目次

★2　オブジェクトの生成とメソッド呼び出し

★2.1　手続き型プログラミングとオブジェクト指向プログラミング

C 言語のような言語では，データとそれに対する処理をばらばらに書く．データにどのような処理を施すか，その手続きを記述するスタイルとなっている．このようなスタイルのプログラミング言語は**手続き型**であるという．

これに対して，Java は**オブジェクト指向**であるといわれる．オブジェクト指向のプログラミング (☆ 1) では，やりたいことを実現するために必要な「もの」を定義してそれを生成し，それにメッセージを送って何かをやってもらう（あるいは「もの」同士が相互作用する）という形をとる (☆ 2)．「もの」にはデータも手続きもひとまとめになっている．

☆ 1) オブジェクト指向プログラミング: object-oriented programming.　長いので，OOP と略すことも．

☆ 2) 例えば，かめとはどんなものかを定義しておき，その定義に従うかめを何匹か生成し，そのうち一匹に「前に 100 すすめ」というメッセージを送る．

★2.2　クラス，オブジェクト，インスタンス

上記の「もの」のことを**オブジェクト**といい，オブジェクトの定義を**クラス**という．また，あるクラスの定義から実際に生成されるオブジェクトの実体のことを**インスタンス**という．以下は，よくある直感的な説明…

object, class, instance

- クラスは設計図，インスタンスはその設計図から作られたもの
- クラスはクッキー型，インスタンスは型抜きしたクッキー

Java プログラミングでは，他人が作ったクラスを利用したり，自分でクラスを定義してそれを利用したりしながらプログラムを作っていく．

★2.3　API (Application Programming Interface)

クラスを用いたプログラムを書く人が利用できる機能（クラスのオブジェクトが受け付けるメソッドなど）のことを，そのクラスの **API**(☆ 3) という．p.14 には，TurtleFrame クラス，Turtle クラスの API の仕様がまとめられている．

☆ 3) API は，「計算機システム II」の授業でも，OS とアプリケーションプログラム (AP) との仲立ちをするものとして登場したかもしれません．

★2.4 オブジェクトの生成とメソッド呼び出し (p.9)

T21.java

```
 1  /** 最初のプログラムの例 */
 2  public class T21 {
 3     public static void main(String[] args){
 4        TurtleFrame f;              // 変数 f の型宣言
 5        f =  new TurtleFrame();     // TurtleFrame を生成し f に代入
 6        Turtle m = new Turtle();    // Turtle を生成し，m の初期値として代入
 7        Turtle m1 = new Turtle();   // もう一つ生成し，m1 の初期値として代入
 8        f.add(m);                   // f に m を追加
 9        f.add(m1);                  // f に m1 を追加
10        m.fd(100);                  // m よ前に 100 進め
11        m.rt(90);                   // m よ右に 90 度回れ
12        m.fd(150);                  // m よ前に 150 進め
13        m1.rt(90);                  // m1 よ右に 90 度回れ
14        m1.fd(100);                 // m1 よ前に 150 進め
15     }
16  }
```

- このプログラムが利用する Turtle クラスと TurtleFrame クラスは，それぞれのクラスファイル (Turtle.class と TurtleFrame.class) で定義されている
- TurtleFrame 型や Turtle 型のような，クラスに対応した型のことを，**クラス型**という
- **メソッド** (method) には，上記のような**インスタンスメソッド**の他に**クラスメソッド**というものもある（次回）

★ 2.5 コンストラクタとメソッドの多重定義 (p.15)

T22.java

```
1  public class T22 {
2     public static void main(String[] args){
3        int x = 300, y = 200, d = 100;           //  int 型の変数を用意
4        TurtleFrame f = new TurtleFrame(700,500);//引数のあるコンストラクタ呼出し
5        Turtle m = new Turtle(x,y,180);
6        Turtle m1 = new Turtle(x+ d,y+ d,0);
7        java.awt.Color c = new java.awt.Color(255,0,0);// 赤色オブジェクトを生成
8        m1.setColor(c);                          // m1 の色を赤色に指定
9        f.add(m);
10       f.add(m1);
11       m.fd(d);
12       m1.fd(d);
13       m.lt(90);
14       m1.lt(90);
15       d = d / 2;   //d の値を d/2 に変更
16       m.fd(d);
17       m1.fd(d);
18       m1.moveTo(m);
19    }
20 }
```

- 整数の型 int, 浮動小数点の型 double などは，クラス型ではなく**プリミティブ型**という．プリミティブ型の値 (☆ 4) は，オブジェクトではない．
- 前の例 (T21) の 4 行目の TurtleFrame() もコンストラクタ．p14 の API 仕様を読むとわかるように，このクラスには「引数なし」,「引数 2 つ」の 2 種類のコンストラクタが用意されている．このように Java では，**同じ名前だけど引数の数や型が異なる**コンストラクタやメソッドを複数定義することができる．これを**多重定義（オーバーロード）**という．

☆ 4) プリミティブ値という．300 や 3.14 はプリミティブ値である．プリミティブ型/値のことを基本型/値ということもある．

primitive a. 原始的な，基礎的な; construct v. 組み立てる，建造する; overload n. 一般的な用法では，過負荷．プログラミング言語の世界では，多重定義

★ 2.6 値を返すメソッドとインスタンス変数 (p.18)

T23.java

```
 1  public class T23 {
 2     public static void main(String[] args){
 3        int d = 100, x, y, a;
 4        TurtleFrame f = new TurtleFrame();
 5        Turtle m = new Turtle(200,300,0);
 6        f.add(m);
 7        m.fd(d);
 8        x = m.getX();                 // m の X 座標のとり出し
 9        y = m.getY();                 // m の Y 座標のとり出し
10        a = m.getAngle() - 45;        // m の角度のとり出し
11        Turtle m1 = new Turtle(x, y, a); //m1 の生成
12        f.add(m1);
13        m1.fd(d);
14        Turtle m2 = m.clone();        //m2 の生成
15        f.add(m2);
16        m.rt(45);
17        m.fd(d);
18        m2.tcolor = new java.awt.Color(0,255,255); // m2 の亀の色を水色に変える
19        m2.tscale = m2.tscale * 4;  // m2 の亀を現在の 4 倍の大きさにする
20        m2.fd(d);
21     }
22  }
```

- **インスタンス変数**の他に，**クラス変数**というものもある（次回）．インスタンス変数をインスタンスフィールド，クラス変数をクラスフィールドといい，両者まとめてフィールドということもある．

宿題？

QUIZ

Q1. `T21.java` の変数 `m1` が指す亀は，p.10 の上下にいる二匹のどちらですか．

Q2. 次の文の説明として間違っているのはどれですか．`Hoge piyo = new Hoge(23);`

1. `Hoge` クラスのインスタンス生成式を含んでいる．
2. `Hoge` クラスのコンストラクタを呼び出している．
3. この文を実行すると，`piyo` は `Hoge` クラスのオブジェクトを指す．
4. 上記の選択肢の中に間違っているものがある．

Q3. Hoge クラスの API 仕様の一部が以下のようになっていたとします．このとき，Q2 で生成した `Hoge` クラスのインスタンス `piyo` に対して `piyo.fuga(3);` というメソッド呼び出しを行うと，何が起こると考えられますか．

`Hoge(int age)` — 年齢 age のほげお君を生成する

`void fuga(Turtle t)` — かめ `t` を追いかける

`void fuga(int n)` — `n` 回「ほげほげ〜」と叫ぶ

`void fuga(int x, int y)` — 座標 (x, y) に移動しそこでこける

次回までに…

次回は教科書第 3 章の内容を説明します．あらかじめ読んでおくこと．例題のプログラムを作成し実行してみておくこと．