SADにおける

クラスの使用例

正確な記述および詳細は森田さんのSAD School の資料を参照のこと。

大西幸喜

2012.01.25

オブジェクト

- オブジェクト指向プログラミングで使われる
- ソフトウエアが扱おうとしている現実世界に存在する物理的あるいは抽象的な実体を、属性（データ）と操作（メソッド）の集合としてモデル化しコンピュータ上に再現したもの。
- オブジェクトを定義するモデルは「クラス」と呼ばれる。

インスタンス

- オブジェクト指向プログラミングで使われる
- クラスを基にした実際の値としてのデータ
- クラスを「型」、インスタンスを「実体」と説明されることもある。
- 「オブジェクト」とほぼ同義語、メモリ上に配置されたデータ集合という意味合いが強い。
- 例：「名前」、「身長」、「体重」というクラスがあるとすれば、そのインスタンスは、田中、175、65というように作られる。

クラスの定義

例：３次元ベクトルのクラス

```
Vector3=Class[{},{},
 {
   m$x=0,
   m$y=0,
   m$z=0
   },
```

メンバー変数（メンバーシンボル）

変数を区別するためにm$を先頭につけている。

```
Constructor[x_,y_,z_]:=(
  m$x=x;
  m$y=y;
  m$z=z;
  );
```

クラスのインスタンスが作られたときに一度だけ自動的に呼ばれる。

```
show[]:=(
  Print["1:  x =",m$x];
  Print["2:  y =",m$y];
  Print["3:  z =",m$z];
  );
```

メンバー関数（メソッド）

次ページへつづく

```
(* norm *)
  norm[]:=Sqrt[m$x^2+m$y^2+m$z^2];

(* add *)
  add[v_]:=(
    Vector3[m$x+v@m$x,m$y+v@m$y,m$z+v@m$z]
    );
(* subtract *)
  sub[v_]:=(
    Vector3[m$x-v@m$x,m$y-v@m$y,m$z-v@m$z]
    );
(* scalar product *)
  spro[v_]:=(
    m$x*v@m$x+m$y*v@m$y+m$z*v@m$z
    );
 (* cross product *)
  cpro[v_]:=(
    Vector3[m$y*v@m$z-m$z*v@m$y,m$z*v@m$x-m$x*v@m$z,
       m$x*v@m$y-m$y*v@m$x]
    );
 (* scale *)
  scale[fac_:1]:=(
    Vector3[fac*m$x,fac*m$y,fac*m$z]
    );
  ];
```

実行例

```
v1=Vector3[1,2,3];
```
Vector3クラスを用いてv1というインスタンスを作成。

```
v2=Vector3[4,5,6];
```
Vector3クラスを用いてv2というインスタンスを作成。

```
v3=v1@add[v2];
```
v1のadd[]というメソッドをv2に作用。

```
v4=v1@sub[v2];

v5=v1@cpro[v2];

v6=v1@scale[-1];

v3@show[];

v4@show[];

v5@show[];

v6@show[];

Print[v1@spro[v2]];
```

```
1:  x =5
2:  y =7
3:  z =9
 v4@show[];
1:  x =-3
2:  y =-3
3:  z =-3
 v5@show[];
1:  x =-3
2:  y =6
3:  z =-3
```

```
v6@show[];
1:  x =-1
2:  y =-2
3:  z =-3
 Print[v1@spro[v2]];
32
```

クラスの継承

1. 大きなクラスが、小さなクラスを継承する。
    - 自動車クラスが、タイヤクラスやエンジンクラスを継承。
    - ４元ベクトルが、３元ベクトルを継承。

2. 特殊なクラスが 、一般的なクラスを継承する。
    - 猫クラスが動物クラスを継承。
    - 四極磁石クラスが、磁石クラスを継承。

クラスの定義

例：４元ベクトルのクラス

```
Vector4=Class[
  {Vector3},
  {},
  {
    m$ct=0
    },
```

←――― スーパークラス

←――― メンバー変数

```
  Constructor[ct_,x_,y_,z_]:=(
    m$ct=ct;
    Vector3`Constructor[x,y,z];
    );

  show[]:=(
    Print["0: ct =",m$ct];
    Vector3`show[];
    );
```

次ページへつづく

```
  norm[]:=(
!   Sqrt[m$ct^2-(Vector3`norm[])^2]  (* Alternative *)
    Sqrt[spro[This]]
    );

  add[v_]:=Module[{vv},
    vv=Vector3`add[v];
    Vector4[m$ct+v@m$ct,vv@m$x,vv@m$y,vv@m$z]
    ];

  sub[v_]:=Module[{vv},
    vv=Vector3`sub[v];
    Vector4[m$ct-v@m$ct,vv@m$x,vv@m$y,vv@m$z]
    ];

  spro[v_]:=(
    Prin[Vector3`spro[v]];
    m$ct*v@m$ct-Vector3`spro[v]
    );

];
```

実行例

４元運動量の例

```
e1=3.5;
px1=py1=0;
pz1=-3.5;

e2=8.0;
px2=py2=0;
pz2=8.0;

p1=Vector4[e1,px1,py1,pz1];
p2=Vector4[e2,px2,py2,pz2];
```

エネルギーと運動量を設定

```
p12=p1@add[p2];
p12@show[];

Print["E_CM =",p12@norm[]," GeV/c^2"];

Print["E_CM =",Sqrt[p12@spro[p12]]," GeV/c^2  (use spro[])"];
```

重心系エネルギーを計算

磁石クラスの定義

```
Magnet=Class[
  {},
  {},
  {
    (* 2:Bend 4:Quad 6: Sext *)
    m$name="",
    m$type=2,
    m$k=Table[0,{22}],
    m$sk=Table[0,{22}],
    m$x=0,
    m$y=0,
    m$q=0,

    m$dk=Table[0,{22}],
    m$dsk=Table[0,{22}],
    m$dx=0,
    m$dy=0,
    m$dq=0
    },
```

次ページへつづく

磁石クラスのメソッド

```
Constructor[nm_]:=(
  m$name=nm;
  m$type=LINE["TYPE",nm];
  m$k=(LINE["K"//#,nm])&/@Range[0,21];
  m$sk=(LINE["K"//#,nm])&/@Range[0,21];
  m$x=LINE["DX",nm];
  m$y=LINE["DY",nm];
  m$q=LINE["ROTATE",nm];
  );

SetErrorK[id_,ns_:""]:=(
  LINE[ns//"K"//id,m$name]=m$k[[id+1]]+m$dk[[id+1]];
  FFS["CALC NOEXPAND"];
  );

ResetErrorK[id_,ns_L""]:=(
  LINE[ns//"K"//id,m$name]=m$k[[id+1]];
  FFS["CALC NOEXPAND"];
  );

];
```

四極磁石クラスの定義

```
Quad=Class[
  {Magnet},
  {},
  {
    m$dk1=0
    },
  Constructor[nm_]:=(
    Magnet`Constructor[nm];
    );

  K1[]:=(LINE["K1",m$name]);

  SetErrorK[]:=(
    m$dk[[2]]=m$dk1;
    Magnet`SetErrorK[1];
    );

  ];
```

使用例

```
q=Quad["QF1"];
Print[q@m$name," ",q@K1[]];

q@m$dk1=q@K1[]*2e-2;
q@SetErrorK[];
```

```
draw bx by & ex ey q*;
Print[q@m$name," ",q@K1[]];
```

まとめ

- クラスの使用は、便利である場合がある反面、その設計には注意が必要である。
- 設計を熟考せず、日和見的にプログラミングを始めると不幸率なプログラムになったり、本来見通しを良くするために導入した「オブジェクト指向」が無意味になったりすることがしばしば起こる。