オブジェクト指向論(Q)

オブジェクト指向概論(B1)
オブジェクト指向(K1)

第4回講義資料
2023/5/1

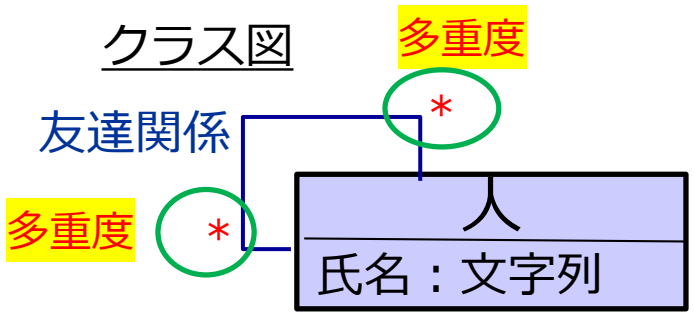
來村 徳信

同じクラスのインスタンス間の関連の多重度

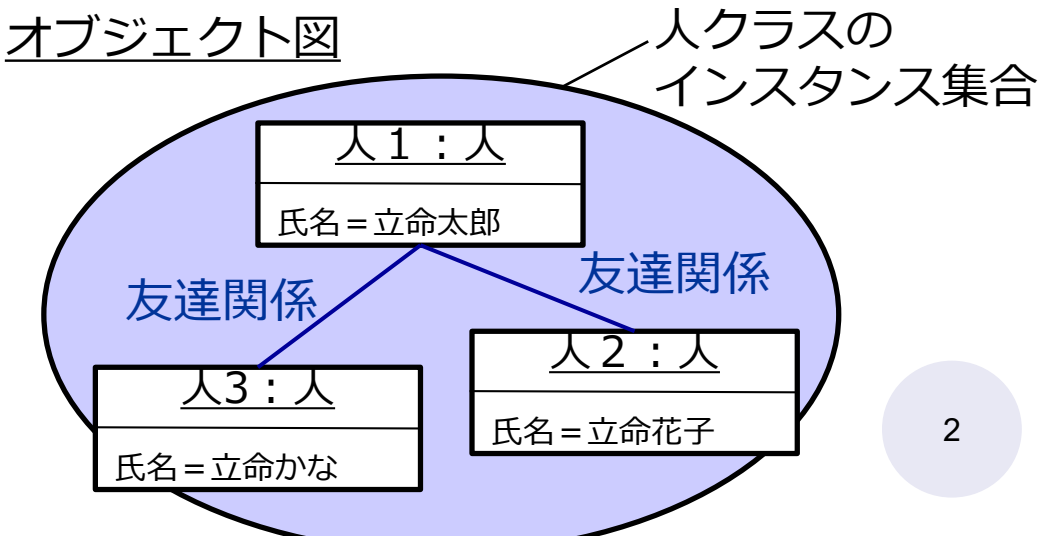
前回講義の補足

第2回講義資料＋α

- 関連＝インスタンスの間の関係
  - 同じクラスに所属するインスタンス間にも関連がありえる
    - クラス図では，ひとつのクラスにループするように関連を書く
    - オブジェクト図では，ひとつのインスタンスでループするように記述されるわけではない．
    - 異なるインスタンス間でも必ずしもループするわけではない．
    - クラス図の関連は，オブジェクト図のリンクの「集合」だから
  - 例：友達関係



※「自己ループしない」（＝同じインスタンスの間には友達関係は成り立たない）という制約の記述が必要．

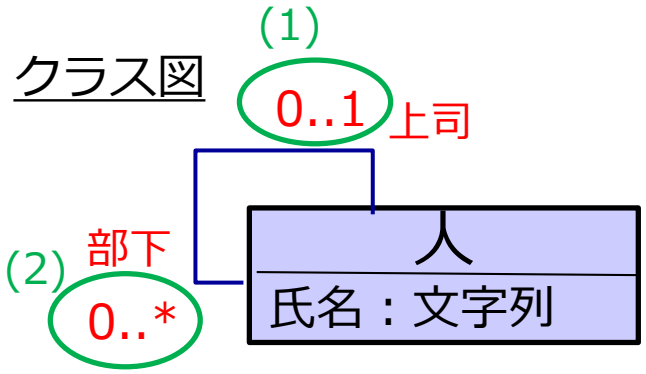


同じクラスのインスタンス間の関連の多重度(2)

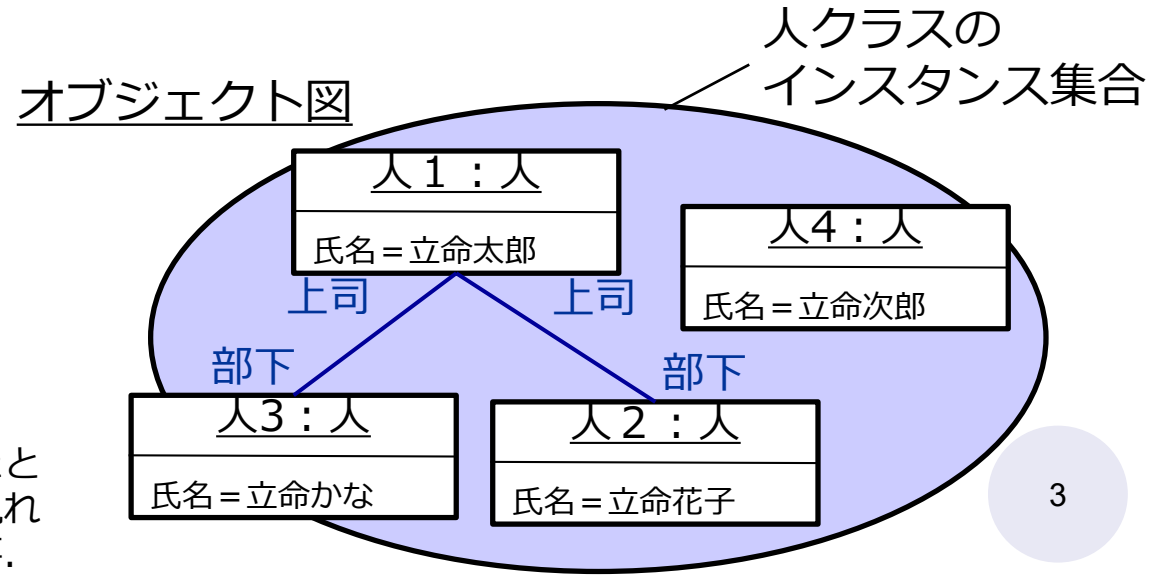
前回講義の補足

NEW

- 例2：会社員の上司ー部下関係
  - 上司はいるとしたら一人．部下はいない人もいるし，部下が複数人いる人もいる，とすると？
  - (1) 一人の人間インスタンスを考えると，上司がいない人もいるので 0.. で，いるとしたら一人のみなので，0..1
  - (2) 同様に考えると，部下がいない人もいるので 0.. で，いるとしたら複数人なので，0..*



※「ループしない」（＝辿ったときに同じインスタンスが２回現れない）という制約の記述が必要．

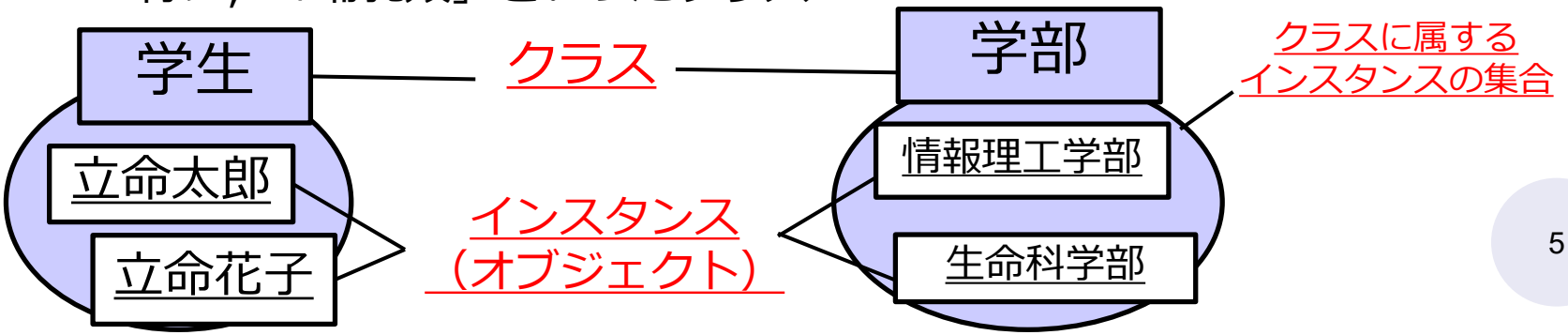


今回の講義のテーマと流れ

- UMLのクラス図の発展
- クラス図における汎化（分類階層）
  - 分類階層・分類関係
    - 汎化と特化．UMLにおける記法．
  - 性質の継承
    - 属性の継承
    - 操作の継承
      - 操作名の継承と実装の継承
      - 抽象クラス・インタフェース
    - 関連の継承
  - 汎化の区別 <重要>
    - (1)クラスーインスタンス関係との区別
    - (2)関連との区別
    - (3)集約との区別
  - カテゴリークラス

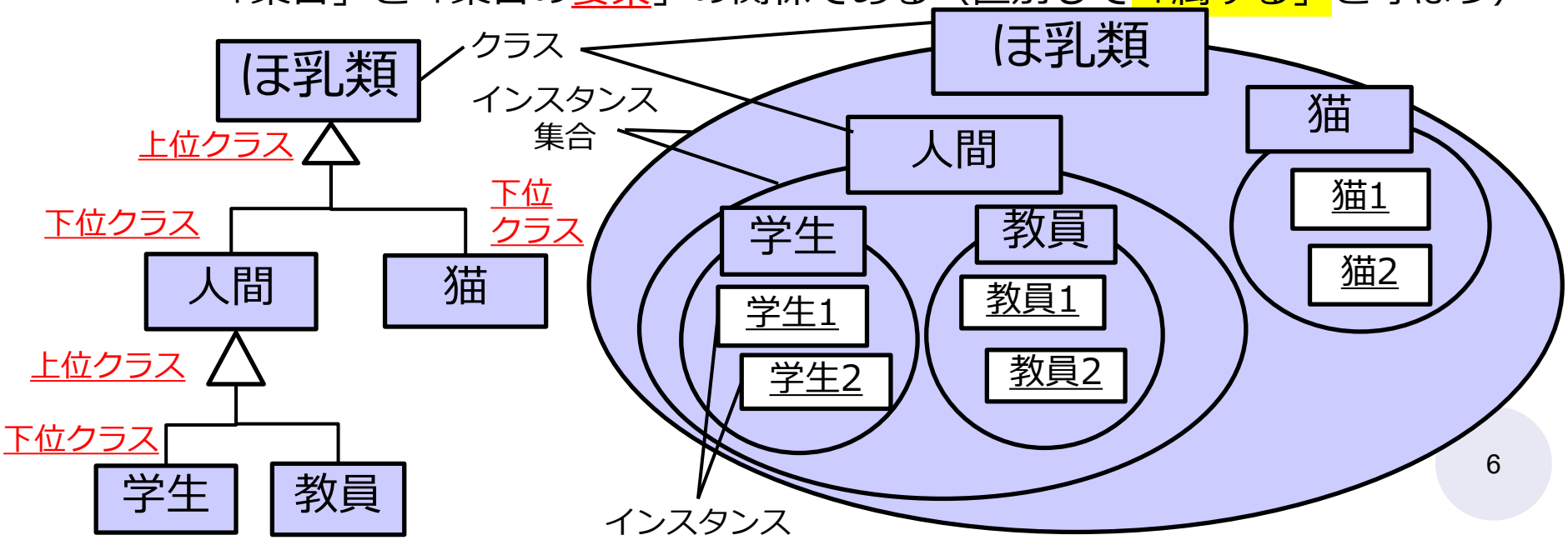
クラスとインスタンス

- インスタンス(instance) ≒ 個物
  - 現実世界に存在する個別のモノ・コト．実例．具体的．
  - オブジェクト ≒ インスタンス
    - 単にオブジェクトといった場合，インスタンスを指す（ことが多い）
    - 特にUML図の「オブジェクト図」はインスタンスの図を表す
- クラス(class)
  - 「共通の性質」を持つインスタンスの「集合」に対応する．
    - あるインスタンス e_1 はあるクラス C に「属する」という．「$e_1 \in C$」
    - 集合論的には，インスタンスは「集合の要素」
  - 「型(type)」≒「種類(kind)」≒「概念」を表す．
    - 型 ≒ インスタンスを作るときの鋳型 (cf. たいやき器の鉄板型)
  - 抽象的：実例ではない．物理的には触れない．
    - 特に，「哺乳類」といったクラス



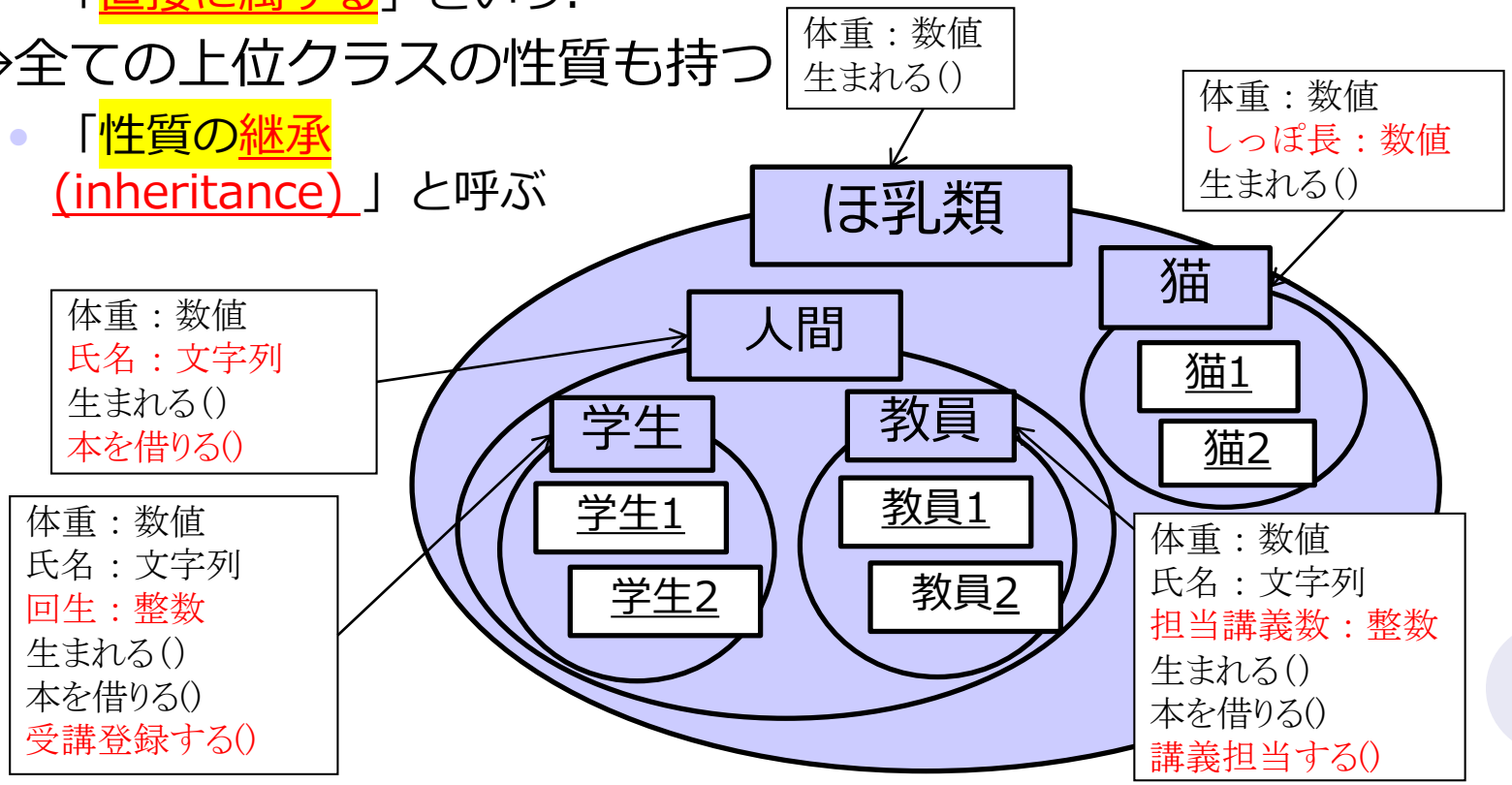
分類階層

- オブジェクトの「分類関係の階層」を表す
  - 例：学生は人間の一種．人間はほ乳類の一種．
    人間のインスタンスは学生，教員...に分類できる．
- 「上位」クラス(superclass) ー「下位」クラス（subclass)
  - インスタンス集合の間の 包含 関係（A ⊂ B)
    - 下位クラスのインスタンス集合Aは，上位クラスのインスタンス集合Bの真「 部分 集合」（下位クラスの集合が上位クラスの集合に含まれる）
    - 注意：「クラスーインスタンス」の関係も，「含まれる」とも言えるが，「集合」と「集合の要素」の関係である（区別して「属する」と呼ぼう）



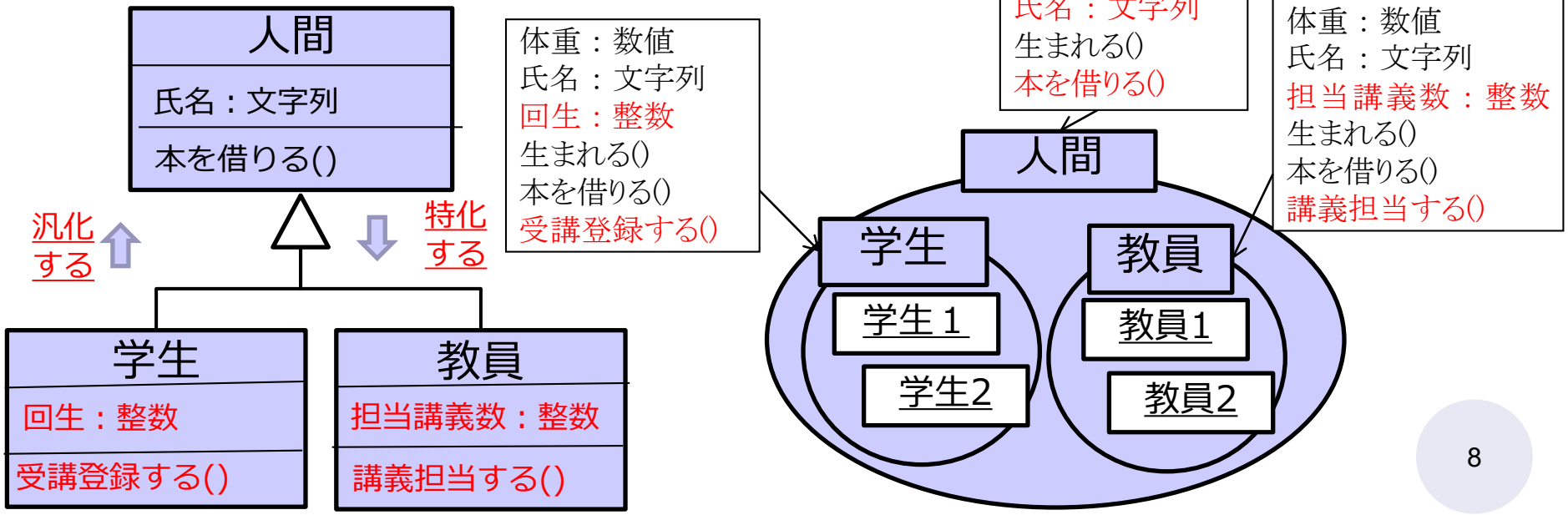
分類階層：性質の継承

- 下位クラスのインスタンスは，上位クラスにも「属する」
  - 「推移律」がなりたつ（より上の上位クラスにも属する）
    - 例：学生１は，人間クラスにも属し，そのインスタンスでもある．
      さらに，ほ乳類クラスにも属し，そのインスタンスでもある．
    - 学生1∈学生の集合，学生1∈人間の集合，学生1∈ほ乳類集合（要素∈集合）
    - 例：学生1は「学生」クラスに
      「直接に属する」という．
  - →全ての上位クラスの性質も持つ
    - 「性質の継承
      (inheritance) 」と呼ぶ



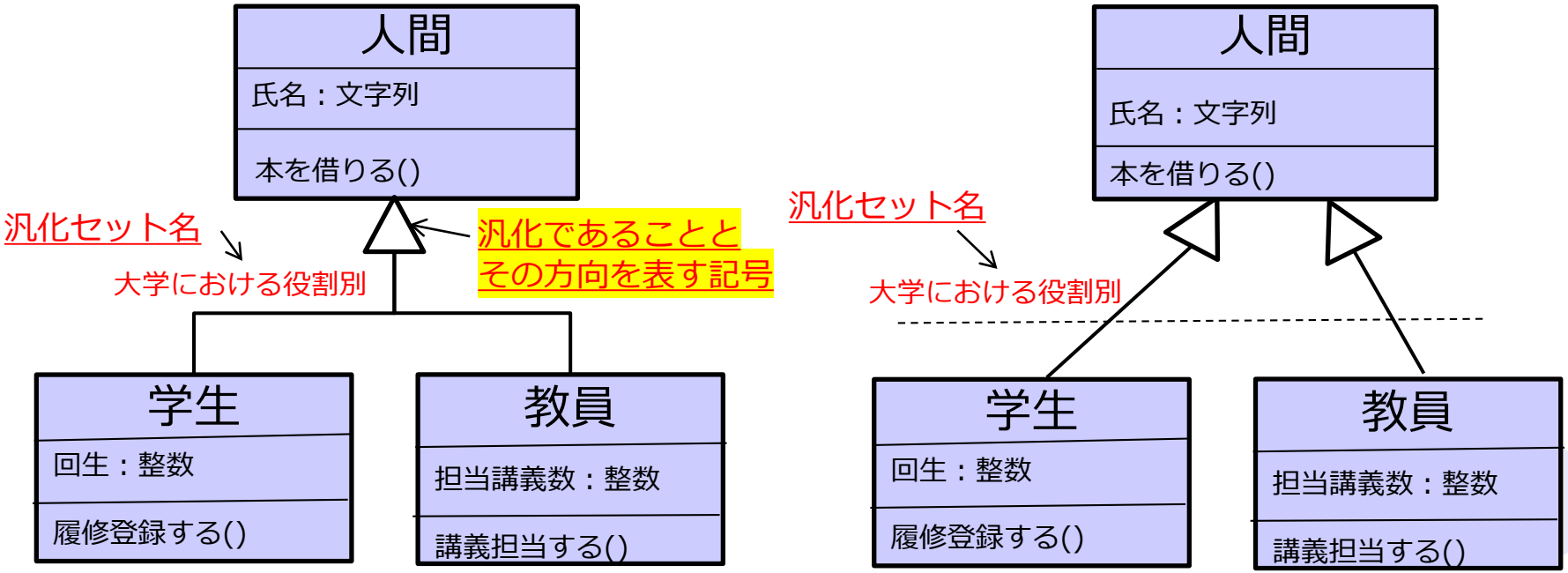
分類階層：継承による定義

- クラスを階層的に定義する
  - クラス≒「似通った性質」を持つインスタンス集合
  - 汎化 ／ 一般化(generalization)：
    - 下位クラスに共通する性質を見つけて，上位クラスを定義する
  - 特化 ／ 特殊化(specialization)：
    - 下位クラスごとに異なる性質を追加して，下位クラスを定義する
    - 共通性質は継承するので書かない



UMLのクラス図における汎化関係

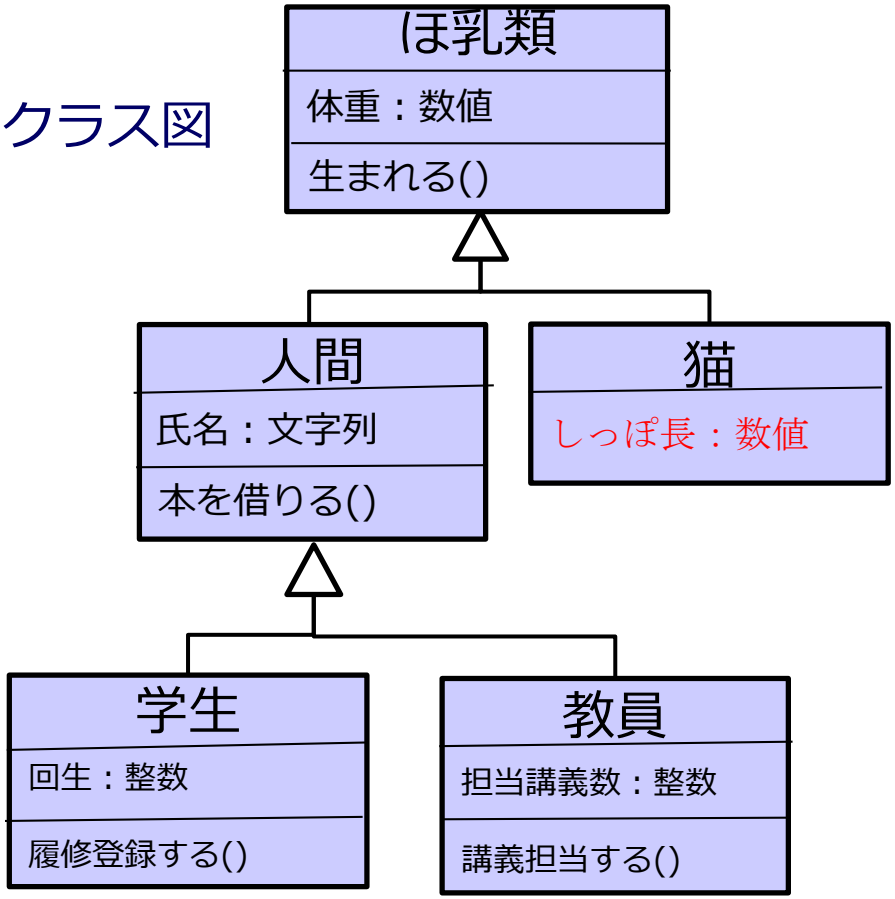
- 「汎化」の方向を示す白三角記号を付けた実線で表す
  - 汎化／一般化(generalization)
    - 上位クラス側にささる向き．左側の表記が一般的．
    - 「関連」との違いは白三角がついているかどうかだけ．
      - 集約の菱形とも図形が違うだけ．
      - しかし大きな違いがある（後述）
  - 「汎化セット」名を書くこともある
    - 分類の基準を表す．省略可能．複数ありうる（後述）



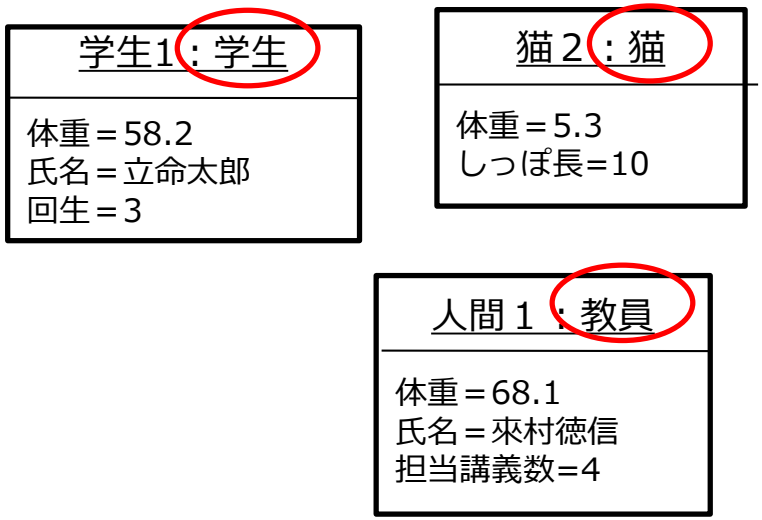
オブジェクト図

- インスタンスには「直接に属するクラス名」を記述する
  - インスタンス名：直接に属するクラス名
    - 上位クラス名はオブジェクト図だけでは分からないが，性質は上位クラスのものも引き継ぐ
  - 直接のインスタンスを作れないクラスもある（後述）

クラス図



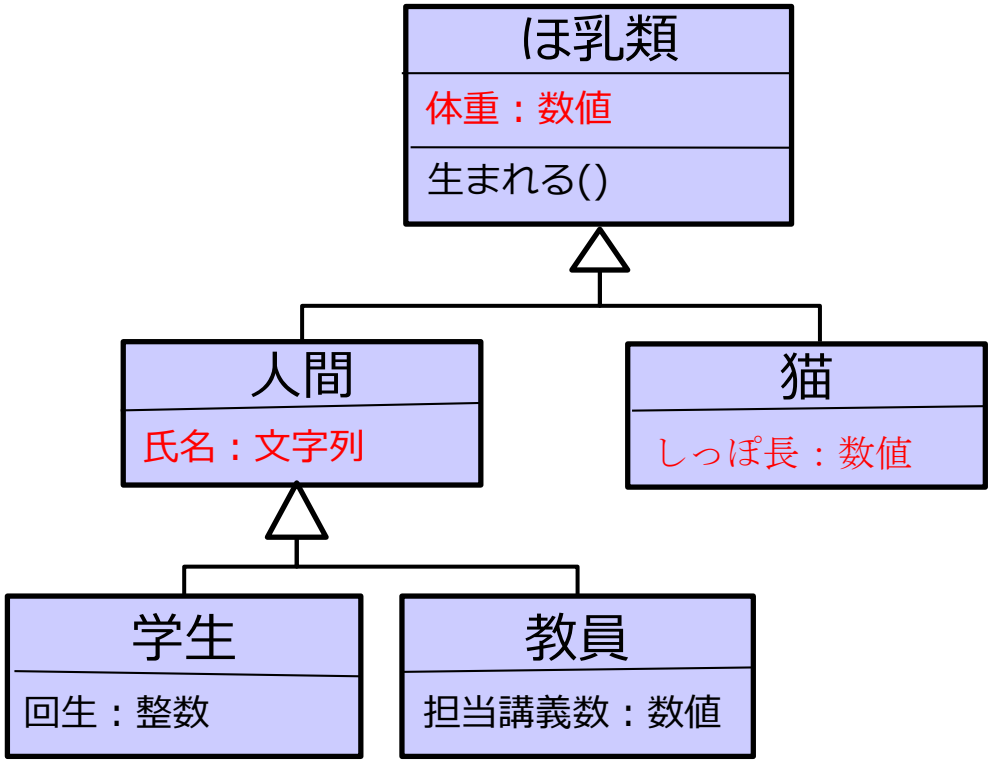
オブジェクト図



※通常，オブジェクト図には処理は書かない．

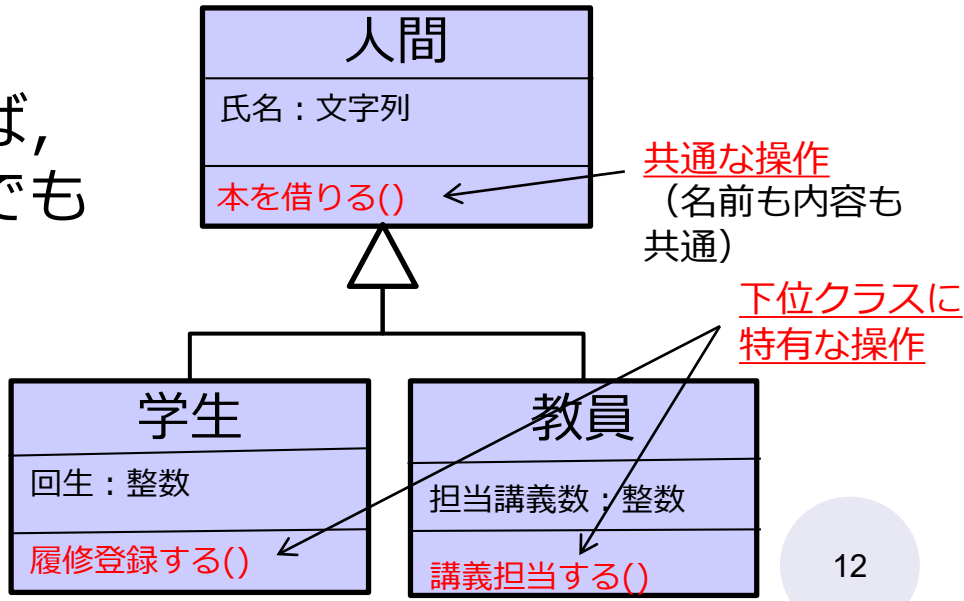
クラスで継承される性質(1)：属性

- 典型的パターン：属性名とそのタイプ
  - 例１：ほ乳類クラスの「体重」属性とそのタイプ「数値」
  - 例２：人間クラスの「氏名」属性とそのタイプ「文字列」



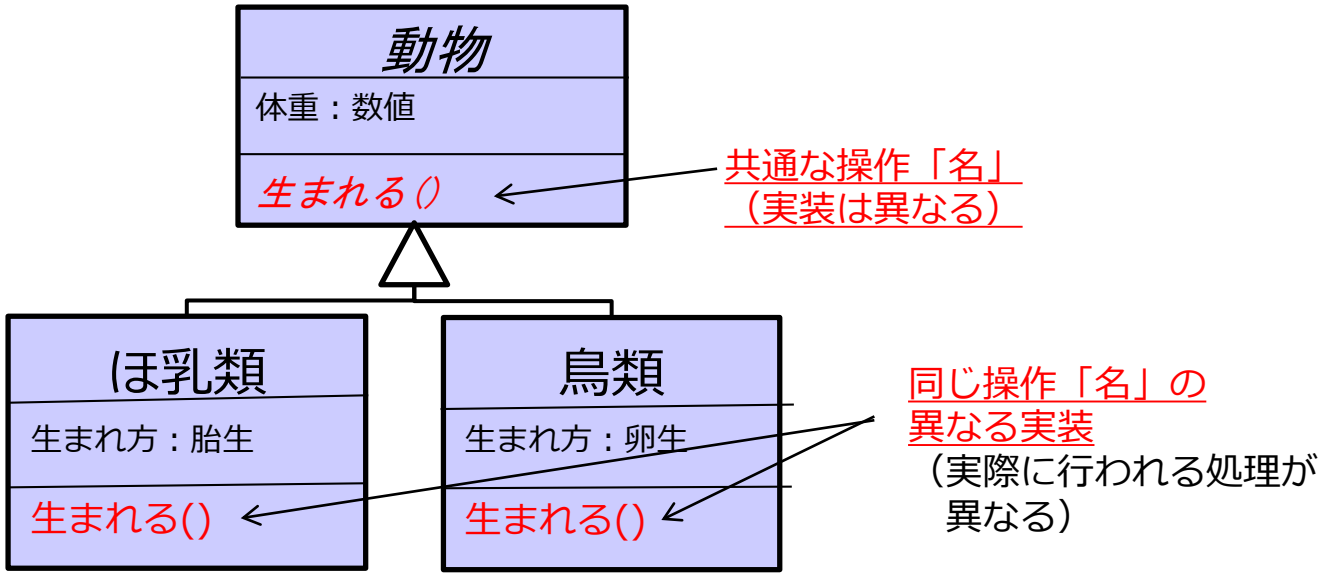
クラスで継承される性質(2)：操作

- 基本：操作はクラスで定義される
  - 操作はそのクラスの全インスタンスに共通.
    - e.g., Java のクラス定義におけるメソッドの定義
- 上位クラスの操作も下位クラスに継承される
  - 操作の継承＝操作「 名 」の継承
  - 操作の「 実装 」の継承＝操作の処理内容・プログラムの継承
- (1)操作(名)も実装も共通
  - 上位クラスで一回だけ書けば，下位クラスのインスタンスでも実行可能.
- (2)操作が(実装も)固有
  - 下位クラスで定義する



操作の実装

- (3)操作(名)は共通．実装(処理内容)は異なる
  - 処理の「実装」の違いと呼ばれる
    - 同じ意味の操作(処理)なのだが，クラスによって実際に行われる処理の内容が異なる．
  - オーバーライドによるポリモーフィズム（多態性）：
    - インスタンスに同じメッセージを送る（同じ名前のメソッドを呼ぶ）と，属するクラスごとに「異なる処理」が行われる．
    - 詳しくはプログラミングの講義回で



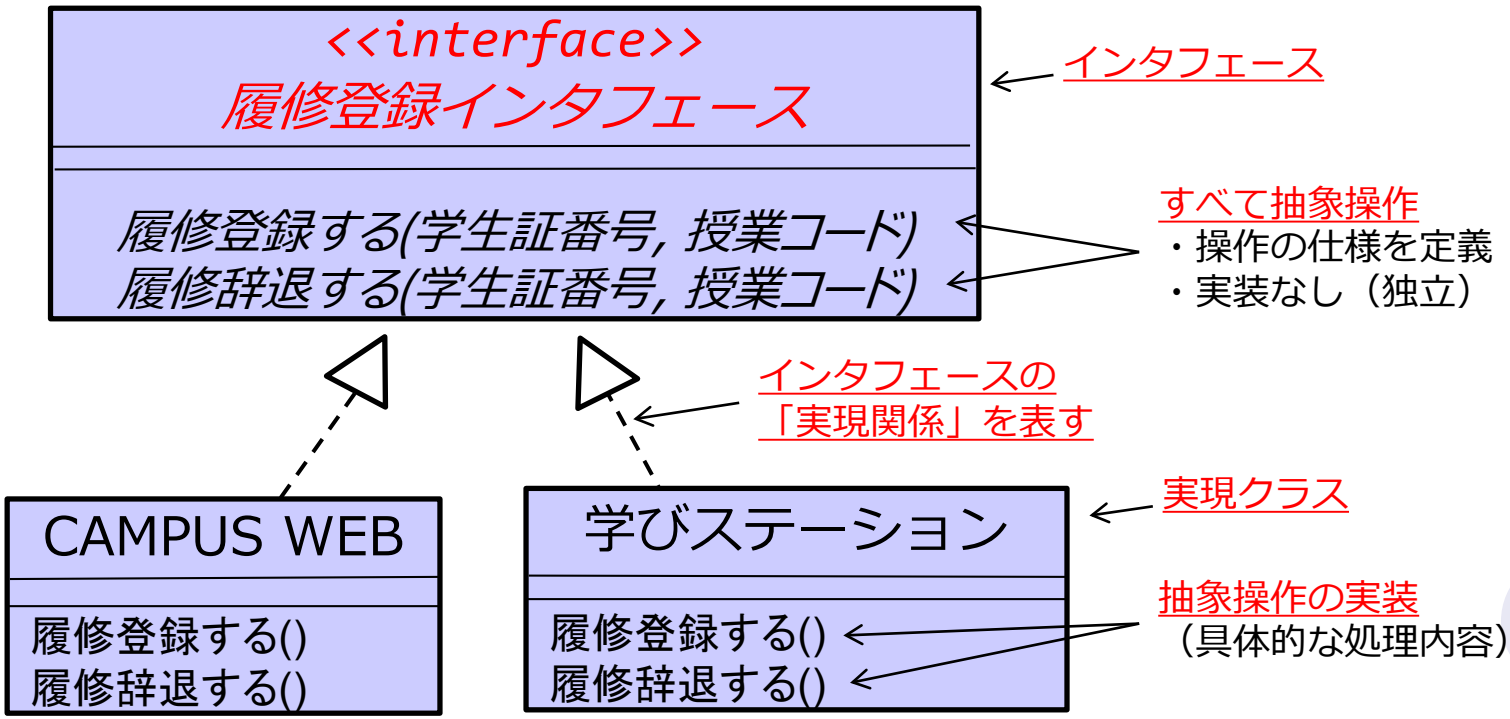
抽象クラス(abstract class)

- 直接のインスタンスを持たないクラス
  - 例：「動物」：属する全てのインスタンスは，いずれかの下位クラス（例：人間）に直接に属する．逆に，「動物」クラスに直接に属するインスタンスはない．

  ※下位クラスの分類は complete な（漏れがない）必要がある
- UMLではクラス名を「*斜字体*」にすることで表す
- 「 抽象 操作」を定義できる．*斜字体*　で表す．
  - そのクラスでは実装が定義されていない操作
  - 例：動物クラスの「*生まれる()*」操作．動物のレベルでは，生まれ方はいろいろあって，定義できない．
- 「抽象操作」は下位クラスで「実装」が定義される．
  - 例：「ほ乳類」クラスでは（胎生で）生まれる() とか，「鳥類」クラスでは（卵生で）生まれる() とか定義できる．

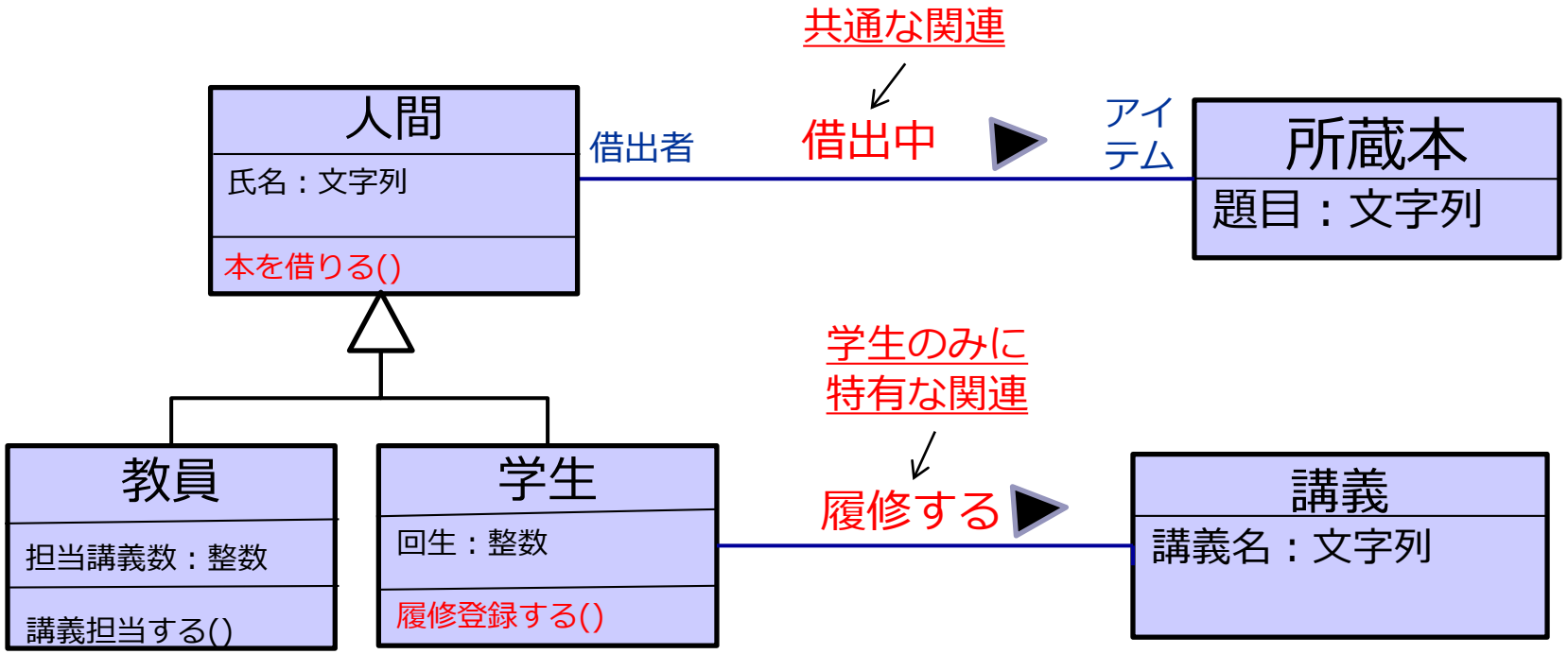
インタフェース(interface)

- 「抽象操作」のみを定義する抽象クラスの一種
  - クラス名に，<<interface>>とつけて，斜字体で表す．
  - 操作≒メソッドの仕様（機能，引数，戻り値）を定義する．
    - 実装（どのように）から独立に．詳しくは プログラム回で．
  - 「抽象操作」の実装を定義するクラス（実現クラス）との間を，点線の汎化関係で結ぶ（実現関係と呼ぶ）



クラスで継承される性質(3)：関連

- 上位クラスの関連（とその制約）も下位クラスに継承される
  - 共通な関連は上位クラスで定義する.
  - 下位クラスに固有な関連だけを下位クラスで定義する.

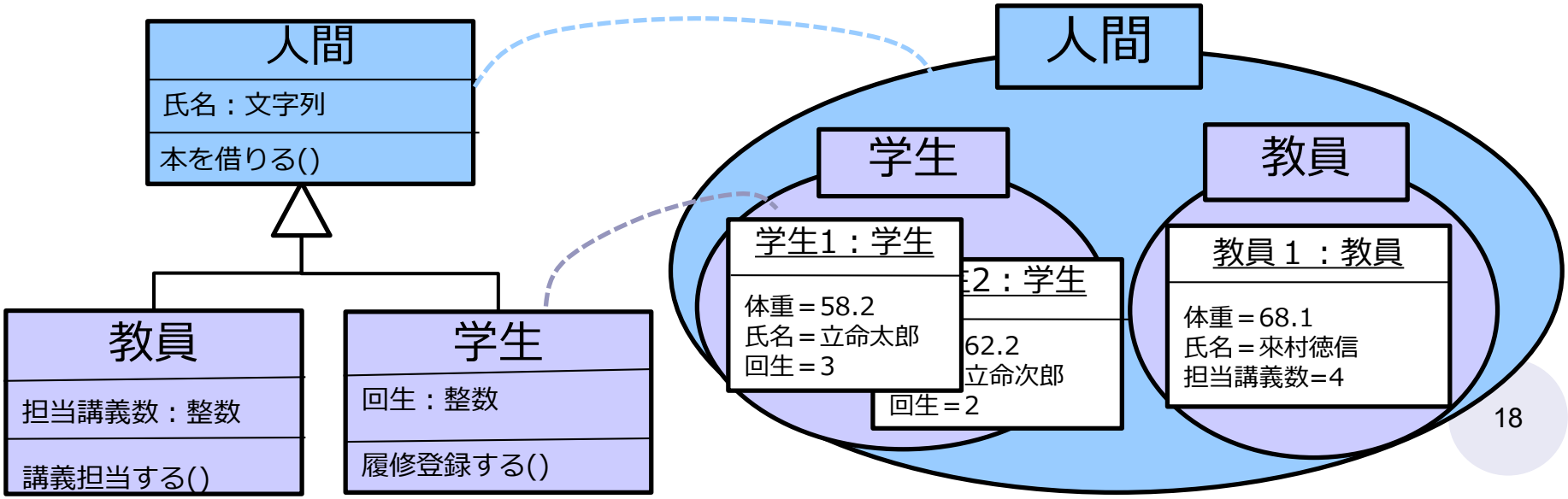


今回の講義のテーマと流れ（再掲1）

- UMLのクラス図の発展
- クラス図における汎化（分類階層）
  - 分類階層・分類関係
    - 汎化と特化．UMLにおける記法．
  - 性質の継承
    - 属性の継承
    - 操作の継承
      - 操作名の継承と実装の継承
      - 抽象クラス・インタフェース
    - 関連の継承
  - 汎化の区別 <重要>
    - (1)クラスーインスタンス関係との区別
    - (2)関連との区別
    - (3)集約との区別
  - カテゴリークラス

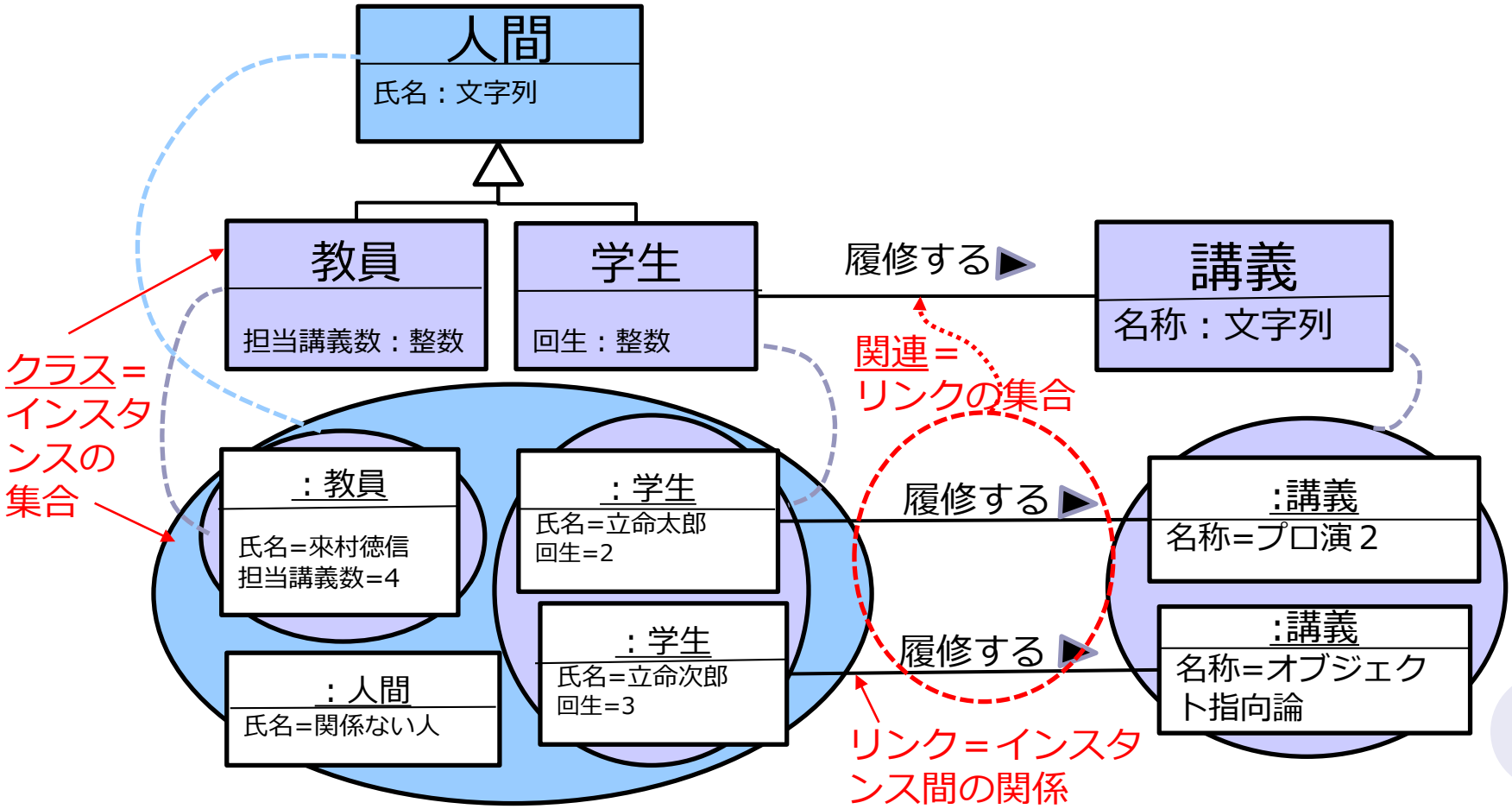
汎化の区別(1)：インスタンス関係

- 汎化／特化はクラス間の関係
  - インスタンス 集合 の間の関係に対応する
- クラスーインスタンスの関係と区別すること
  - クラスとインスタンスの関係は，集合とその要素の関係
  - 混同しやすい用語：所属する，含まれる，抽象化, 具体化, 詳細化, is-a関係, 分類
    - 下線を引いた用語は本来はクラスーインスタンス関係を表す.
    - インスタンスとカテゴリークラス（後述）の区別は難しいが…



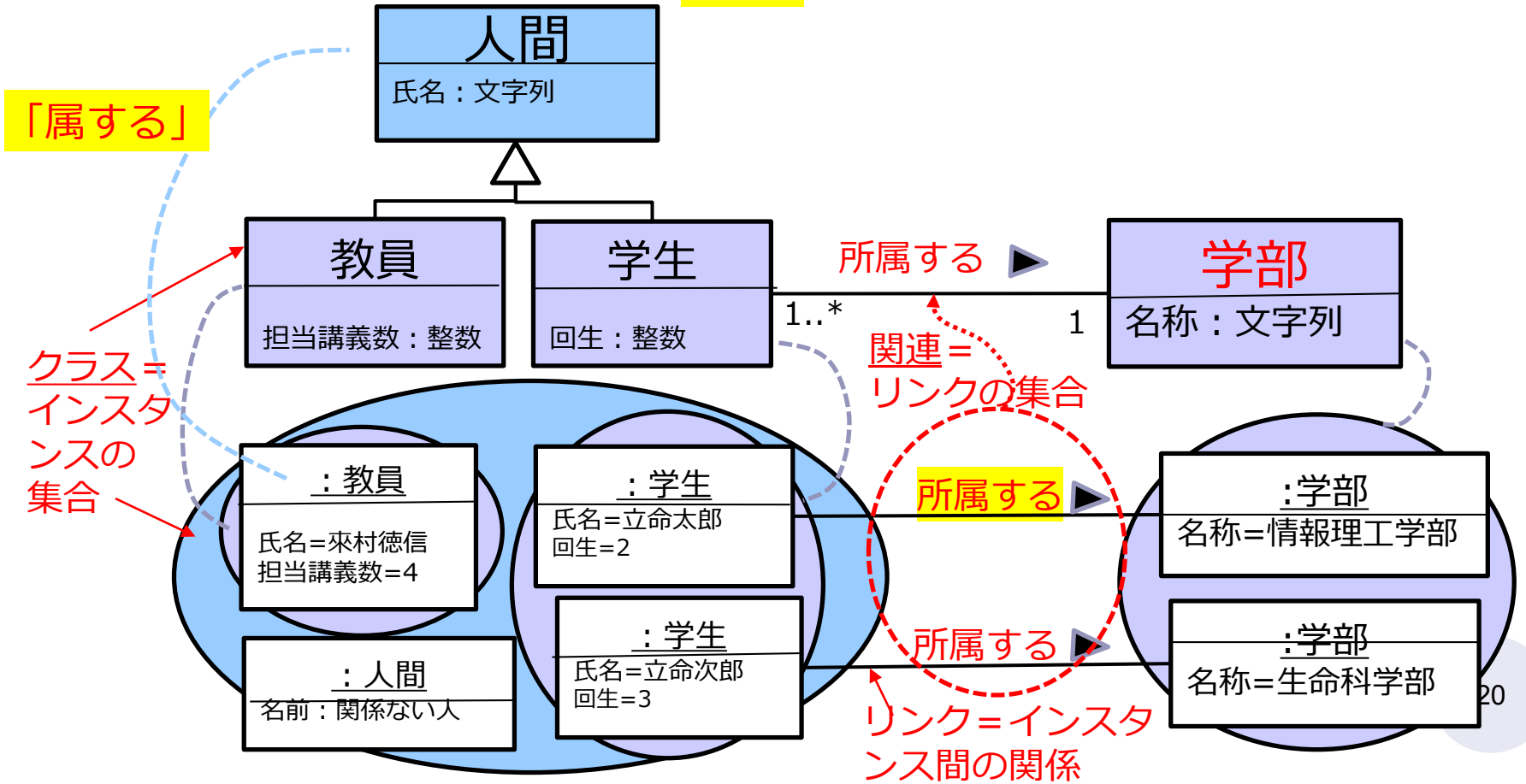
汎化の区別(2)：関連

- 汎化と「関連」との区別
  - 関連は『「インスタンス間の関係」＝「リンク」』の「集合」
  - 汎化は「クラス」間の関係＝「インスタンスの集合」間の関係



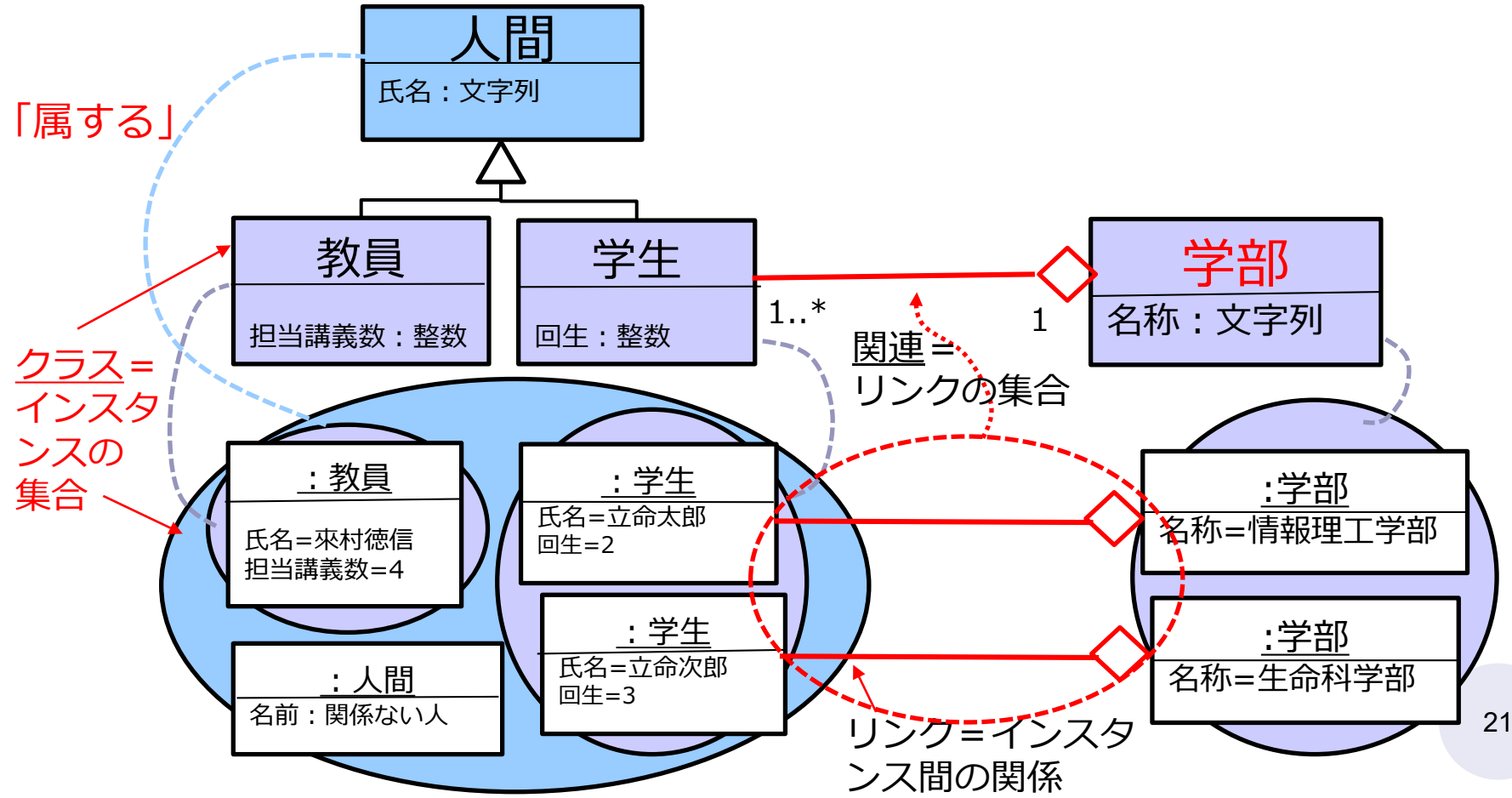
汎化の区別(2)：関連：例2

- 前回までの講義の「所属する」関連は
  - 汎化とも，インスタンスがクラスに「属する」とも異なる．
    - この「所属する」関連は，リンク＝インスタンス間の関係（の集合）
    - 汎化関係は「クラス」間の関係＝「インスタンスの集合」間の関係
    - 「インスタンスがクラスに属する」とは，要素と集合の関係



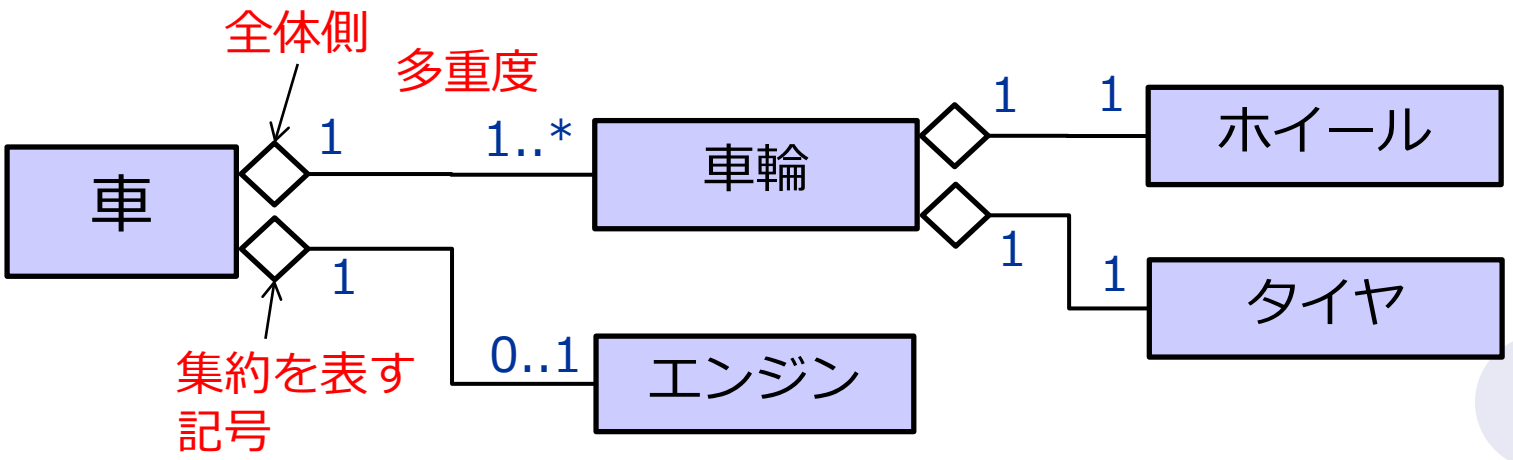
汎化の区別(2)：関連：例2　集約

- 前回までの講義の「所属する」関連は
  - インスタンスがクラスに「属する」という意味とは異なる.
  - この関連は「集約」と捉えてもよい
    - 学部が「全体」側，学生が「部分側」



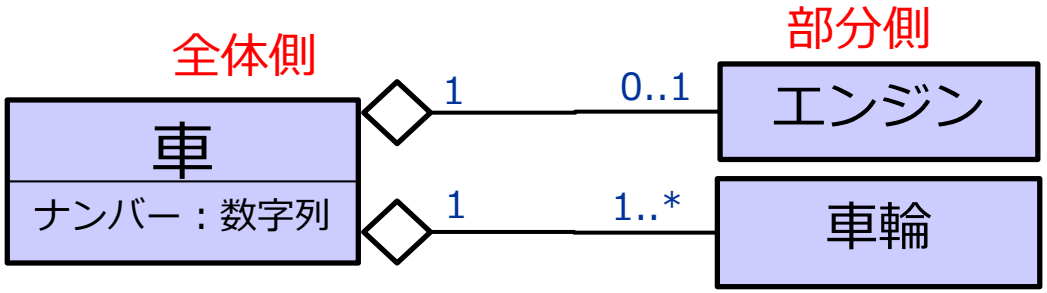
集約

- 集約(aggregation)
  - 全体ー部分関係を表す
    - 全体側からみて「has-part」，または，
      部分側からみて「part-of」と表記されることが多い.
  - 白い菱形を全体側に付けることで表す
  - 意味論的には通常の関連と変わらない
    - 「全体ー部分関係」という関連名を付けたものと同じ.
    - 全体ロール側に白い菱形を書く.



汎化の区別(3)：集約

- 汎化は「集約」とも異なる（重要！）
  - 集約は関連の一種だから当然.
  - 集約の場合は：
    - (1) 部分クラスのインスタンスは全体クラスに属しない.
      - 例：車のエンジンは，車クラスのインスタンスではない.
    - (2) 「性質の継承」も（一般には）成り立たない.
      - 例：車のナンバーは，エンジンには継承されない.



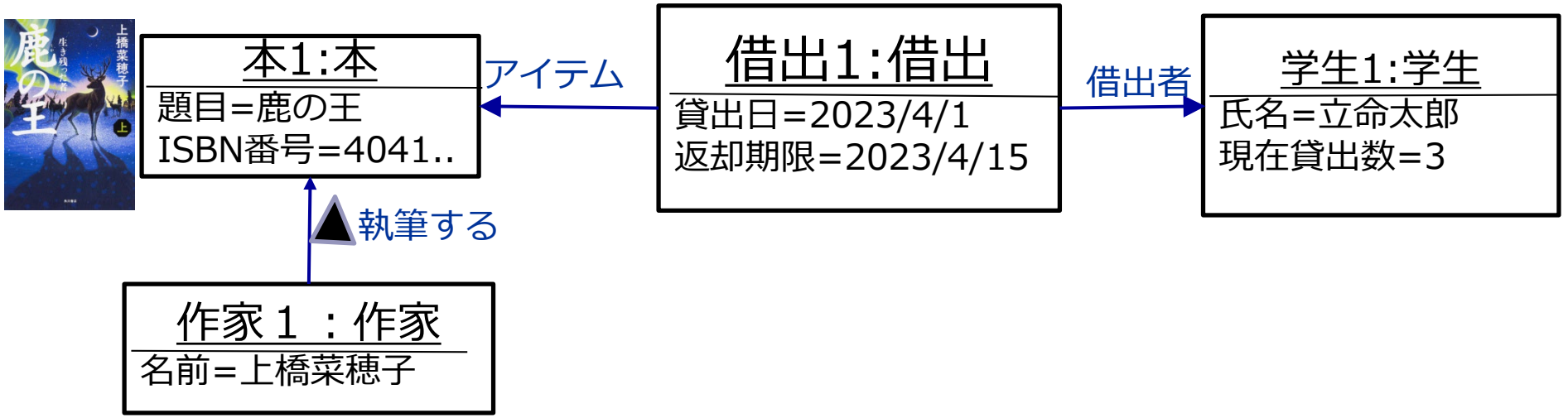
  - しかし区別しくにい場合／人がいる
    - 「含む」というと同じ．（集合か要素かを識別しないと）
    - 詳細化や具体化というと，一緒なように聞こえる.
    - 位置情報などを考えると共通なので，継承と混乱する.

今回の講義のテーマと流れ（再掲2）

- UMLのクラス図の発展
- クラス図における汎化（分類階層）
  - 分類階層・分類関係
    - 汎化と特化．UMLにおける記法．
  - 性質の継承
    - 属性の継承
    - 操作の継承
      - 操作名の継承と実装の継承
      - 抽象クラス・インタフェース
    - 関連の継承
  - 汎化の区別 <重要>
    - (1)クラスーインスタンス関係との区別
    - (2)関連との区別
    - (3)集約との区別
  - カテゴリークラス

クラスとインスタンスの区別：本の「インスタンス」とはなにか？

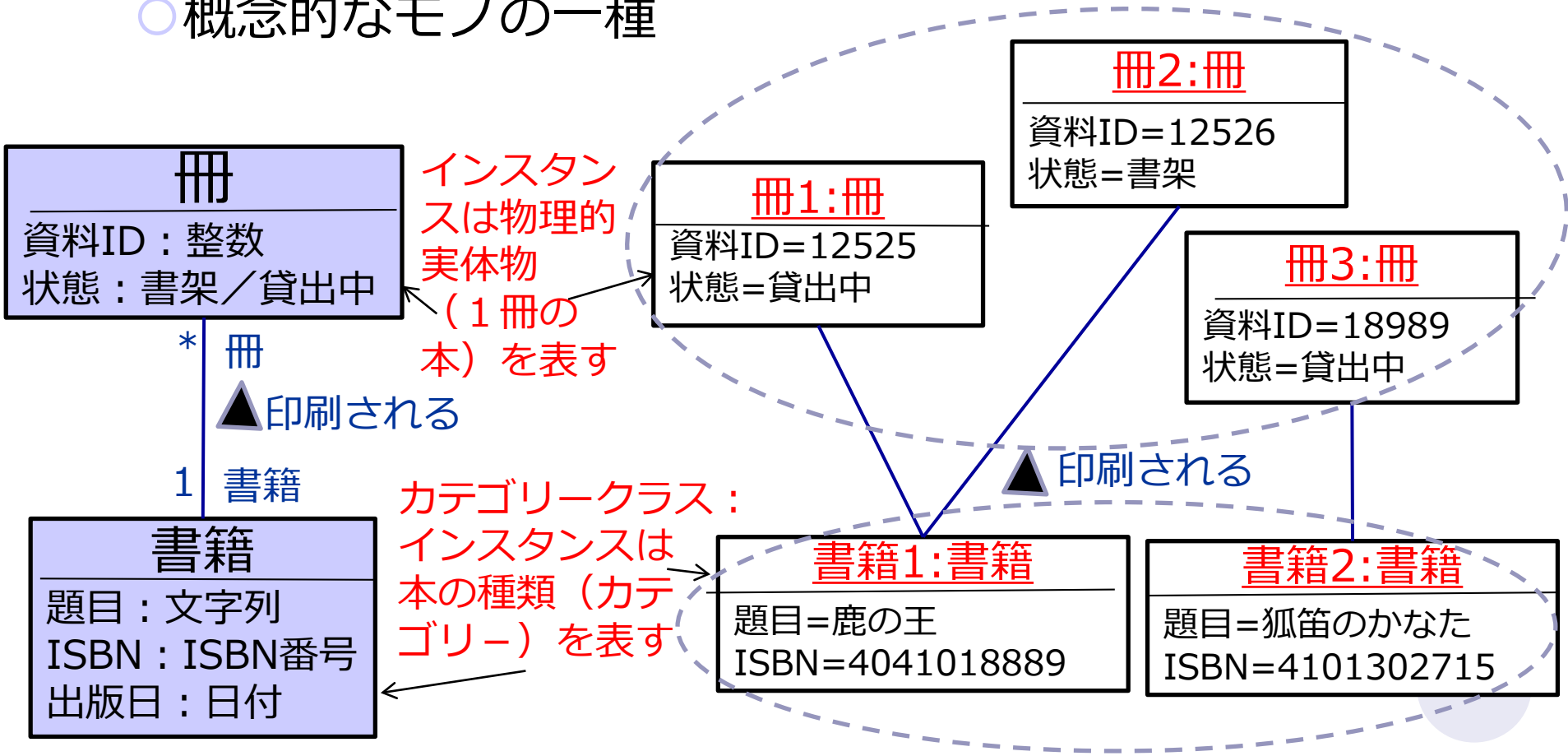
- 例：図書館情報システムのオブジェクト図



- 借り出される「本」のインスタンスとは？
  - 物理的に存在する「一冊の」本
  - 作家が執筆した本（１つのISBN番号で同定される）は複数印刷される．図書館にも複数冊が蔵書される（ことがある）．
- １つの書籍と一冊の本（「冊」）は別のオブジェクト
  - 「冊は書籍のインスタンス」とは捉えない方がよい．
  - １つの書籍が書籍クラスのインスタンスだから（次スライド：カテゴリークラス）

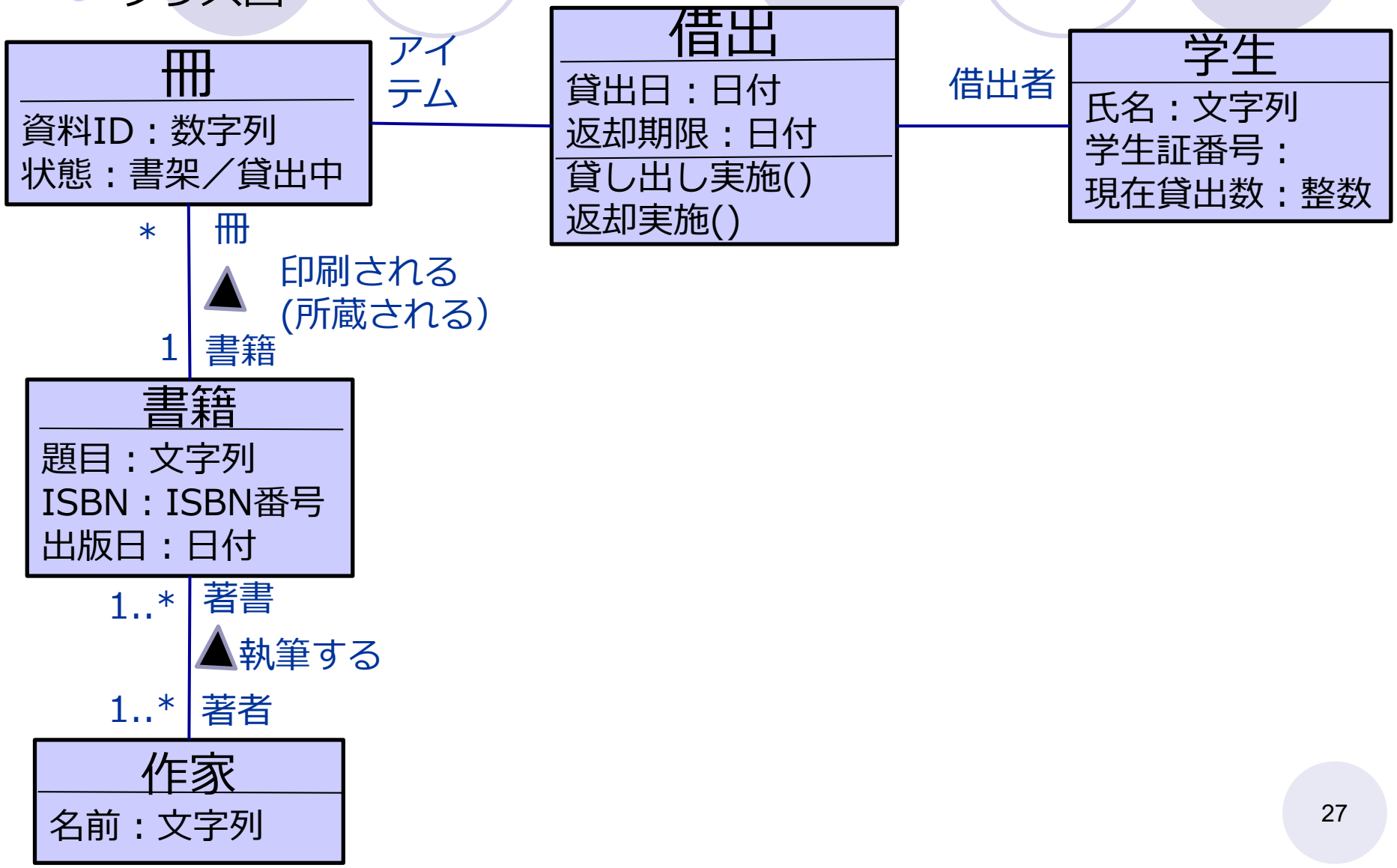
カテゴリークラス

- 「カテゴリー（種類）を表すインスタンス」をもつクラス(powertype)
  - 例：書籍，車種（車の名前）など
  - 概念的なモノの一種



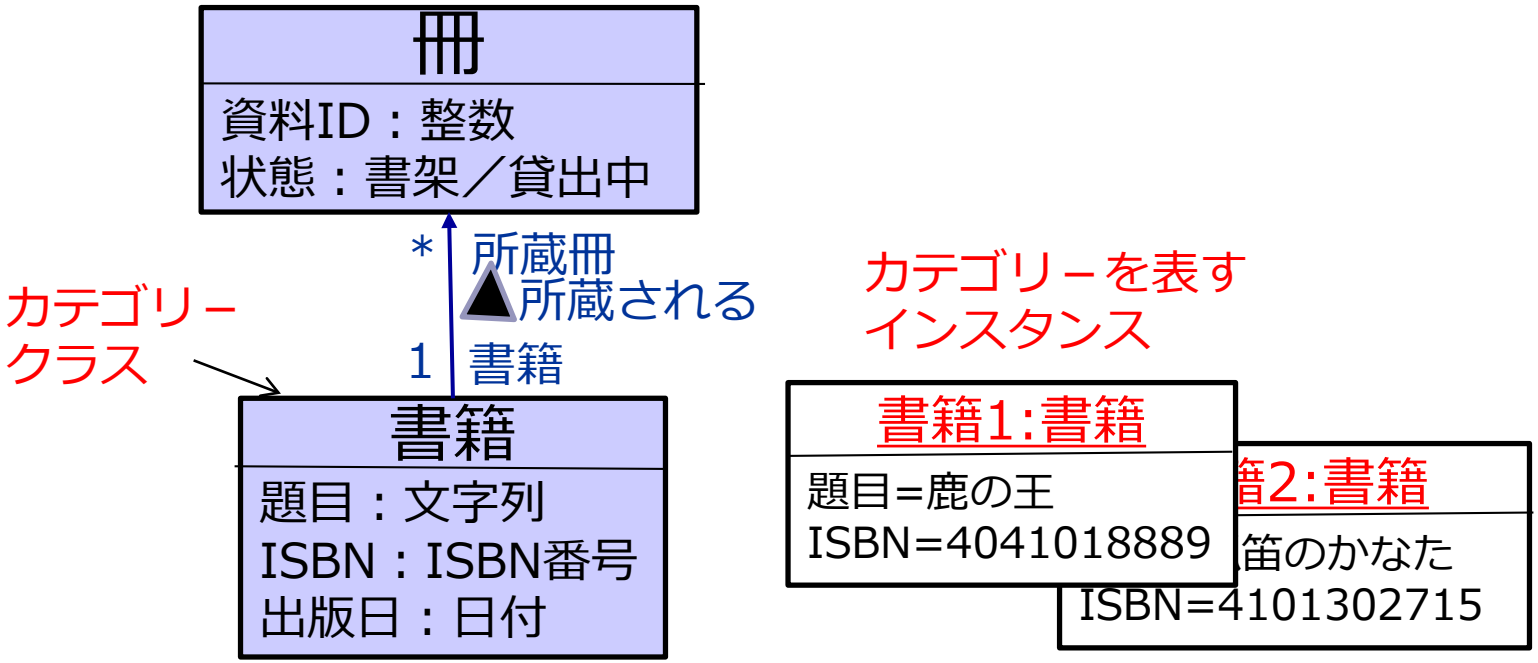
本：2つのクラスに分離する

- クラス図



カテゴリー化と汎化

- 似ているが異なる
  - 同じ点：インスタンスを分類するという目的
  - 異なる点：
    - 汎化では(通常)クラスを動的に追加できない
      - 書籍のように増えるものはカテゴリークラスにすべき
    - 汎化ではサブクラスごとに異なる処理を書ける



今回の講義のまとめ

- UMLのクラス図の発展
- クラス図における汎化（分類階層）
  - 分類階層・分類関係
    - 汎化と特化．UMLにおける記法．
  - 性質の継承
    - 属性の継承
    - 操作の継承
      - 操作名の継承と実装の継承
      - 抽象クラス・インタフェース
    - 関連の継承
  - 汎化の区別 <重要>
    - (1)クラスーインスタンス関係との区別
    - (2)関連との区別
    - (3)集約との区別
  - カテゴリークラス