≫ 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6回　派生型①…難しい配列やポインタをtypedefで分かりやすくする

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．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6回からは派生型，いわゆる基本型から派生した配列，ポインタ，構造体，共用体，ビットフィールドについて紹介します．各派生型が移植性を重視した組み込み系システムで利用できるものなのかを理解しましょう．〈編集部〉

● 今回解説すること…typedefを使って配列やポインタを分かりやすくする

C言語で一番難しいのはポインタ型です．それは間違いないと思いますが，難しいポインタでC言語の理解度を判断するのは誤りです．筆者はポインタよりも整数型や浮動小数点数型の算術変換の方が重要だと思っています．極端な話，ポインタを使わなくてもプログラムは作成できますが，算術変換を知らないと正しい数値計算が行えません．筆者の個人的な意見ですが，組み込みシステムの開発では構造体を指すポインタが理解できていれば十分です．次に示すような，ポインタの配列，ポインタを指すポインタ，2次元配列を指すポインタなどは，組み込みシステム開発では必要ないと思います．

```
int *a[5];      // ポインタの配列
int **pa;       // ポインタを指すポインタ
int (*pb)[5]; // 2次元配列を指すポインタ
```

本連載では難しいポインタの宣言や使い方は解説しません．これらのポインタの解説は，既に多く出版されている書籍に譲ります．その代わりに，本稿ではtypedefを使って難しい配列やポインタを分かりやすくする方法を紹介します．

● 識別子の置き換えを行うキーワード「typedef」

▶使い方

typedefは，識別子の置き換えを行うキーワードで，#defineと少しだけ似ています．例えば，次のように宣言した場合で考えてみます．

```
typedef int seisu;
```

太字部分が置き換え対象，下線部が置き換え後の識別子です．このように宣言すると，以降の処理ではseisuと記述すればint型を意味することになります．

```
seisu a, b;
```

このように宣言すれば，aとbは共にint型の変数になります．特に難しいこともなく，ここまでであれば#defineとあまり変わらないという印象だと思います．

▶機能1：ポインタ型の名前の置き換え

しかし，ここからがtypedefならではの機能です．typedefは，正確には型の名前を置き換えるキーワードです．型であれば，何でも識別子と置き換えられます．例えば，次のように宣言した場合で考えてみます．

```
typedef int *point;
```

このように宣言すると，int型を指すポインタ型に対して，pointという型名を付けたことになります．間違えないようにしましょう．スペースがあると，誰しもそこで区切りたくなるのですが，typedefは型の名前の置き換えであり，int *までが型なので，pointはint型を指すポインタ型になります．

```
point a, b;
```

このように宣言すれば，aとbは共にint型を指すポインタ型の変数となります．*がないのにbもポインタ変数になります．

▶機能2：配列型の名前の置き換え

それではもう1つ見ておきましょう．

```
typedef int array[10];
```

このように宣言すると，int型10個の配列型に対して，arrayという型名を付けたことになります．int [10]が型なので，arrayはint型10個の配列型です．そのため，次のように宣言すれば，aとb

第1回　整数型①…値の範囲と負の内部表現（2021年1月号）
第2回　整数型②…表現できない値を代入したときの規則（2021年2月号）
第3回　整数型③…整数の格上げと算術変換規則（2021年3月号）