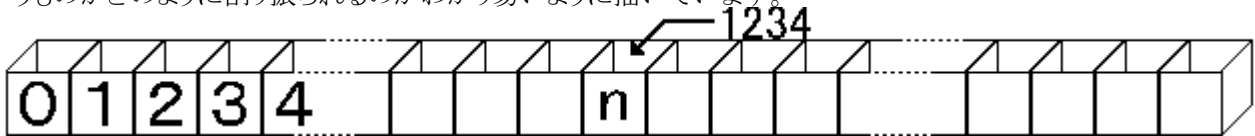


第7回 ポインタ

今回はポインタについて学んでいきます。ポインタは関数をまたいで変数の値を操作する際に用います。ポインタはC言語において最も特徴的な部分であり、重要な部分でもあります。少しややこしいかもしれませんが、ソフトゼミAの最終回、張り切っていきましょう。

○ポインタの宣言

ポインタとは、変数のアドレスを記憶する変数です。変数はメモリ上のどこかに記憶され、そしてその場所を表す数値がアドレスです。メモリを大量に並んだ箱、アドレスはその箱が何番目の箱かの番号と考えるとわかり易いかもしれません。ただし実際のメモリにおいても仕切りがあるわけではありません。下図ではあくまでもアドレス、というものがどのように割り振られるのかわかり易いように描いています。



上図をプログラムで表すと、下記ようになります。

```
int num = 1234 ;
```

このとき変数「num」のアドレスが「n」にあたります。nの値はプログラムが実行されることに異なります(必ず異なるわけではありません)。

ポインタ自体も変数(ポインタ変数)なので、まず宣言する必要があります。ポインタの宣言は次のようになります。

```
//データ型 変数名; または データ型* 変数名;  
int *p;
```

「*」は間接参照演算子と呼ばれます。「*」が付いた以外は違いはありません。また、その位置は「データ型」の直後でも、「変数名」の直前でも構いません。プログラマーが好きなほうに統一するとよいでしょう。「int* p;」を「イントポインター型」の変数「p」を宣言している、と考ええると考えやすいかもしれません。

ポインタ変数はあくまでもポインタなので、代入する値は正しい型の変数のアドレスである必要があります。ポインタに与えるアドレスは、変数や関数のアドレスになります(関数は難しいのでここでは触れません)。アドレスを定数で指定することはありません。メモリ上のどのアドレスに何があるかはわからないからです。

実際に上記のプログラムの続きとして変数「num」のアドレスをポインタ変数「p」に代入すると以下ようになります。

```
p = &num ;
```

このように、変数名の直前に「&」を付けると、その変数のアドレスを意味します。また、「&」はアドレス演算子と呼ばれます。

○アドレスの代入と間接参照

さて、アドレスを保持しているだけでは意味がありません。ポインタの具体的な使い方を学んでいきましょう。ポインタとはその名のとおり、「どこかを指し示すもの」です。変数のアドレスを保持している場合、「どこか」とは「何らかの変数」という意味になります。つまり、ポインタを通じて変数にアクセスできるということです。これを**間接参照(逆参照)**と呼びます。これを用いて、先ほどの変数の初期化は次のように置き換えることができます。

```
int num ;  
int *p = &num ;  
*p = 0 ;
```

ポインタ変数名の直前に「*」を付けると、そのポインタが保持しているアドレスを間接参照します。p から num を間接参照し、num に 0 を代入しています。直接 num に触れていないように見えますが、printf 関数などで表示すると「100」になっています。ただし、この場合は「num = 0 ;」と書けばいいだけです。あまり意味のないプログラムといえるでしょう。上記のプログラムを main 関数に入れ、これに下記のプログラムを追加し、実行してみます。

```
printf( "num = %d\n", num ) ;  
printf( "p = %d\n", p ) ;  
return (0) ;
```

すると、num が 0 で初期化されていることがわかりますが、p の値がわけのわからない値であることがわかります。このことから、メモリ上のどのアドレスに変数があるかは実行するまでわからないことが裏づけできます。

では、ここでポインタを用いることでどのようなことができるのかを紹介しましょう。

```
void a( int *p){  
    *p = 0 ;  
    return ;  
}  
int main( void ){  
    int num ;  
    a( &num ) ;  
    printf( "num = %d", num ) ;  
    return (0) ;  
}
```

こうすることで、関数「a」の中で、関数「main」内のローカル変数「num」の値を書き換えることができたことが確認できます。これは今までどおりの変数の使い方だけではできないことです。最初に述べた「関数をまたいで変数の値を操作する」ということです。

○練習問題

二つの int 型の変数のアドレスを受け取り、その変数の値を交換する関数「swap(int *a , int *b)」を作り、main 関数ないで実行し、printf でその動作を確認してください。