を32ビットとする処理系で動作させた場合）

int型は32767までしか代入できないのに対し，100000を代入している．文法上は結果が不定になるが，実は上位ビットが切り捨てられているだけだ．変数yは2の補数表現，変数zは2のべき乗で見ているだけで，どちらも16進数で表現すると0x86A0になる

```
long x = 100000; // long型は32ビットとする
int y;           // int型は16ビットの-32767～+32767とする
unsigned int z;
                 // unsigned int型は16ビットの0～65535とする

    y = x;     // 100000はint型では表現できないので不定となる
    z = x;     // 100000%(65535+1)の34464となる

    printf("x = %ld, y = %d, z = %u", x, y, z );
```

（a）ソースコード

```
x = 100000, y = -31072, z = 34464
```

（b）実行結果

リスト1に示すのは，整数型同士で代入を行った例です．ここではint型を16ビット，long型を32ビットとする処理系で動作させたとします．

変数xはlong型なので，初期値の100000（10万）は表現可能です．これを符号付きのint型の変数yに代入するとどうなるでしょうか．int型が16ビットの場合，最大値は32767なので，代入元の100000は表現できません．文法上の結果は不定となります．

変数xの値を符号なしのunsigned int型の変数zに代入するとどうなるでしょうか．int型が16ビットの場合，文法ではunsigned int型も同じ16ビットであり，その最大値は65535です．unsigned int型でも代入元の100000は表現できませんが，結果は確定します．

unsigned int型の最大値は65535なので，100000を最大値＋1の65536で除算した余りである34464が変数zに格納されます．

この結果は，int型が16ビットである限り不変です．