第7章　7.1節
オブジェクト指向概念

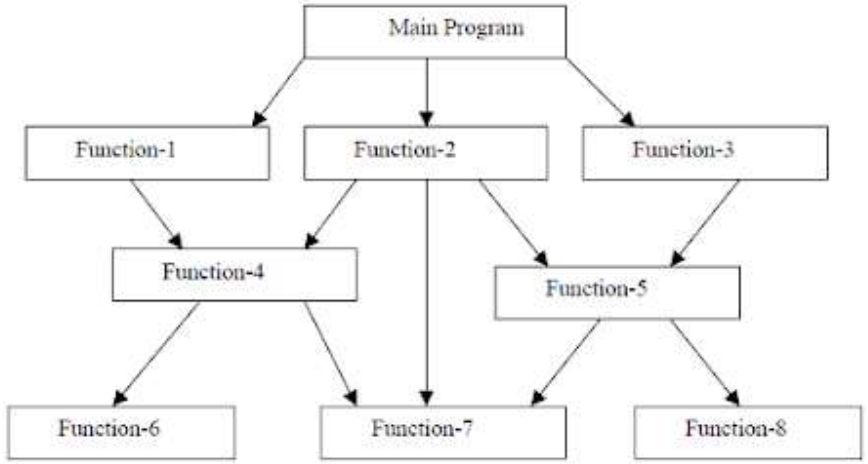
面向过程[手続き型]

- 我们到目前为止学过的都是面向过程[手続き型]编程（Procedure Oriented Programming）

定义方法：
method1
method2
method3
…

定义数据：
int
float
char
String
…

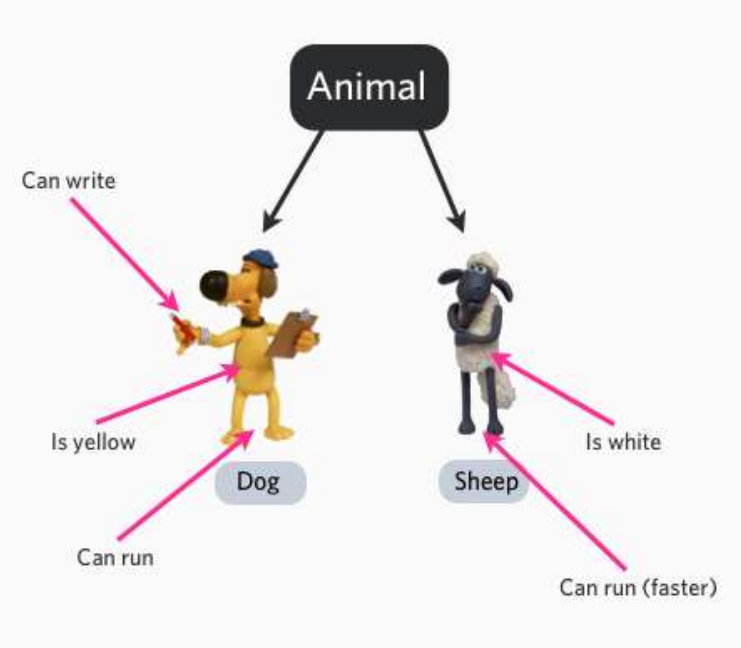
然后，各种方法，数据间的操作



Structure of procedural oriented programs

面向对象[オブジェクト指向]

- 今天要学习另一种编程思想：面向对象编程（Object oriented programming、オブジェクト指向型プログラミング）
- 对象=东西・事情，万物皆对象

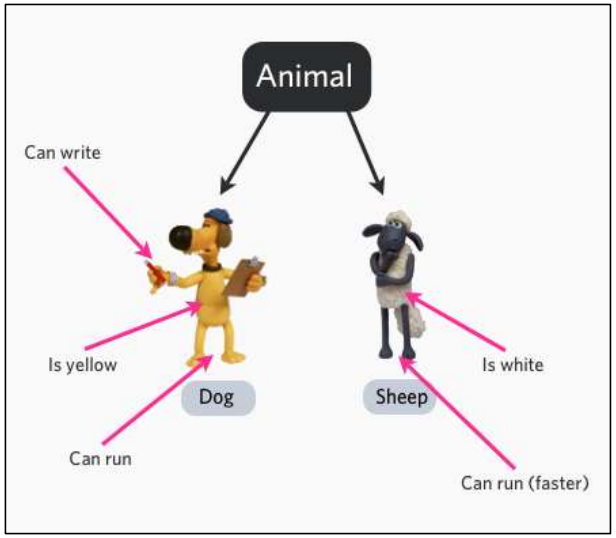


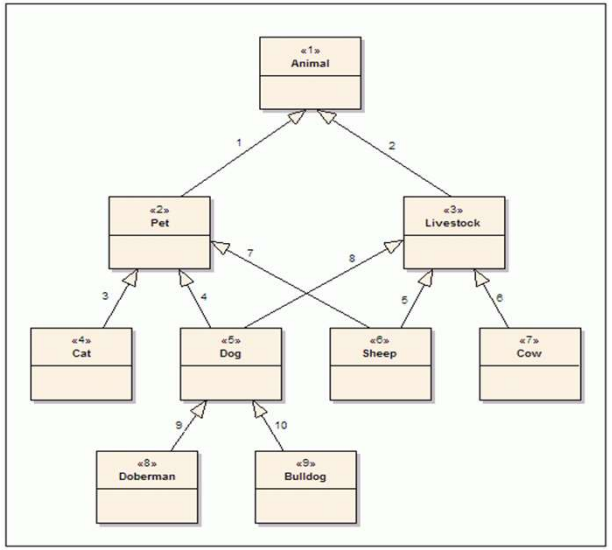

```java
public class Animal {

    public String name;
    //protected String sex;

    public Animal(String name) {
        this.name = name;
    }

    public void eat() {
        System.out.println(this.name+"吃零食！");
    }
}
```

面向对象[オブジェクト指向]

- 主要可以分为两种对象：
  - 外部对象：系统所涉及的现实物理世界中的对象
  - 内部对象：用系统实现的计算机上的表现

- 外部对象→内部对象转变难点：
  - 分析阶段：认识到问题所涉及的外部对象
  - 设计阶段：变换成内部对象的表现
  - 实现阶段：用程序表现内部对象





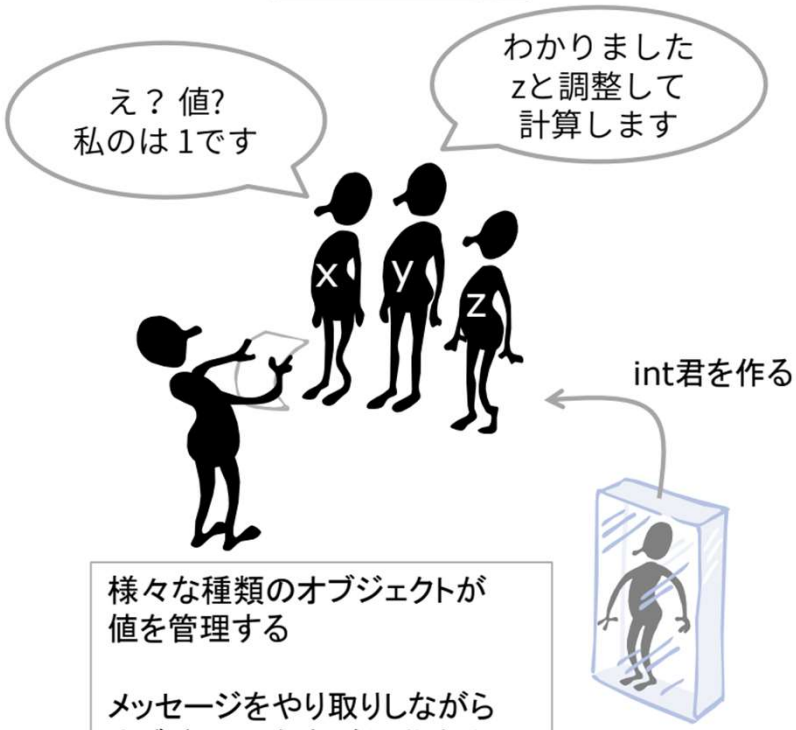
面向过程[手続き型] VS 面向对象[オブジェクト指向]

手続き型言語

有効期間(≒スコープ) の異なる
メモリ領域に変数名を付けて
値を利用(参照＆操作)する.

値を利用する関数・手続きを定義して
利用する

オブジェクト指向



様々な種類のオブジェクトが
値を管理する

メッセージをやり取りしながら
オブジェクト自身が行動する
(行動内容とその指示の分離)

面向对象编程[オブジェクト指向プログラミング]

- 想象自己是创造世界的“上帝”，需要决定这个世界里所有角色的类别和每种类别的属性以及动作。

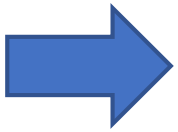
- 任务：在我们的世界里创造下面四只可爱的猫猫
  - 虽然是不同的猫猫，但是具有相同属性，可以做到的相同事情




面向对象编程[オブジェクト指向プログラミング]

- 面向对象编程： 不是一个一个编，而是将它们总结为“猫”类[クラス]，只编一个猫类

- 先设定一个猫的通用模板/蓝图，然后根据他们的不同属性和行为，创造出来四只具体的猫猫



面向对象编程[オブジェクト指向プログラミング]

猫的类 Cat class

猫的对象1

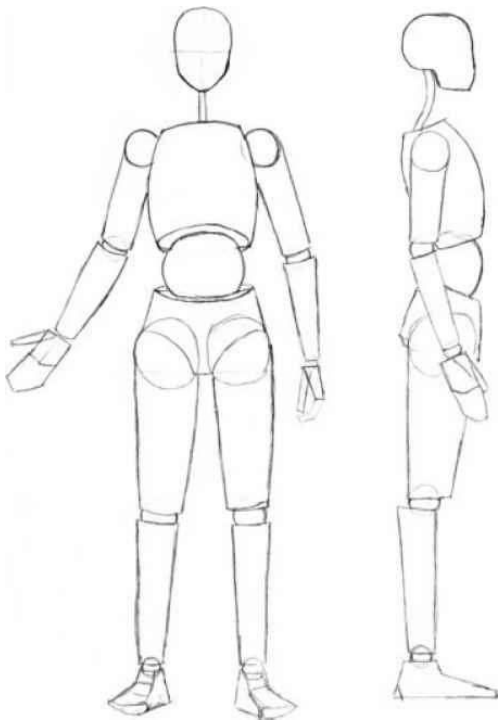
猫的对象2

猫的对象3

猫的对象4

面向对象编程[オブジェクト指向プログラミング]

人的类 Human class



人的对象1

人的对象2

人的对象3

人的对象4

对象[オブジェクト]（object）和类[クラス]（class）

- Class 类[クラス]：
  - 定义的新的数据结构的机制
  - 可以定义每一类所拥有的属性 field和动作 method
  - 只是定义，框架，创建对象的蓝图，不具有实际值（人类没有具体的名字）
- Object 对象[オブジェクト]：
  - 类的实例
  - 具体某类的对象（比如：创建一个名为Satoshi的人，那么Satoshi是人class的一个对象。还可以创建另一个名为Nakamoto的人，那么Nakamoto是人这一类的另一个对象）
  - 具有实际值（此人类对象有名字为Satoshi的属性）

面向对象编程[オブジェクト指向プログラミング]

猫的类 Cat class

- 定义猫所拥有的属性和行为

猫的对象1

猫的对象2

猫的对象3

猫的对象4

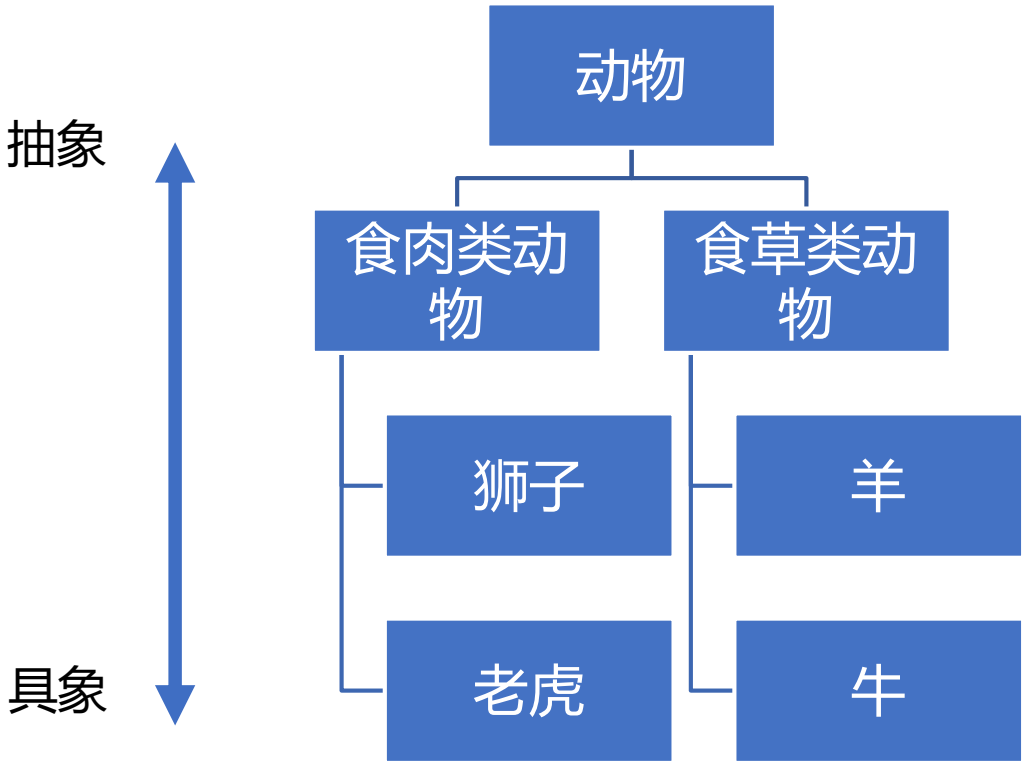
属性/成员变量[フィールド]（Field）和行为/方法[メソッド]（Method）

- 属性/成员变量[フィールド] = 属于猫的变量：
  - 具体的数据：猫毛长度，皮毛颜色，名字…
  - 对象的状态：吃饭了没，起床了没，抓到老鼠了没…
- 行为/方法[メソッド] = 猫可以执行的函数：
  - 属于这个类的对象所固有的动作：四种猫都会叫“喵”



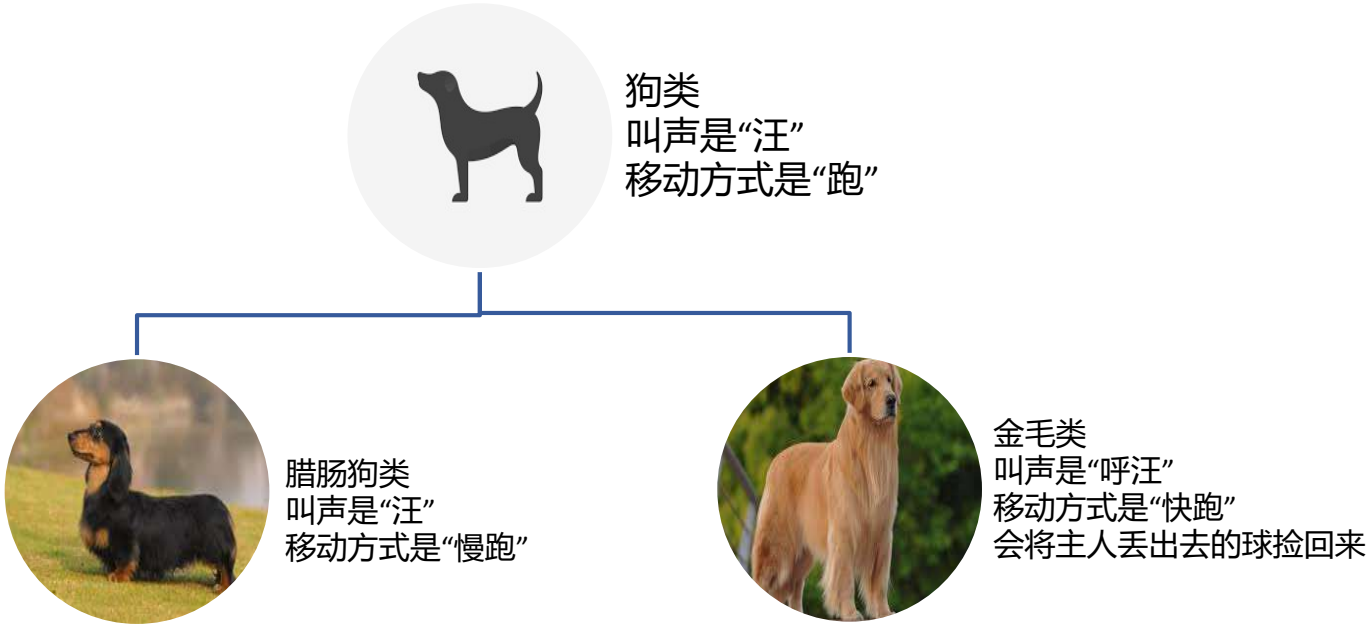

父类[スーパークラス]与子类[サブクラス]

- 子类就是父类再进行细分得到的类：
  - 狮子和老虎是食肉类动物和动物的子类
  - 食肉类动物是动物的子类，是狮子和老虎的父类
  - 动物是以下所有类的父类



继承[繼承]

- 子类继承了父类的所有属性（field）以及方法（method）
- 子类一定拥有父类所拥有的属性与方法，但属性的值以及方法的具体形式可以不同于父类，并且子类可以有父类没有的属性及方法。



面向对象编程[オブジェクト指向プログラミング]

- 在面向对象编程的世界中，一切皆为对象，**对象**都有**属性**和**行为**，每个对象都是独一无二的，而且对象一定属于某个类（型）。当我们把拥有**共同特征**的对象的静态特征（属性）和动态特征（方法）**抽取**出来后，就可以定义出一个叫做“类[クラス]”的东西。
- 按照这种编程理念，程序中的数据和操作数据的函数是一个逻辑上的整体，我们称之为“对象[オブジェクト]”，而我们解决问题的方式就是创建出需要的对象并向对象发出各种各样的消息，多个对象的协同工作最终可以让我们构造出复杂的系统来解决现实中的问题。
- 资料来源

面向对象编程[オブジェクト指向プログラミング]

- 面向对象编程 Object-oriented Programming：
  - 把一组数据结构和处理它们的方法组成对象（object），把相同行为的对象归纳为类（class），通过类的封装[カプセル化]（encapsulation）隐藏内部细节，通过继承[継承]（inheritance）实现类的特化[特化]（specialization）和泛化[汎化]（generalization），通过多态[ポリモーフィズム]（polymorphism）实现基于对象类型的动态分派。

- 这里看不懂不要紧，只是放一个定义给大家参考。