マイコンを正しく操縦するための作法

基礎から学ぶC言語講座

岡田 好一
Yoshikazu Okada

第4回 構造体，共用体，ビットフィールドを攻略しよう

複数のデータを集めた複合データ型「構造体」

構造体(structure)は一体としてふるまう複数のデータを集めた複合データ型です．例えば，書籍なら「著者」「書名」「出版社」「発行年」「ISBN」「価格」などとなるでしょう．

座標などは配列が便利なこともありますが，通常は，点「x」「y」，直線「x0」「y0」「x1」「y1」(始点と終点)などと構造体にします．一体で扱われるからです．

● 構造体はデータベースか？

ところで，上の例で書籍の著者が複数ならどうするのだ，という突っ込みが入りそうです．これは意外に本質を突いた疑問(多対多の対応解決)なので，簡単に述べておきましょう．

私の使っているデータベース・システムでは「アリス；ボブ；イブ」などと区切り文字で区切ればよいので問題になりませんし，参照方向も明白です．

しかし，構造体では困ってしまいます．しかたがないので通常は「著者1」「著者2」「著者3」などと多めに項目を取っておきます．

構造体には要素数不定の配列という書きかたが用意されており，OSなどではよく使われます．作るにも使うにも少々コツが要求されるとはいえ，いざ使う段になると使いかたは明白なので，本連載では省略し，形のみを紹介します(**リスト4-1**)．任意個の領域確保には標準ライブラリ関数malloc を使います．

構造体の宣言

それぞれ座標(x, y)をもつ点aと点bを構造体で表現してみましょう(**図4-1**)．

構造体は宣言で，

```
struct point {int x; int y;};
```

としておき，変数の定義は，

```
struct point a, b;
```

とします．名前pointは構造体タグと呼ばれます．xとyは構造体「struct point」のメンバと呼ばれます．

リスト4-1 要素数不定の配列
標準ライブラリ関数mallocとともに使う

```
struct ptable {
   unsigned int psmax;
   unsigned int psn;
   Point ps[1];  // c99ではps[]と書ける
};
```

Keyword 1 構造体，ビット操作命令

構造体：データがメモリ上にまとまって並んでいるだけなのに「構造」とは大げさに思えます．おそらく，意味的な相互依存関係があるための命名でしょう．

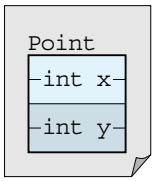
各種計算機言語に同じようなものはあるので，プログラマが考えやすい単位と言えます．

C++ではオブジェクトそのものなので，個々の構造体自体が存在，各メンバは属性，つまり重さや色や名前などの，事物に与えられた性質と考えれば意義がわかりやすいでしょう．

ビット操作命令：R8Cには1ビット単位でANDやORなどの論理操作をする一群の命令が用意されています．いかにも組み込みマイコンらしい命令群です．

R8C純正のコンパイラは1ビットのビットフィールドに対して可能であればビット命令を生成します．

特に，スペシャル・ファンクション・レジスタに対しては意義が大きいです．上述したC言語の_Bool型のふるまいとR8Cのビット命令には整合性があります．



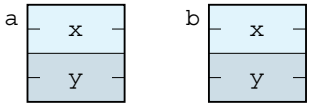


図4－1　Point型の宣言と変数a,bの宣言
座標(x, y)をもつ点aと点bを構造体で表現

● typedefの活用

上記の書きかたでまったく問題は起こらないのですが，いきなりtypedef指定子を使って，

```
typedef struct point {
    int x;
    int y;
} Point;
```

と前準備しておき，

```
Point a, b;
```

と変数定義することが，むしろ多いと思います．typedefはマクロ定義の一種で，名前Pointを型名のように使うためのしかけです．

データの扱いかたはtypedefを使っても使わなくても同じです．

```
a.x = 3;
a.y = 4;
```

などとなります．つまり，a.xとa.yがint型の変数として使えます．

構造体の代入は一気に行うことができます．

```
b = a;
```

で，メンバxもメンバyもコピーされます．

リスト4－2　構造体のプログラム例
「書籍」構造体．メンバは「著者」「書名」「出版社」「発行年」「ISBN」「価格」

```
typedef struct shoseki {
   char author[30];
   char bname[40];
   long pubix;
   int pubyr;
   char isbn[10];
   long price;
} Shoseki;
```

● 構造体のメンバ

構造体のメンバは型が異なっていてもかまいません．これは，同じ型しか許されない配列と異なる点です．文字列(charの配列)やポインタのみならず，ほかの構造体が入ることも珍しくありません．

例えば，例示した「書籍」構造体を作ってみましょう．メンバは「著者」「書名」「出版社」「発行年」「ISBN」「価格」ですから，**リスト4－2**のようになります．

構造体は使うことも作ることも多いので，具体的な使用例はそのつど説明することになるでしょう．以下では簡単のために，上で作ったPoint構造体を例として用います．

● 構造体の配列

構造体は配列の要素としても多用されます(**図4－2**)．例えば，4点なら，

```
Point sq1[4];
```

などです．メンバにアクセスするには，

```
sq1[1].x = 12; sq1[0].y = 13;
```

などとします．

このように，素直な配列の書きかたで押し通せる場合が多いと思います．しかし，ポインタ経由のメンバ指定の書きかたが特別に用意されています．

```
Point sq1[4], *sq;
int y;
```

と宣言しておいて，

Keyword 2　1件のデータ

本稿では表の1行や，ファイル内ではレコード(record)と呼ばれる単位を指します．

「1件のデータ」(datum)をどのように構成するかは，時として困難を伴います．そうした場合は，データベースや統計など，データ処理の専門家に相談するとヒントが得られるかもしれません．

統計なら統計処理技法，データベースには集計法と検索法があり，処理技法は限られていますから，要請されるデータ形式の範囲が決まります．単純なデータ処理と考えられる場合でも，通常は対応するアルゴリズムによるデータ形式の要請があります．

パンフレットやマニュアルの表などを見ていると，項目が細分化されたり，途中で欄が結合されたりと，結構複雑な構成になっています．構造体の中の構造体や共用体が連想されます．