構造体とオブジェクト指向

平成 27 年 9 月 29 日

1 構造体で簡単なプログラム

構造体を使ってプログラムを書く場合には、構造体を処理するための補助関数を作ると、プログラムが読みやすくなる。

```
struct Person {        // Person 構造体
   char  name[ 10 ] ; // 名前
   int   age ;        // 年齢
} ;
void set_Person( struct Person *p , char s[] , int a ) {
   strcpy( p->name , s ) ;
   p->age = a ;
}
void print_Person( struct Person *p ) {
   printf( "%s %d\n"" , p->name , p->age ) ;
}
int scan_Person( struct Person *p ) {
   return scanf( "%s%d" , p->name , &(p->age) ) ;
}
void main() {
   struct Person tsaitoh ;
   struct Person table[ 10 ] ;
   int i ;

   // 構造体を初期化して表示
   set_Person( &tsaitoh , "t-saitoh" , 50 ) ;
   print_Person( &tsaitoh ) ;

   // 構造体を 10 件分読み込んで表示
   for( i = 0 ; i < 10 ; i++ ) {
      if ( scan_Person( &(table[ i ]) ) != 2 )
         break ;
      print_Person( &table[ i ] ) ;
   }
}
```

例えば、上記のプログラムでは、Person は、名前と年齢の構造体、print_Person() は出力、scan_Person() は入力と読めば、main() の中の処理は、構造体の文法に慣れる必要はあるが、print_Person(),scan_Person() の中身を知らなくても、tsaitoh を初期化して表示、10 件のデータを入力して表示... と大体の処理が把握できる。

特に、xxx_Person() が、Person に対する命令とイメージすると、「tsaitoh を初期化しろ」、「table[i] に入力しろ、出力しろ」とデータに命令しているように見える。

このような、データに対して命令するイメージでプログラムを記述する方法は、オブジェクト指向プログラミング (Object Oriented Programming) と呼ばれる。

特に、オブジェクト指向では、データ構造の宣言 と、データに対する命令 をペアに記述する。
オブジェクト指向プログラミングは最近の主流であり、C++,Java などの最近のプログラム言語であれば、記述方法の中心となっている。

2 構造体からクラスへ

オブジェクト指向の C++では、前述のプログラムは、以下のように記述できる。

```
class Person {       // Person 構造体
private:
   char  name[ 10 ] ; // 名前
   int   age ;        // 年齢
public:
   Person( char s[] , int a ) { // コンストラクタ
      strcpy( name , s ) ;      // 初期化関数
      age = a ;
   }
   void print() {
      printf( "%s %d\n"" , name , age ) ;
   }
   int scan() {
      return scanf( "%s%d" , name , &age ) ;
   }
} ;
void main() {
   // クラスを初期化して表示
   Person tsaitoh( "t-saitoh" , 50 ) ;
   tsaitoh.print() ;

   // 構造体を 10 件分読み込んで表示
   Person table[ 10 ] ;
   for( int i = 0 ; i < 10 ; i++ ) {
      if ( table[i].scan() != 2 )
         break ;
      table[i].print() ;
   }
}
```

クラス (class) は構造体struct の拡張であり、この中にデータ (要素) と、データを扱う関数 (オブジェクト指向では メソッド と呼ぶ) を記述する。メソッドを使うときは、対象となるデータの後ろに、関数 (メソッド) を記述する。
クラス変数. メソッド (実引数 , ...) ;
メソッドの呼び出しでは対象となるデータが明記されるため、メソッドの宣言の引数には、構造体へのポインタなどが不要となる。メソッドの中では、クラス変数をそのまま記述すればよい。
クラス名を型として使う際には、構造体のようにstruct を前につける必要はない。クラス名の後ろに変数名を記述すれば、そのデータ構造の実際の値を格納する入れ物が作られる。具体的な値が割り当てられた入れ物は、オブジェクト と呼ばれる。そして、オブジェクト指向プログラミングでは、オブジェクトに対して命令するスタイルでプログラムを記述する。
これは、データを擬人化しているようなイメージで考えるとわかりやすい。
このような記述方法をとると、プログラムを複数人で開発するときに、クラスを設計する人と、そのクラスを用いてプログラムを作る人に分業してプログラムを作ることができる。プログラムでおかしい点があれば、クラスの中の間違いなのか、そのデータの使う方の間違いなのか、責任分担が明確になる。