第6章

C言語理解の鬼門？　これを使いこなせれば怖いものなし

最強の武器，ポインタの使い方を身につけよう

ポインタはC言語を習得する上での難関の一つとされている．しかし実際にプログラムを書いて，動作を確認しながらポインタの挙動を観察していけば，ポインタを使ったプログラムがどのように動き，何のために必要なのかを目視しながら確認できる．ここでは，ポインタの動作を確認して，その概念について学習する．（編集部）

岡崎 光隆

1．そもそも，ポインタってどんな意味？

ポインタ，という言葉を聞くと読者の皆さんは何を思い出すでしょうか．筆者は，コンピュータ用語の「ポインタ」より先に，まずポインタという種類の犬を想像してしまいます．

ポインタという犬の名前の意味をご存じでしょうか？ポインタは，狩りで獲物を見つけたら，その場所を主人に「指し示す」姿勢をとるように訓練された猟犬です．「指し示す」動作を英語ではポインティング（pointing）といい，指し示す動作をする主体をポインタ（pointer）といいます．というわけで，ポインタという犬種の名前には，獲物の位置を「指し示す者」という意味があります．

さて，C言語のポインタという用語も，実はポインタ犬と同じで，「指し示す者（物）」という意味で使われます．ポインタ犬の役目は，獲物を指し示すことでしたが，C言語のポインタの役目は，コンピュータのメモリ上に格納されたデータを指し示すことです（**図1**）．

2．C言語の変数とメモリの関係

プログラムが動作するとき，プログラムが扱うデータはメモリに格納されます．しかし，実際にメモリ上にデータが格納されているようすを見たことのある方は少ないと思います．データがどのようにメモリに入っているか知っていると，ポインタの概念はぐっとわかりやすくなります．そこで，ポインタの説明に入る前に，プログラムのデータがメモリに格納されているようすを見てみましょう．**リスト1**のプログラムを見てください．このプログラムは4個の変数を定義して，その変数のメモリ上の格納場所を表示するプログラムです．

図2は**リスト1**のプログラムをIAR Embedded Workbench上でデバッグ実行したようすです．プログラムが完全に終了すると，メモリの内容が消えてしまうので，`main`

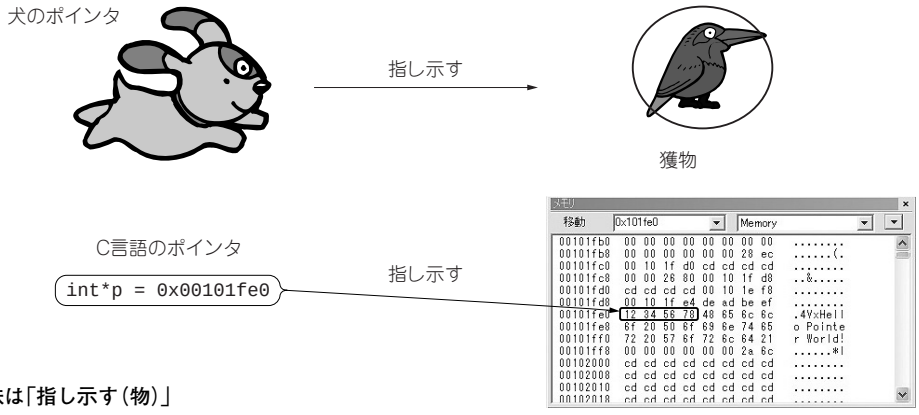

図1
ポインタの意味は「指し示す（物）」

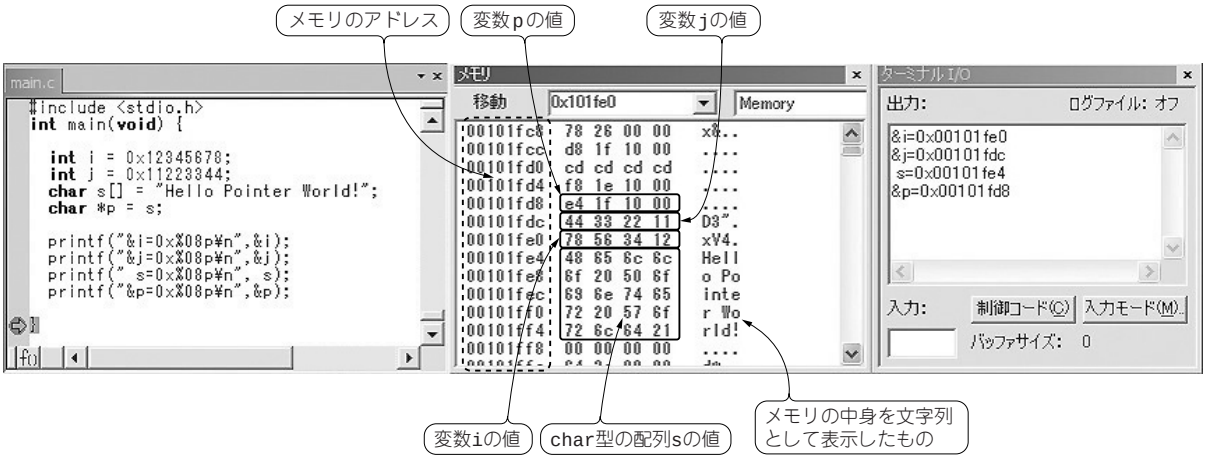

図2　変数の格納位置を表示するプログラムの実行結果

リスト1　変数の格納位置を表示するプログラム

```
#include <stdio.h>
int main(void) {

  int i = 0x12345678;
  int j = 0x11223344;
  char s[] = "Hello Pointer World!";
  char *p = s;

  printf("&i=0x%08p¥n",&i);
  printf("&j=0x%08p¥n",&j);
  printf(" s=0x%08p¥n", s);
  printf("&p=0x%08p¥n",&p);

  return 0;
}
```

%08pはprintf関数の書式指定．引き数をメモリ上のアドレスと見なして16進8けたで表示する

変数に&を付けると，その変数をメモリ上に格納しているアドレスを取得できる

配列変数の場合，配列の名前がそのまま配列が格納されているメモリの先頭要素のアドレスを表す

リスト2　ポインタを使ったプログラム例

```
int main(void) {
  int foo=0x12345678;
  int *p;
  p=&foo;
  printf("gfoo=%x¥n",foo);
  *p=0x11111111;
  printf("gfoo=%x¥n",foo);
  return 0;
}
```

普通の変数fooを宣言．初期値は0x12345678

これがポインタ変数だ！

変数fooの値を16進数で表示

変数fooの値を16進数で表示

関数の最後でプログラムを一時停止させた状態にしてあります．中央の「メモリ」ウィンドウには，このプログラムが使っているメモリの内容が表示されています．「メモリ」ウィンドウの中央に出ている78 26 00 00などの数値がメモリの内容を表しています．

一方，「メモリ」ウィンドウの左側に出ている8けたの数字がメモリの場所を表す番号です．この番号のことをメモリのアドレスといいます．右側の「ターミナルI/O」ウィンドウにはプログラムのprintf関数から出力された結果が表示されています．この出力結果は，変数i，j，s，pが格納されているメモリ上のアドレスを示しています．「メモリ」ウィンドの画面でそれぞれのアドレスのメモリを確認してみましょう．確かにそれらの変数らしき値が入っています．例えば変数iのアドレス00101fe0には78 56 34 12という値が入っています．これは変数iの値0x12345678がリトル・エンディアン方式（第5章のコラム参照）でメモリに格納されている状態です．

3. ポインタの基本を理解しよう

ここでは，ポインタを使った単純なプログラムの動作を調べながら，プログラム中でのポインタの働きを説明します．**リスト2**のプログラムを見てください．

まだ3，4，6行目に登場する記号「*」と「&」の意味がよくわからないかもしれませんが，後ほど説明します．とりあえずこのプログラムを実行して動作を見てみましょう．このプログラムをコンパイルして，IAR Embedded Workbenchでデバッグを実行すると，**図3**の出力結果が得られます．

ちょっと不思議なことが起きているのがわかるでしょうか．**リスト2**のプログラムの6行目では，*pという変数

```
foo=0x12345678
foo=0x11111111
```

図3　ポインタを使ったプログラム例の実行結果