明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

プログラミング演習2
クラスと継承

中村，高橋，小林，橋本

はじめに

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 今後は、講義中の課題チェックはしません
  - 2019年度までは、P演2で講義中の課題チェックをしておらず、今年度試しにやってみたのですが、やはり無理がありました。
  - 理由は、
    - 採点に時間を割くことで、十分に質問対応できていないこと
    - 問題をよく読んでいないパターンが散見されること
    - 無理な採点で、採点ミスが発生してしまっていること
  - です。そのかわり、授業時間中のOK/NGのどちらかでの判定をやめて、出来によって部分点をつけます

  - フォルダ名ミスなどはくれぐれもないように注意！
  - 課題ができていたかどうかの問い合わせは受け付けません！

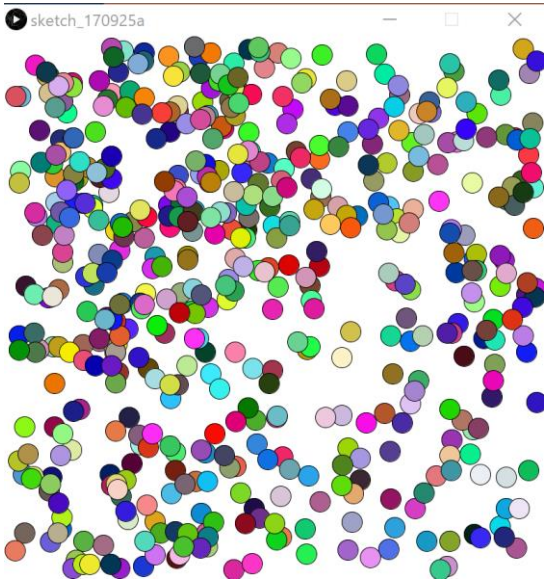
  - （注）askTAでも、これであってますか？は不可

宿題2-1: hw_listBall

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- ウインドウ800x800を用意し，Ballクラスを利用してマウスクリックするたびに丸が生まれ，上下左右にランダムな速度で動き，上下左右の端で跳ね返るプログラムを作成せよ．なお，クリックでいくらでも作成できるようにせよ
- また，Ballクラスを改良してそれぞれの丸の色は初期値としてランダムに設定するようにせよ

  ［ヒント］色情報を持つインスタンス変数を追加する
  - int red;
  - int green;
  - int blue;
- たくさん増やして管理するにはArrayListを使う！



宿題2-2: hw_Covid19Genome

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 下記のPDEを利用し，指定の要件を満たすようにプログラムを完成させよ
  - basic_Covid19Genome.pde
    - SARS-CoV-2（新型コロナウィルス）の塩基配列
  - 部分点あります
- 要件
  - 塩基配列の長さを標準出力
  - 最初の100文字と最後の100文字を標準出力
  - A,T,G,Cのそれぞれの出現率を標準出力
  - AAA,TTT,GGG,CCCのそれぞれの出現回数を出力
    - https://www.ncbi.nlm.nih.gov/nuccore/MN908947

宿題2-3: hw_imageProc

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- PImage クラスを使って画像処理をしてみよう
  - 適当な画像をダウンロードし，その画像に対してフィルタを掛けてみましょう！
  - 一番左にオリジナル画像，次にTHRESHOLDで二値化したもの，さらに次にBLURでぼかしたもの，一番右にINVERTでネガポジ反転したものを表示しよう

（ヒント）PImage を4つ定義すると良いよ！

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

プログラミング演習2
クラスと継承

中村，高橋，小林，橋本

動物園の動物クラス群

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 猫，犬，猿，象，熊を定義
  - それぞれの座標は意識したくない
    - cat.x, cat.y, dog.x, dog.y, monkey.x, monkey.y, ...
    - 内部で適当に処理してもらう
  - 描画はシンプルにしたい
    - cat.draw(), dog.draw(), monkey.draw(), elephant.draw(), ...
  - 移動もシンプルにしたい
    - cat.move(), dog.move(), monkey.move(), elephant.move(), ...
  - 睡眠も任せてしまう
    - cat.sleep(), dog.sleep(), monkey.sleep(), elephant.sleep(), ...

クラスの定義

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- Ball クラスを定義

  class クラス名 {

  　クラスの諸要素に関する定義

  }

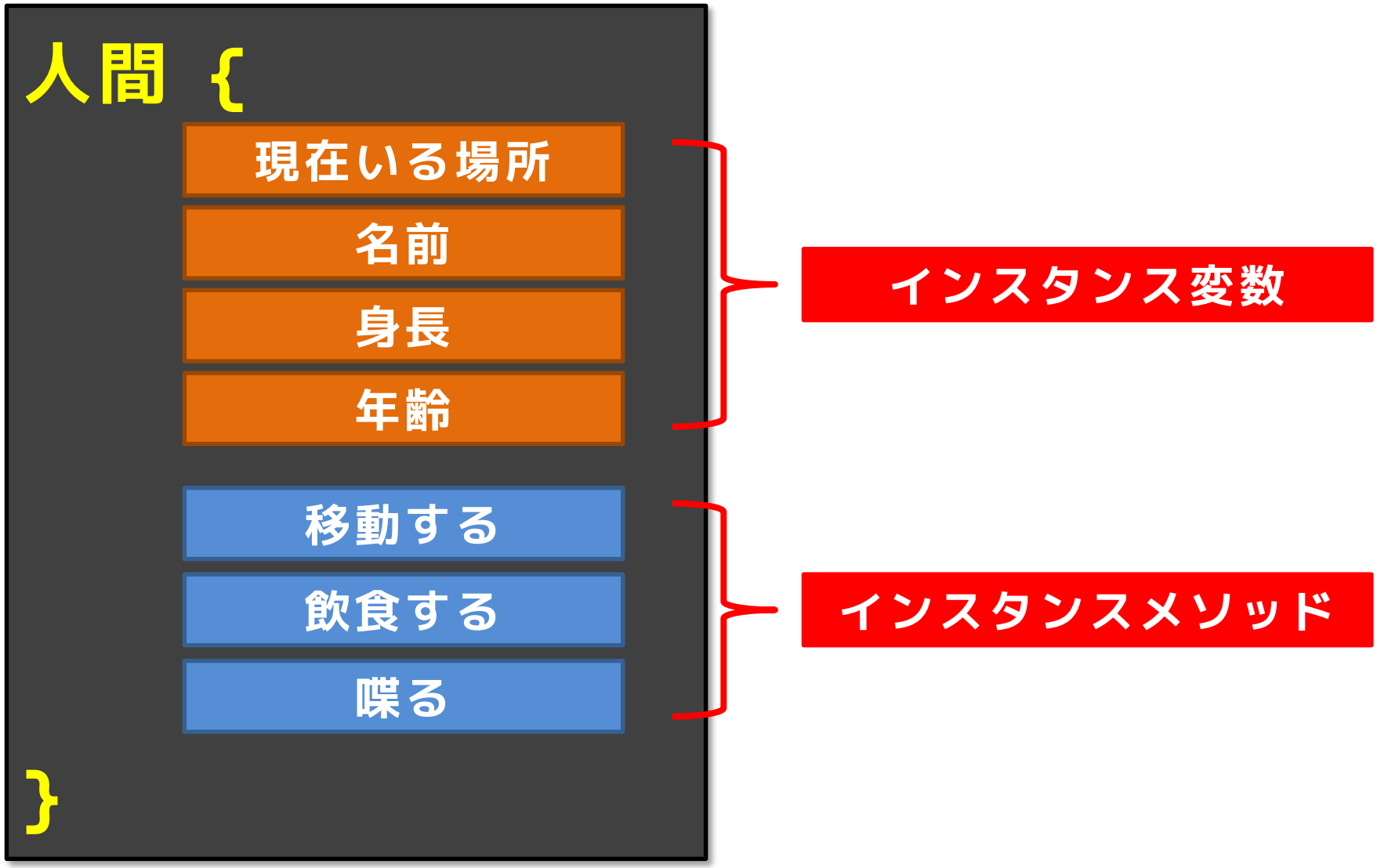
```
class Ball {

}
```

```
class Human {

}
```

```
class Animal {

}
```

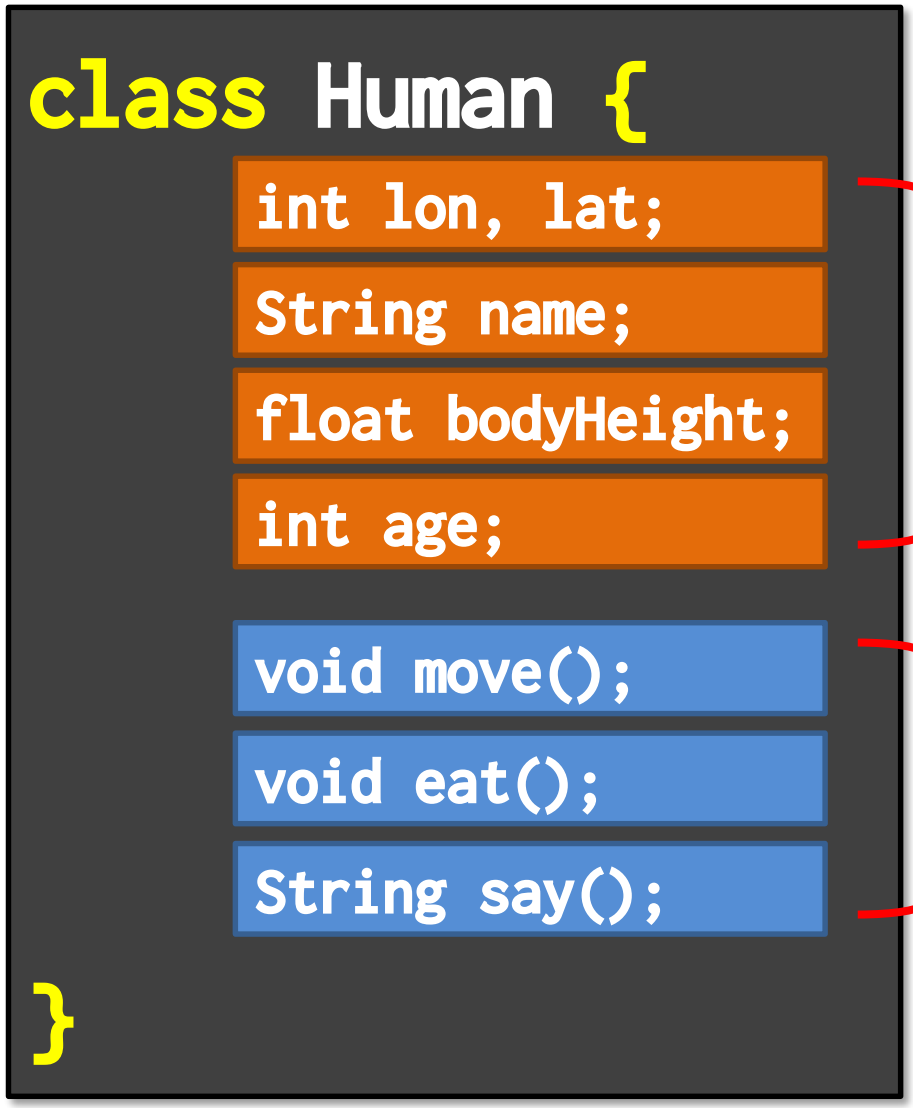
クラスの定義

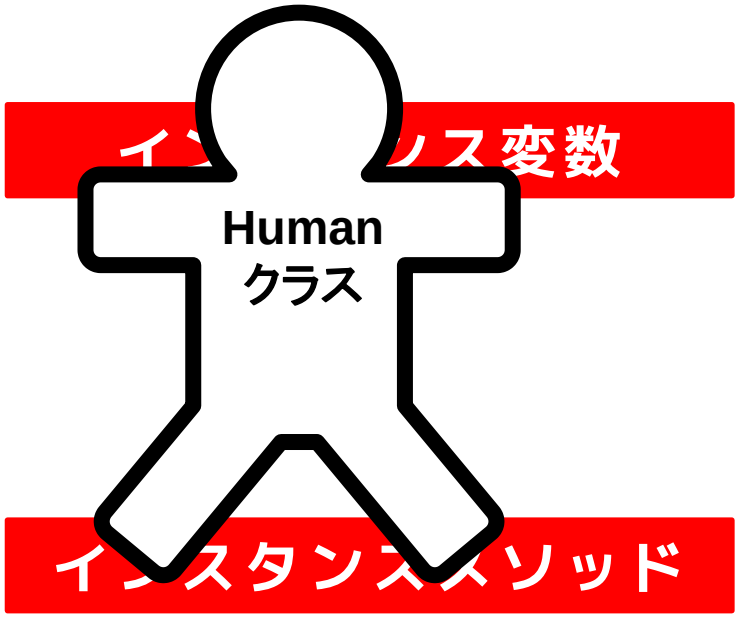
明治大学総合数理学部
先端メディアサイエンス学科
中村研究室



クラスの定義

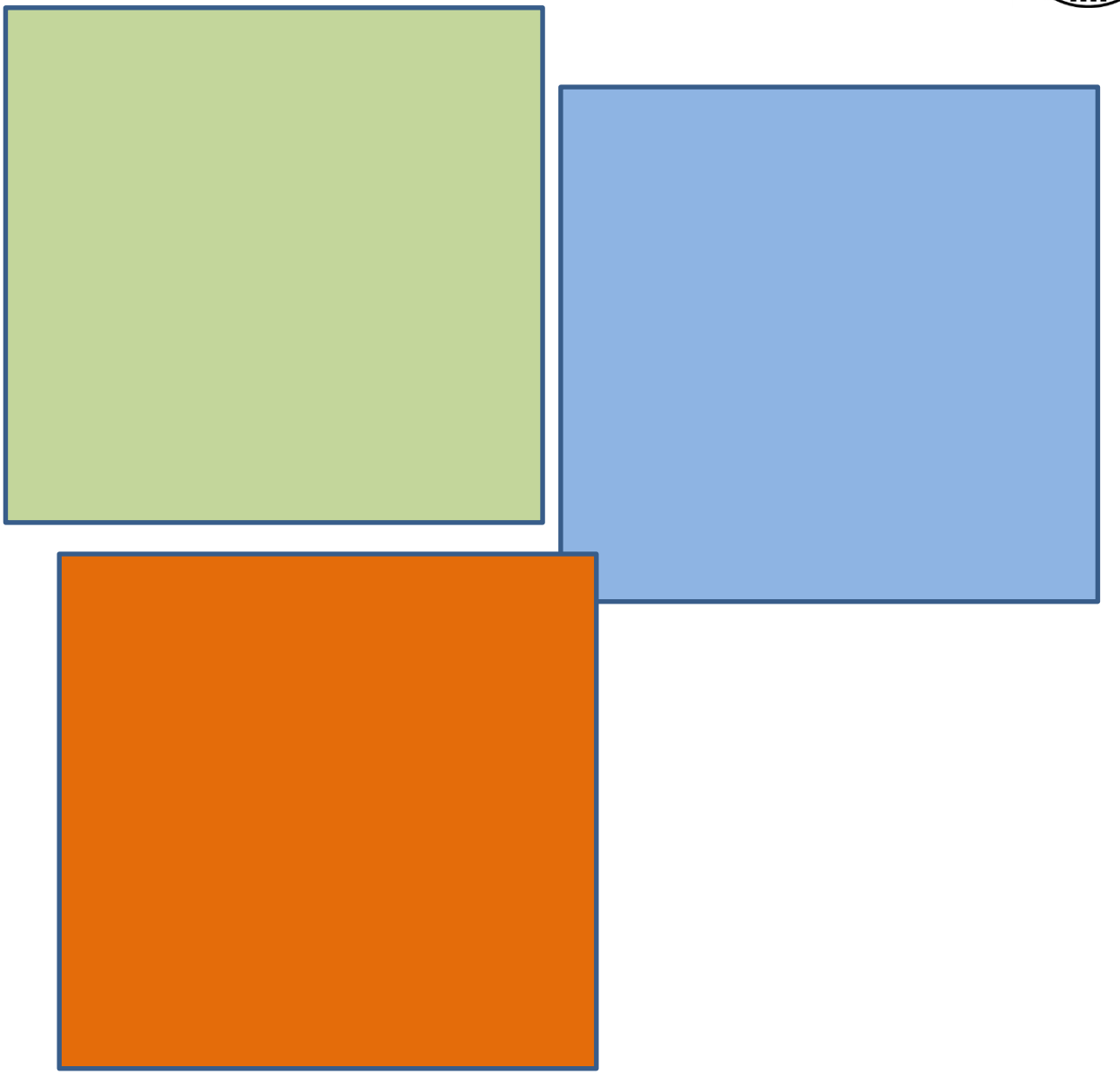
明治大学総合数理学部
先端メディアサイエンス学科
中村研究室





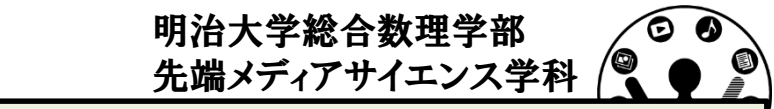
インスタンス化

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

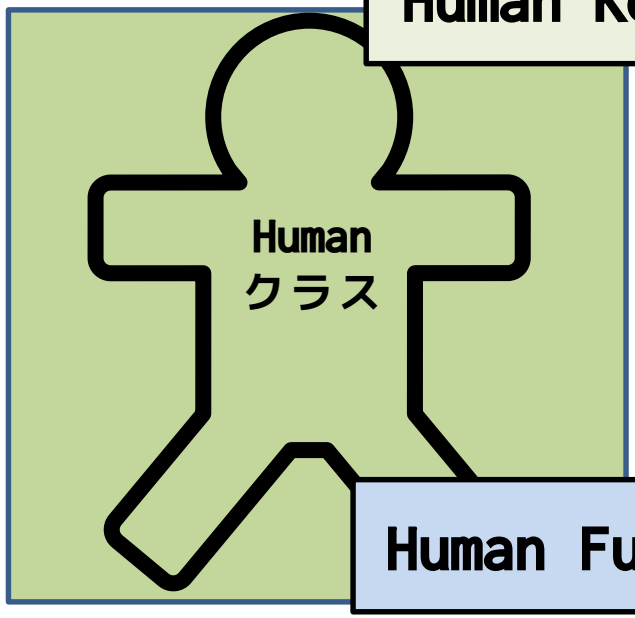


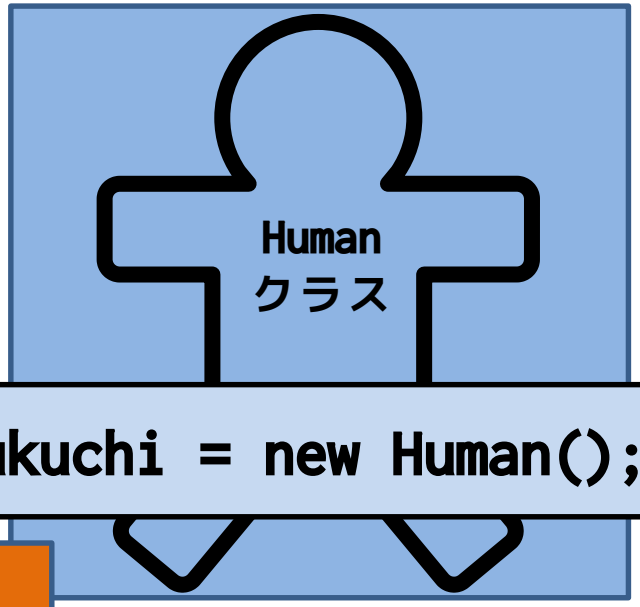
インスタンス化

明治大学総合数理学部
先端メディアサイエンス学科

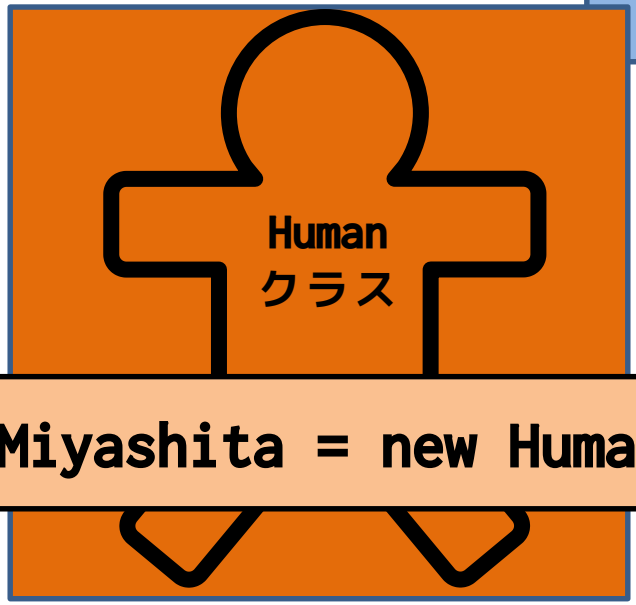


```
Human Komatsu = new Human();
```






```
Human Fukuchi = new Human();
```

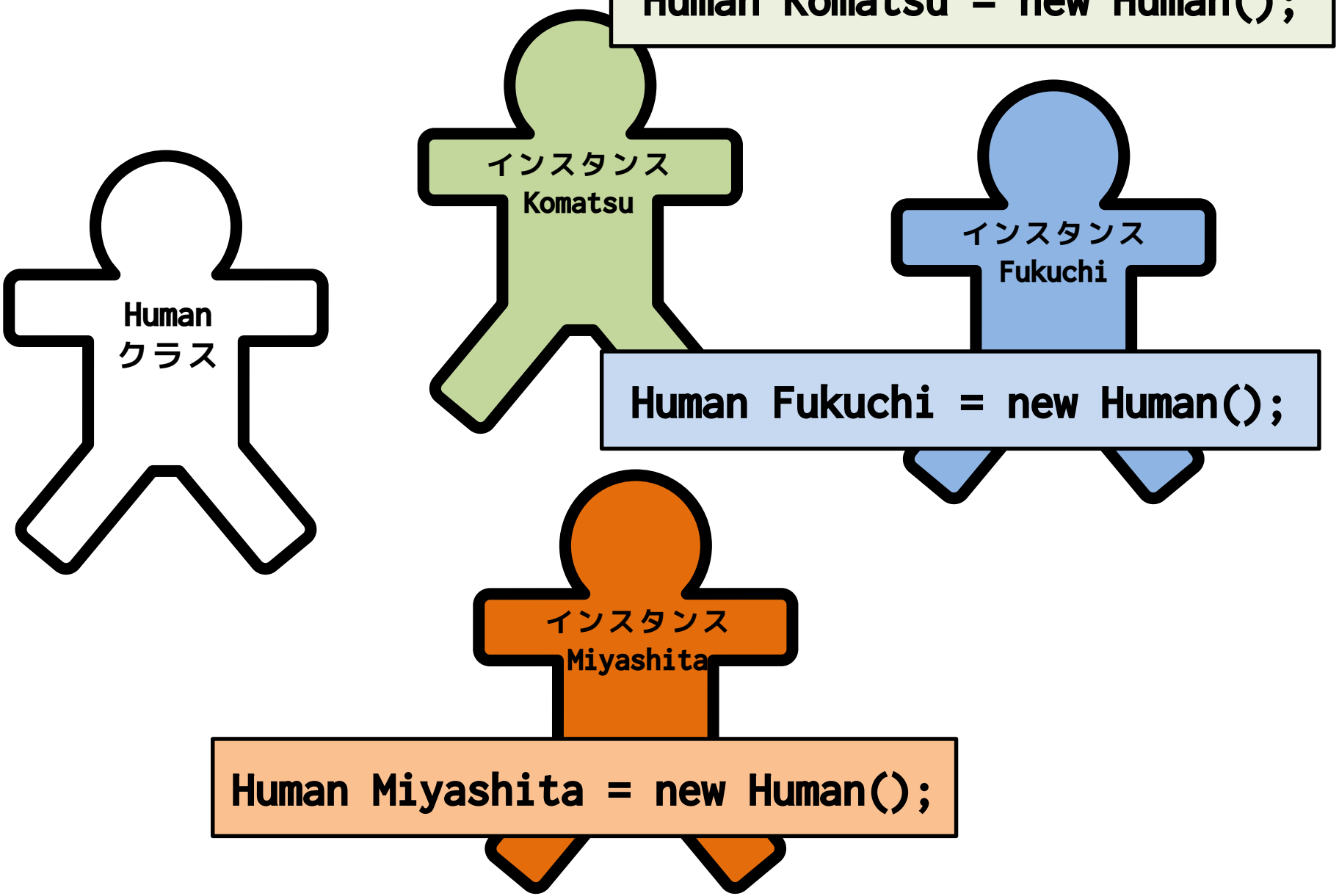



```
Human Miyashita = new Human();
```

インスタンス化

明治大学総合数理学部
先端メディアサイエンス学科

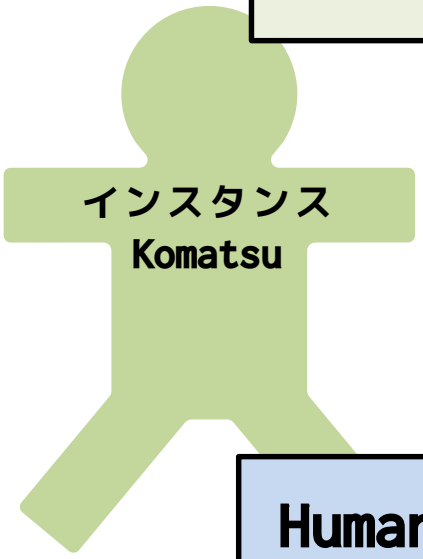
```
Human Komatsu = new Human();
```

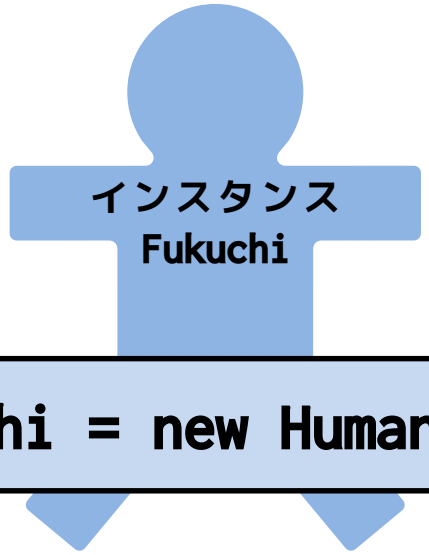


インスタンス化

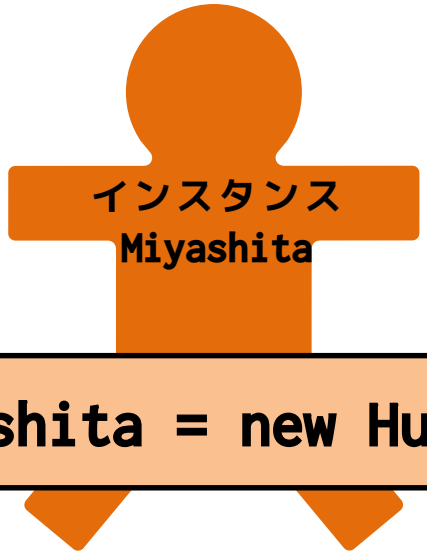
明治大学総合数理学部
先端メディアサイエンス学科


```
Human Komatsu = new Human();
```





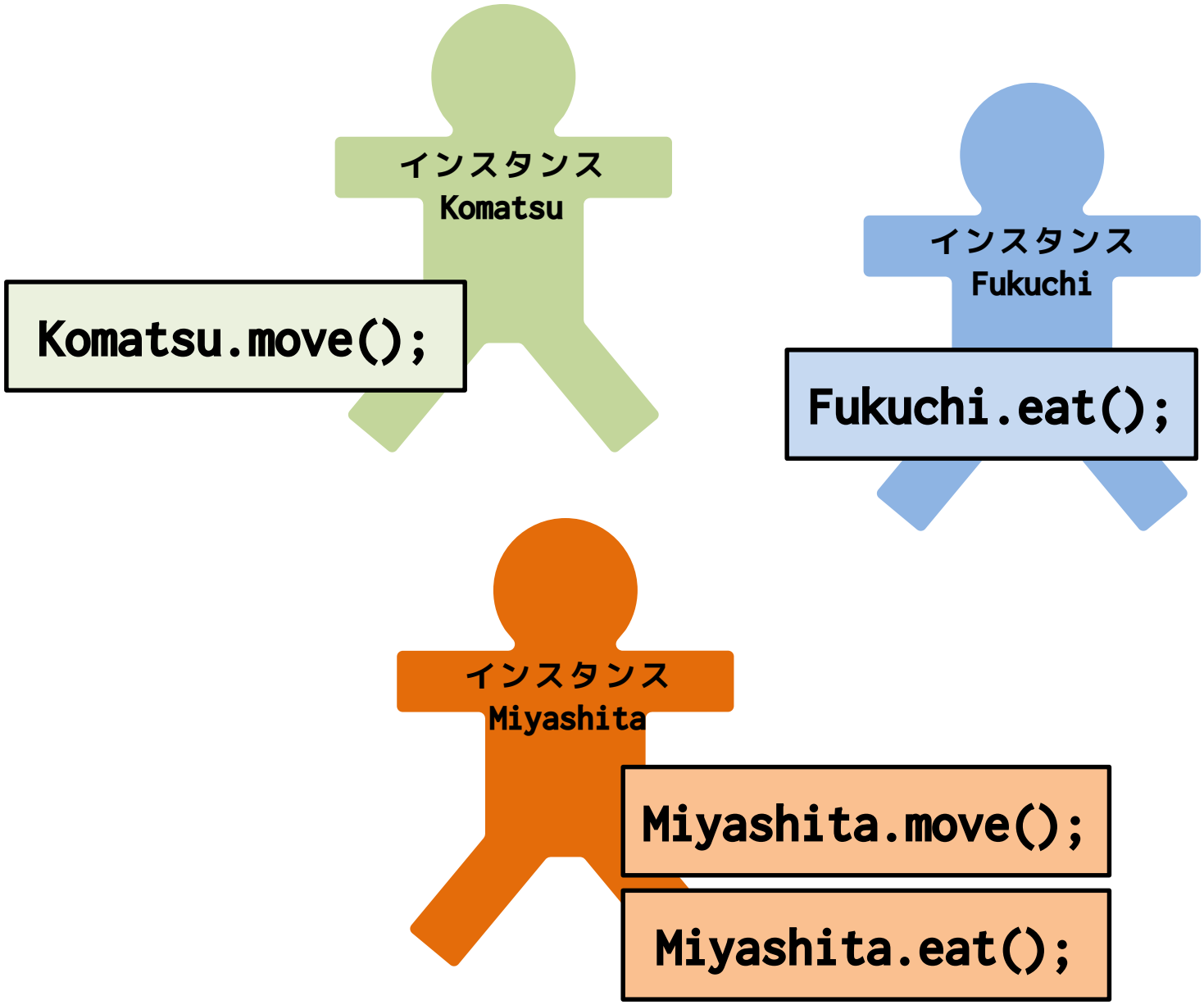

```
Human Fukuchi = new Human();
```




```
Human Miyashita = new Human();
```

使い方

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室



クラス

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 財布クラスを作るにはどうする？
  - 内部的に持つ情報は？ 必要なメソッドは？
- 人を管理するクラスを作るにはどうする？
  - 姓，名，年齢
- ペイントツールで1つずつのストロークを管理するには？
- 文字列を扱うにはどうするか？

他の例

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 時計というクラスを作る
  - 時計のインスンタンス変数
    - 現在時間（時分秒）の情報
    - 目覚ましのON/OFF状態管理変数
    - 目覚ましの設定時間
  - 時計のインスタンスメソッド
    - 分変更メソッド
    - 時計停止メソッド（秒針停止）
    - 目覚まし設定時間変更メソッド
    - 目覚ましのON/OFF切り替えメソッド

先週のまとめ

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- オブジェクト指向のさわりを学んだ
  - 色んなクラス
    - ArrayList, PImage, Minim, AudioPlayer, String
  - インスタンス化
    - Human nkmr = new Human();
  - インスタンス
    - nkmr
  - インスタンス変数
    - nkmr.speed
  - インスタンスメソッド
    - nkmr.move()
  - コンストラクタ
    - Human(){ 初期化処理 }

定義したクラスの使い方

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- クラスの中で変数を定義すると，そのクラス内の変数として使うことができる
  - クラス内で変数を定義
  - クラスを使う場合は new！
  - クラス内の変数を使う場合はドットでつなぐ

```
class Ball
{
  float posX;
  float posY;
  float speedX;
  float speedY;
}

Ball ball;
ball = new Ball();
ball.posX += ball.speedX;
ball.posY += ball.speedY;
```

インスタンス化

クラス名　変数名 = new クラス名();

クラスの定義

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 財布クラスを作るにはどうする？
  - 内部的に持つ情報は？ 必要なメソッドは？
- 人を管理するクラスを作るにはどうする？
  - 姓，名，年齢
- ペイントツールで1つずつのストロークを管理するには？
- 文字列を扱うにはどうするか？

宿題2-1：hw_listBall

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- ウインドウ800x800を用意し，Ballクラスを利用してマウスクリックするたびに丸が生まれ，上下左右にランダムな速度で動き，上下左右の端で跳ね返るプログラムを作成せよ．なお，クリックでいくらでも作成できるようにせよ
- また，Ballクラスを改良してそれぞれの丸の色は初期値としてランダムに設定するようにせよ

  ［ヒント］色情報を持つインスタンス変数を追加する
  - int red;
  - int green;
  - int blue;
- たくさん増やして管理するには ArrayListを使う！

クリックした場所に登場させたい！

コンストラクタで定義

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- インスタンスを作るときに初期位置を
  マウスの位置に！

**コンストラクタ内で
mouseX，mouseYを使う？**

```
class Ball{
  float posX;
  float posY;
  float speedX;
  float speedY;

  Ball(){
    posX = mouseX;
    posY = mouseY;
    speedX = random(1,5);
    speedY = random(1,5);
  }
```

- これだと色々使い回せないので困る！！！
- 普通に作ったときにおかしくなる

コンストラクタで定義！

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- インスタンスを作るときに初期位置を明示的に決めたい！

  new Ball(mouseX, mouseY)

  と指定したい！

コンストラクタとして座標を指定できるものを用意する！

```
class Ball{
  float posX;
  float posY;
  float speedX;
  float speedY;

  Ball(){
    posX = random(width);
    posY = random(height);
    speedX = random(1,5);
    speedY = random(1,5);
  }

  Ball(float x, float y){
    posX = x;
    posY = y;
    speedX = random(1,5);
    speedY = random(1,5);
  }
```

new Ball(mouseX, mouseY)

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- クリックするたびにその場所からBallがうまれる！

```
ArrayList<Ball> balls = new ArrayList<Ball>();

void setup() {
  size(400, 300);
}

void draw(){
  background(255);
  for(int i=0; i<balls.size(); i++){
    balls.get(i).move();
    balls.get(i).display();
  }
}

void mousePressed(){
  Ball ball = new Ball(mouesX, mouseY);
  balls.add( ball );
}
```

newで指定！

オーバーロード！

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- Ball(){…}とBall(float x, float y){…}と2つあってもよいの？
  - 引数の数とか型が違えば大丈夫！！
  - new のときに，どう呼び出すかの違い

    Ball() は Ball b = new Ball();

    Ball(float x, float y) は
    Ball b = new Ball(500.0, 100.0);

    で，それぞれ呼び出される．

オーバーロード（多重定義）
同じ関数の名前だけれど，引数の数や型を変えて振る舞いを変えることができるもの

動物園の動物クラス群

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 猫，犬，猿，象，熊を定義
  - それぞれの座標は意識したくない
    - cat.x, cat.y, dog.x, dog.y, monkey.x, monkey.y, ...
    - 内部で適当に処理してもらう
  - 描画はシンプルにしたい
    - cat.draw(), dog.draw(), monkey.draw(), elephant.draw(), ...
  - 移動もシンプルにしたい
    - cat.move(), dog.move(), monkey.move(), elephant.move(), ...
  - 睡眠も任せてしまう
    - cat.sleep(), dog.sleep(), monkey.sleep(), elephant.sleep(), ...

いいんだけど，重複があるのでは？

課題2-2: basic_boundAll2

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- Ball クラスを改良し，xが動き回るCrossクラスと△が動き回るTriangleクラスを作成せよ
- またこれを利用して5個の〇と，4個のxと，3個の△が動き回るプログラムを作成せよ
  - ただし，その速度はx、y方向それぞれ-5～5の実数値とせよ
  - また，〇は壁で跳ね返り，xと△は跳ね返らずに反対側から出てくるようにせよ

Ballを改良しCrossを作る

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

```
class Cross{
  float posX;
  float posY;
  float speedX;
  float speedY;

  Cross(){
    posX = random(width);
    posY = random(height);
    speedX = random(-5, 5);
    speedY = random(-5, 5);
  }

  void display(){
    line(x-15, y-15, x+15, y+15);
    line(x+15, y-15, x-15, y+15);
  }
```

```
  void move(){
    posX += speedX;
    posY += speedY;
    if(posX > width){
      posX = width * 2 - posX;
      speedX = -speedX;
    }
    if(posX < 0){
      posX = -posX;
      speedX = -speedX;
    }
    if(posY > height){
      posY = height * 2 - posY;
      speedY = -speedY;
    }
    if(posY < 0){
      posY = -posY;
      speedY = -speedY;
    }
  }
}
```

Crossクラスを使うと

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

```
Cross cross1;
Cross cross2;
Cross cross3;
void setup() {
  size( 400, 300 );
  cross1 = new Cross();
  cross2 = new Cross();
  cross3 = new Cross();
}

void draw() {
  background(255);
  cross1.move();
  cross2.move();
  cross3.move();
  cross1.display();
  cross2.display();
  cross3.display();
}
```

ここで・・・

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- Ballクラスと，Crossクラスと，Triangleクラスは似ている

クラス名	Ball	Cross	Triangle	
インスタンス変数	posX, posY speedX, speedY	posX, posY speedX, speedY	posX, posY speedX, speedY	一致
コンストラクタ	Ball() 座標速度初期化	Cross() 座標速度初期化	Triangle() 座標速度初期化	名は違うが 内部は一致
void move()	はねかえる	はねかえらない	はねかえらない	一部一致
void display()	円を描く	xを描く	△を描く	違う

無駄じゃね？
もっと手軽にできないの？

どう無駄をなくす？

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- そもそもBallクラスというのがダメなのでは？
  - Objectクラスという名前にして、objectTypeなどの変数を用意し、displayの時に切り替えては？

```
class Object {
  float posX;
  float posY;
  float speedX;
  float speedY;
  int objectType;

  void display(){
    if(objectType == 0){
      ellipse(posX, posY, 30, 30);
    } else if(objectType == 1){
      rect(posX, posY, 30, 30);
    }
  }
```

これも一つの方法だが
跳ね返りでも条件が必要で
複雑なものだと厳しくなる

そこで継承！

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 大辞林 第三版
  1. 先の人の身分・権利・義務・財産などを受け継ぐこと。「王位を－する」
  2. インヘリタンス→（オブジェクト指向プログラミングにおいて，クラス間でデータの共有を行う機構。新しく定義するクラスを既存のクラスの下位クラスとして記述し，上位クラスより属性やメソッドを引き継ぐ仕組みをいう。上位クラスに対する差分のみを記述するだけで新しいクラスを定義することが可能となる。）

一緒の部分をまとめた
スーパークラス（親クラス）を作る

ここで・・・

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- ObjectBaseというクラスを作る
  - その機能を，BallやCross，Triangleに提供する

クラス名	Ball	Cross	Triangle	ObjectBase
インスタンス変数	posX，posY speedX，speedY	posX，posY speedX，speedY	posX，posY speedX，speedY	posX，posY speedX，speedY
コンストラクタ	Ball() 座標速度初期化	Cross() 座標速度初期化	Triangle() 座標速度初期化	ObjectBase() 座標速度初期化
void move()	はねかえる	はねかえらない	はねかえらない	はねかえらない
void display()	円を描く	xを描く	△を描く	何も描かない

ObjectBaseクラス

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

```
class ObjectBase {
  float posX;
  float posY;
  float speedX;
  float speedY;

  ObjectBase(){
    posX = random(width);
    posY = random(height);
    speedX = random(-5, 5);
    speedY = random(-5, 5);
  }

  void display(){
  }
```

```
  void move(){
    posX = posX + speedX;
    posY = posY + speedY;
    if(posX > width){
      posX = posX - width;
    }
    if(posX < 0){
      posX = posX + width;
    }
    if(posY > height){
      posY = posY - height;
    }
    if(posY < 0){
      posY = posY + height;
    }
  }
}
```

display() は何もしないので空っぽにする

一緒の部分をまとめる

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- BallはObjectBaseの変数（posX，posY，speedX，speedY）や機能（初期化）をもち，独自の移動と表示に関する機能（メソッド）をもつクラス
  - はねかえる＋円の描画
- CrossとTriangleはObjectBaseの変数（posX，posY，speedX，speedY）や機能（移動や初期化）をもち，独自の表示に関する機能（メソッド）をもつクラス
  - バツまたは三角形の描画

- インスタンス変数や，インスタンスメソッドを引き継ぐことを継承と呼ぶ！

継承

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 継承すると，親の力をすべて引き継ぐ！
- 継承の方法は extends とやるだけ！

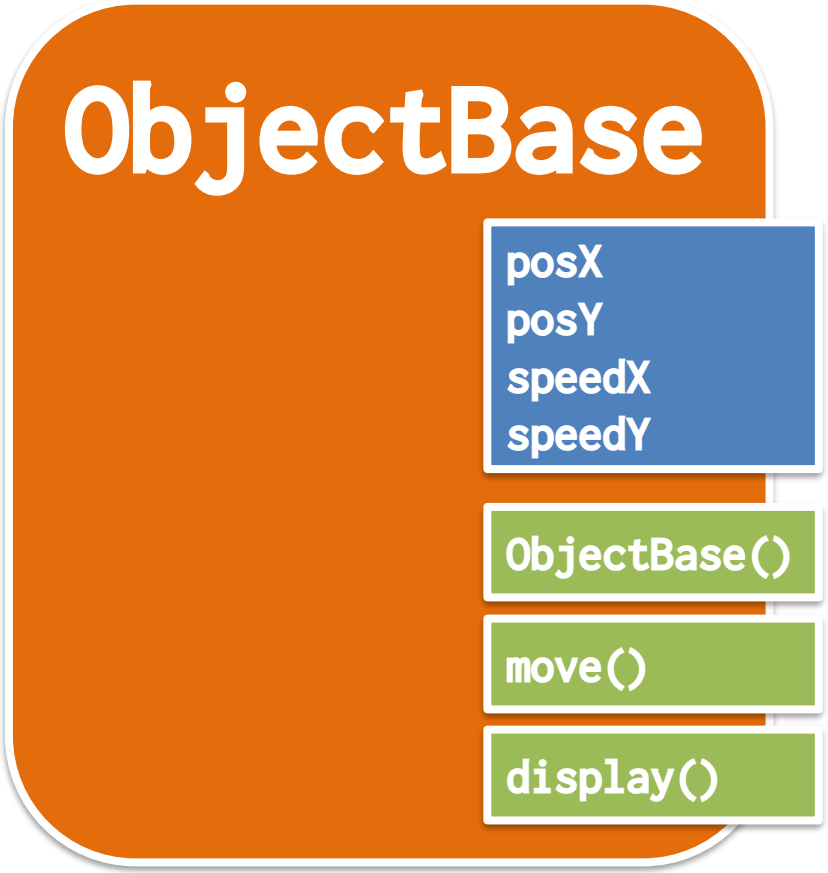
```
class クラス名 extends 親クラス名 {

}
```

- 継承により親の能力，値はすべて引き継ぎます

Crossをどう作る？

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

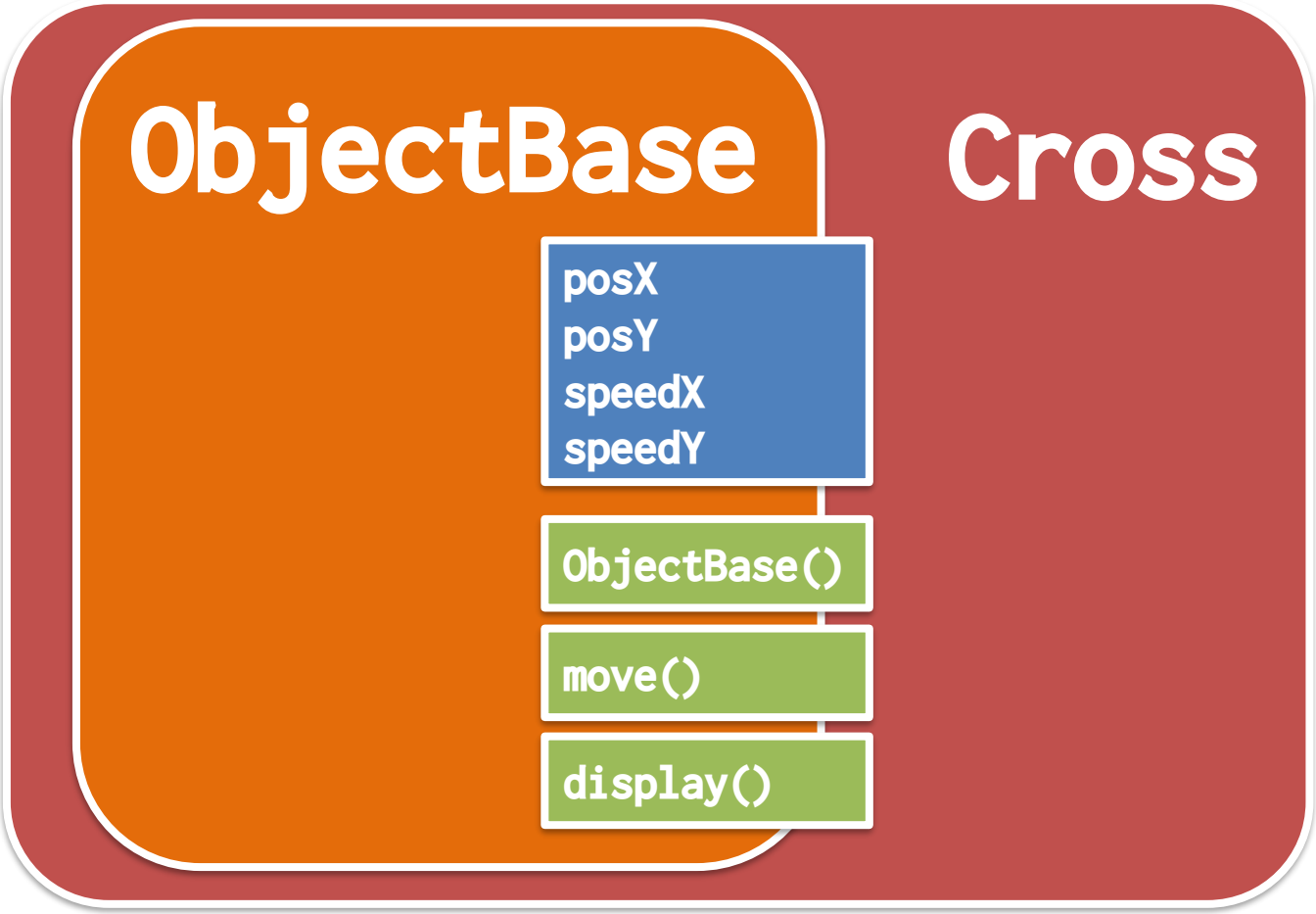
- ObjectBaseクラスを継承してCrossクラスを作る！



Crossをどう作る？

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

```
class Cross extends ObjectBase
{
}
```



使ってみよう！

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

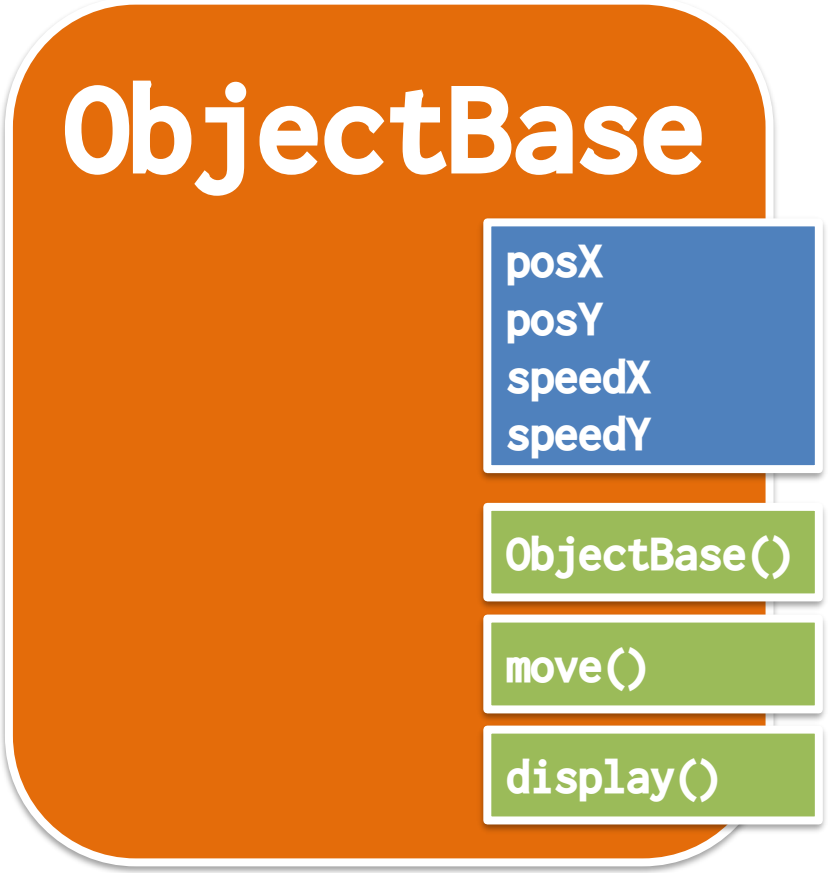
```
Cross cross1;
Cross cross2;

void setup() {
  size( 400, 300 );

  cross1 = new Cross();
  cross2 = new Cross();
}

void draw() {
  background(255);

  cross1.move();
  cross2.move();
  cross1.display();
  cross2.display();
}
```

- なにも表示されない
  - なんで？
- ObjectBaseのdisplay()は何も描画しないから！
  - バツを描画するにはどうしたら良い？
  - ObjectBaseのdisplayを書き換える？
    - → だめ！！

Crossをどう作る？

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

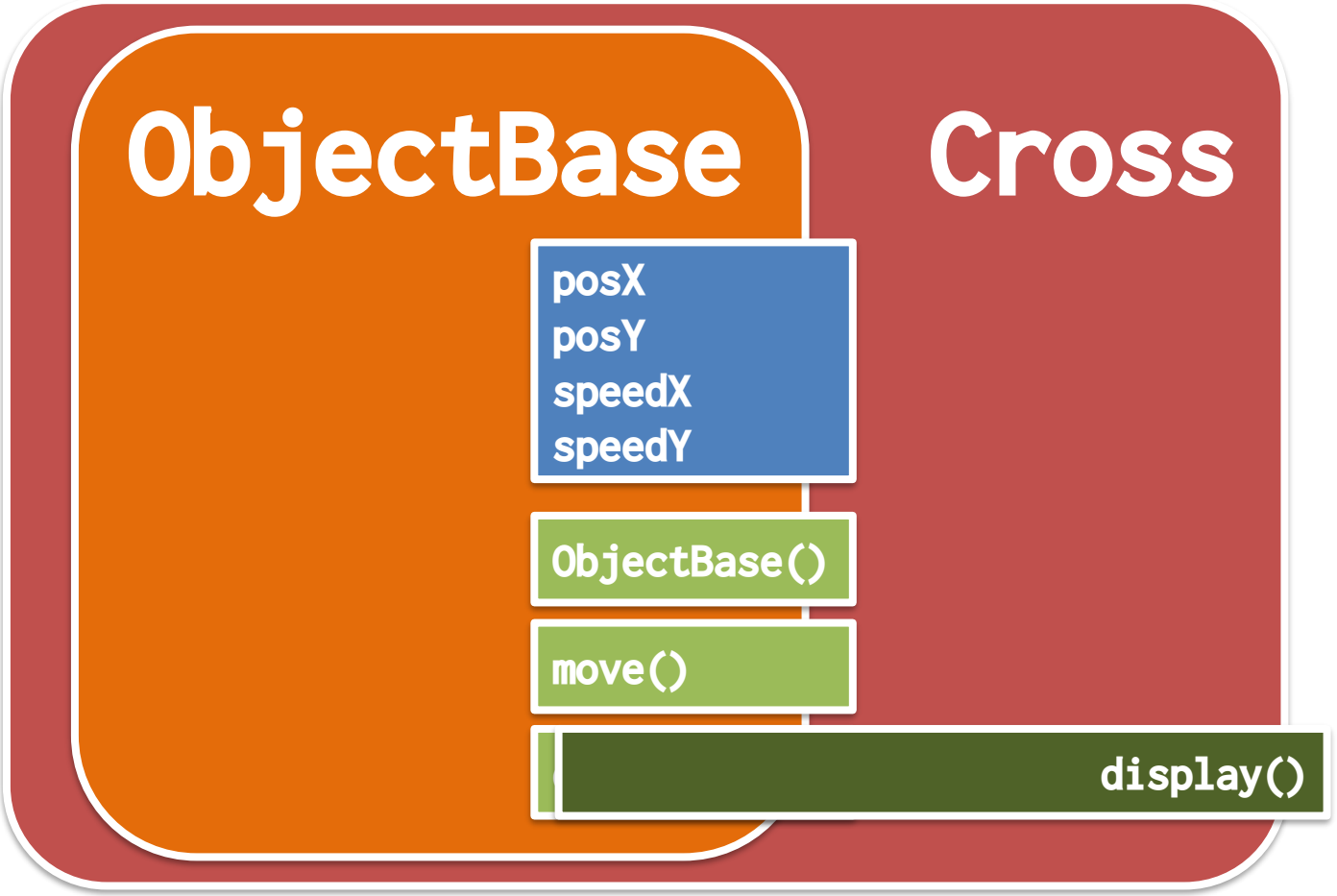
- ObjectBase の変数やメソッドを引き継ぎつつ，必要なところを上書きする！



Crossをどう作る？

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- display() を上書きしてしまう！
  - displayメソッドをオーバーライドする



ObjectBaseクラスを使う

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

```
class Cross extends ObjectBase
{
  void display(){
    line(posX-15, posY-15, posX+15, posY+15);
    line(posX+15, posY-15, posX-15, posY+15);
  }
}
```

displayをオーバーライドして
親のdisplayメソッドが
呼ばれないようにする

Crossクラスが劇的に短く！

使ってみよう！

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

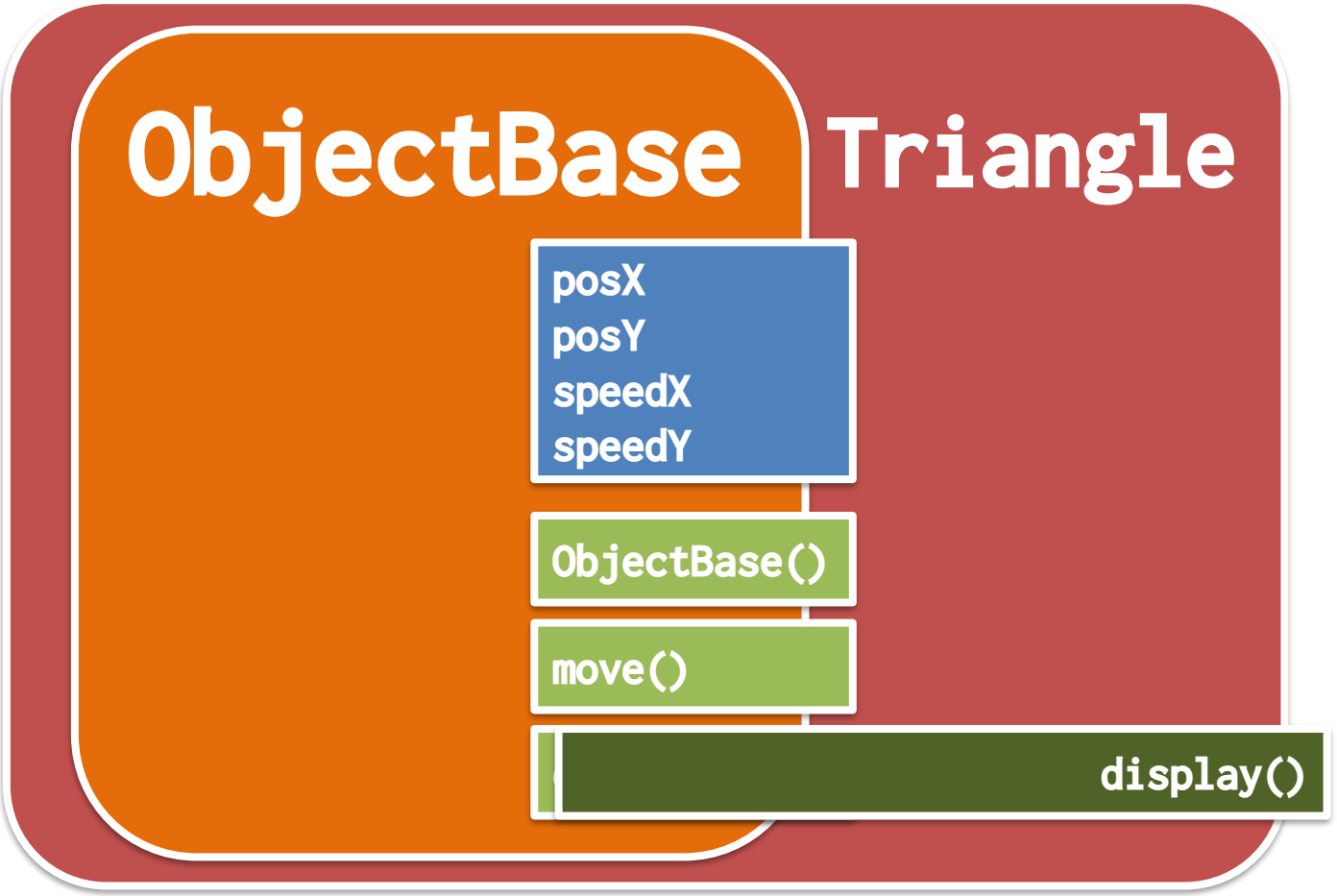
```
Cross cross1;
Cross cross2;

void setup() {
  size( 400, 300 );

  cross1 = new Cross();
  cross2 = new Cross();
}

void draw() {
  background(255);

  cross1.move();
  cross2.move();
  cross1.display();
  cross2.display();
}
```

- 表示された！
  - display()が上書きされている！

Triangleをどう作る？

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- display() を上書きしてしまう！
  - displayメソッドをオーバーライドする



ObjectBaseクラスを使う

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

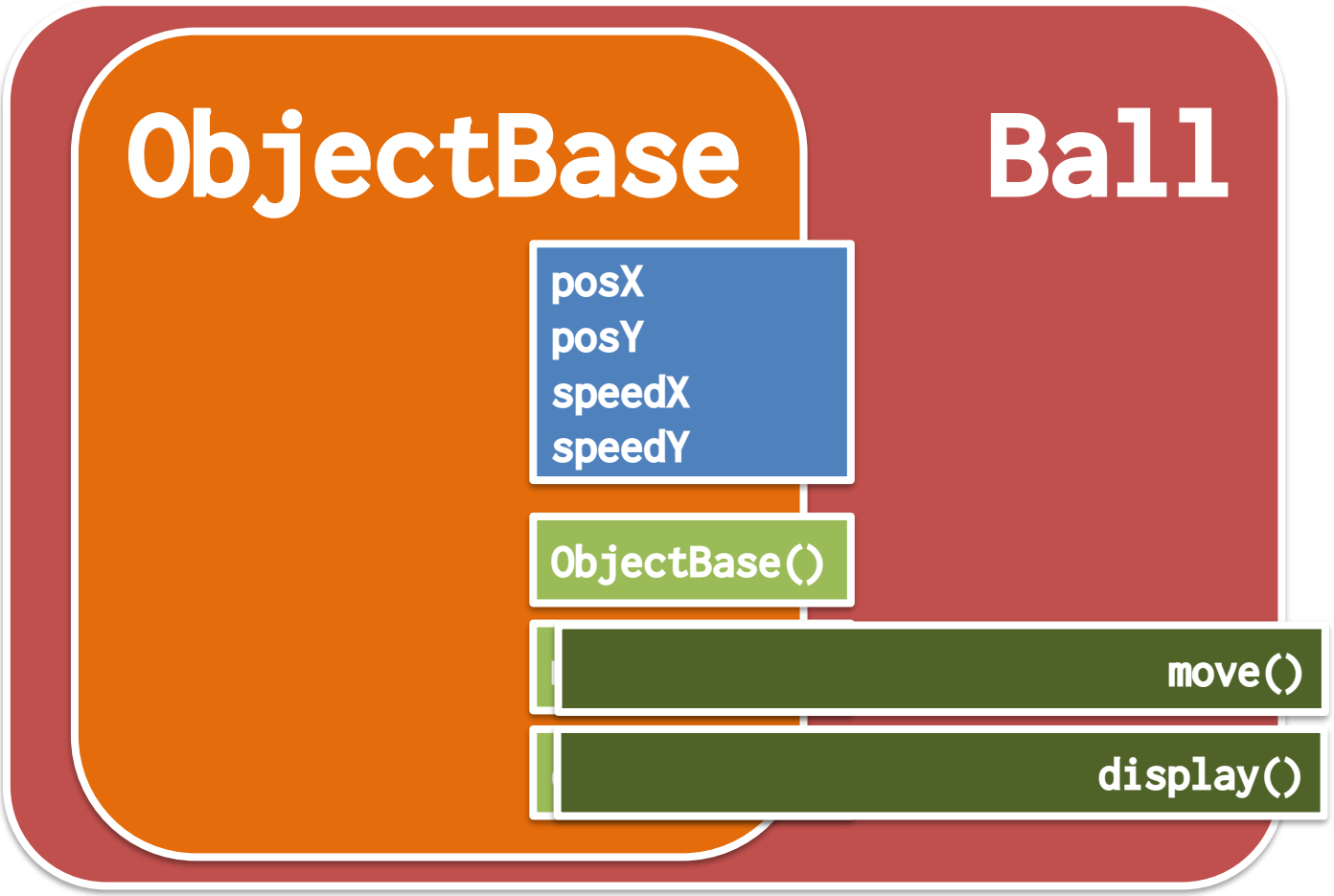

```
class Triangle extends ObjectBase
{
  void display(){
    fill(0, 255, 0);
    triangle(posX, posY-15, posX+15, posY+15, posX-15, posY+15);
  }
}
```

displayをオーバーライドして
親のdisplayメソッドが
呼ばれないようにする

Triangleクラスが劇的に短く！

Ball をどう作る？

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室



- move() と display() をオーバーライド！

Ballクラスはどう作る？

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

```
class Ball extends ObjectBase
{
  void display(){
    fill(255, 0, 0);
    ellipse(posX, posY, 30, 30);
  }
```

```
  void move(){
    posX += speedX;
    posY += speedY;
    if(posX > width){
      posX = width * 2 - posX;
      speedX = -speedX;
    }
    if(posX < 0){
      posX = -posX;
      speedX = -speedX;
    }
    if(posY > height){
      posY = height * 2 - posY;
      speedY = -speedY;
    }
    if(posY < 0){
      posY = -posY;
      speedY = -speedY;
    }
  }
}
```

**display と move を
オーバーライドして
親の move と display が
呼ばれないようにする**

ObjectBaseクラスを使う

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 使うときは，継承していることは気にしないで，対象とするクラスを使うだけでよい！
  - move()
  - display()

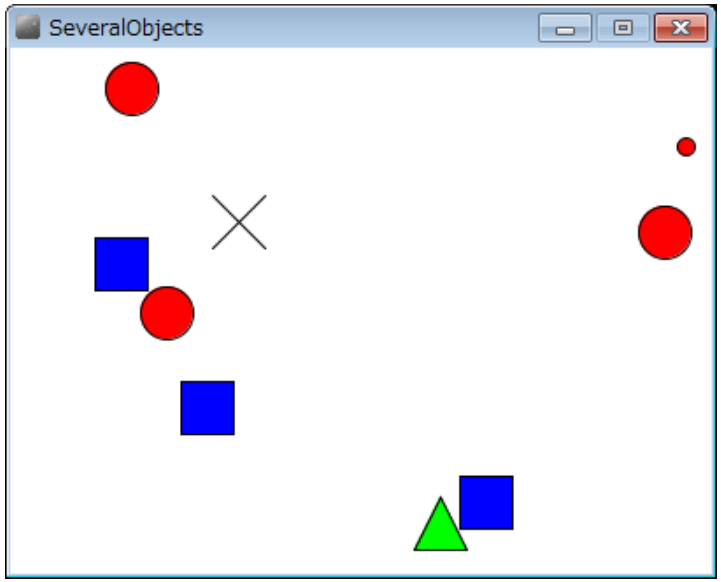
  - すばらしい！！

```
Ball ball;
Cross cross;
Triangle triangle;

void setup() {
  size( 400, 300 );
  ball = new Ball();
  cross = new Cross();
  triangle = new Triangle();
}

void draw() {
  background(255);
  ball.move();
  cross.move();
  triangle.move();
  ball.display();
  cross.display();
  triangle.display();
}
```

じゃあ

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- ObjectBase クラスを継承して青色の四角形を描画するSquareクラスを作るには？
  - 四角形は左右の壁は跳ね返り，上下の壁はすり抜けて反対側から出てくる



四角形のクラス

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

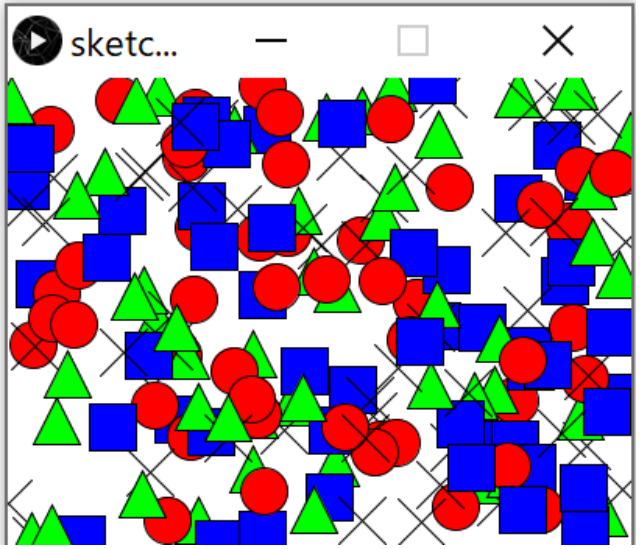
- 動く青色の四角形のクラス

```
class Square extends ObjectBase
{
  void display(){
    fill(0, 0, 255);
    rect(posX-15, posY-15, 30, 30);
  }
  void move(){
    posX += speedX;
    posY += speedY;
    if(posX > width-15){
      posX = width-15;
      speedX = -speedX;
    }
    if(posX < 15){
      posX = 15;
      speedX = -speedX;
    }
    if(posY > height) posY = posY - height;
    if(posY < 0)      posY = posY + height;
  }
}
```

50個の○△□xを描画！

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 50個の○と，50個のxと，50個の△と，50個の□が動き回るプログラムを作成せよ
  - ただし，その速度はx、y方向それぞれ-5～5の実数値とせよ
  - また，○は壁で跳ね返り，xと△と跳ね返らずに反対側から出てくるように，□は上下では跳ね返らず反対側から出てきて左右では跳ね返るようにせよ



配列＋クラス

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室


```
Ball[] balls = new Ball[50];
Cross[] crosses = new Ball[50];
Triangle[] triangles = new Triangle[50];
Square[] squares = new Square[50];

void setup() {
  size(400, 300);
  for (int i=0; i<50; i++) {
    balls[i] = new Ball();
    crosses[i] = new Cross();
    triangles[i] = new Triangle(
    squares[i] = new Square();
  }
}
```

```
void draw() {
  background(255);
  for(int i=0; i<50; i++){
    balls[i].move();
    balls[i].display();
    crosses[i].move();
    crosses[i].display();
    triangles[i].move();
    triangles[i].display();
    squares[i].move();
    squares[i].display();
  }
}
```

ArrayList＋クラス

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室


```
ArrayList<Ball> balls = new ArrayList<Ball>();
ArrayList<Cross> crosses = new ArrayList<Cross>();
ArrayList<Triangle> triangles = new ArrayList<Triangle>();
ArrayList<Square> squares = new ArrayList<Square>();

void setup() {
  size(400, 300);
  for (int i=0; i<50; i++) {
    balls.add(new Ball());
    crosses.add(new Cross());
    triangles.add(new Triangle());
    squares.add(new Square());
  }
}
```

```
void draw() {
  background(255);
  for(int i=0; i<50; i++){
    balls.get(i).move();
    balls.get(i).display();
    crosses.get(i).move();
    crosses.get(i).display();
    triangles.get(i).move();
    triangles.get(i).display();
    squares.get(i).move();
    squares.get(i).display();
  }
}
```

ArrayListでもあまり改善されてない？

ポリモーフィズム

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- Ball, Cross, Triangle, Squareの親クラス（継承元）はObjectBaseで，同じメソッドを持っている
- ポリモーフィズム（多態性，多様性）
  - 複数の型に属することを許すこと
  - 親クラスから継承されたクラスのインスタンスは，それぞれ親クラスの型に代入できる．呼び出されるのは，継承されたクラスのメソッド
  - つまり，ObjectBaseとして定義しておき，BallもCrossもTriangleもSquareも放り込んで，ただのメソッドで呼び出しが可能！

配列＋ObjectBase

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- ObjectBaseにはBallもCrossもTriangleもSquareも入れることができるよ！

- ポリモーフィズムバンザイ！

```
ObjectBase[] list = new ObjectBase[200];

void setup() {
  size(400, 300);
  for (int i=0; i<list.length; i+=4) {
    list[i+0] = new Ball();
    list[i+1] = new Cross();
    list[i+2] = new Triangle();
    list[i+3] = new Square();
  }
}

void draw() {
  background(255);
  for(int i=0; i<list.length; i++){
    list[i].move();
    list[i].display();
  }
}
```

ArrayList＋ObjectBase

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室


```
ArrayList<ObjectBase> list = new ArrayList<ObjectBase>();

void setup() {
  size(400, 300);
  for(int i=0; i<50; i++){
    list.add( new Ball() );
    list.add( new Cross() );
    list.add( new Triangle() );
    list.add( new Square() );
  }
}

void draw() {
  background(255);
  for( int i=0; i<list.size(); i++ ){
    list.get(i).move();
    list.get(i).display();
  }
}
```

<ObjectBase> で型を定義

BallもCrossもTriangleもSquareも
入れることができるよ！

list.size()で数を取得

list.get(i)でi番目を取得

ArrayList＋ObjectBase

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室


```
ArrayList list = new ArrayList();

void setup() {
  size(400, 300);
  for(int i=0; i<50; i++){
    list.add( new Ball() );
    list.add( new Cross() );
    list.add( new Triangle() );
    list.add( new Square() );
  }
}

void draw() {
  background(255);
  for(int i=0; i<list.size(); i++){
    ObjectBase obj = (ObjectBase)list.get(i);
    obj.move();
    obj.display();
  }
}
```

型を定義しなくても使えます
その場合は使うときにキャストを！

(ObjectBase) でキャスト

ArrayList＋拡張for文

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室


```
ArrayList<ObjectBase> list = new ArrayList<ObjectBase>();

void setup() {
  size(400, 300);
  for(int i=0; i<50; i++){
    list.add( new Ball() );
    list.add( new Cross() );
    list.add( new Triangle() );
    list.add( new Square() );
  }
}

void draw() {
  background(255);
  for( ObjectBase obj: list ){
    obj.move();
    obj.display();
  }
}
```

めっちゃシンプル！

何に使えるの？

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 継承とかポリモーフィズムとかなんか色々あるけど何に使うの？
  - シューティングで色んな動きをする敵を作る
  - RPGゲームで色んな戦い方をする敵を作る
  - 色々な光り方・広がり方をする花火を作る
  - 色々な結晶の雪を各々の特性で降らせる
  - 草原に色々な草木花を生えさせる
  - ひとの混雑に関するシミュレーションをする

親クラスを定義して継承してクラスを作り
ArrayListで意識せずまとめて動かす！

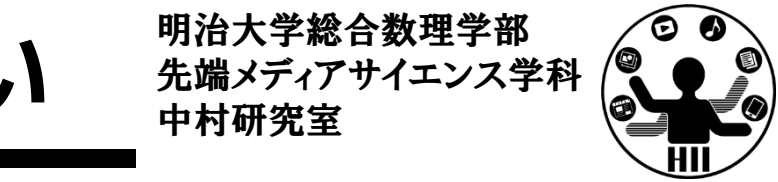
覚える必要はないけれど

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

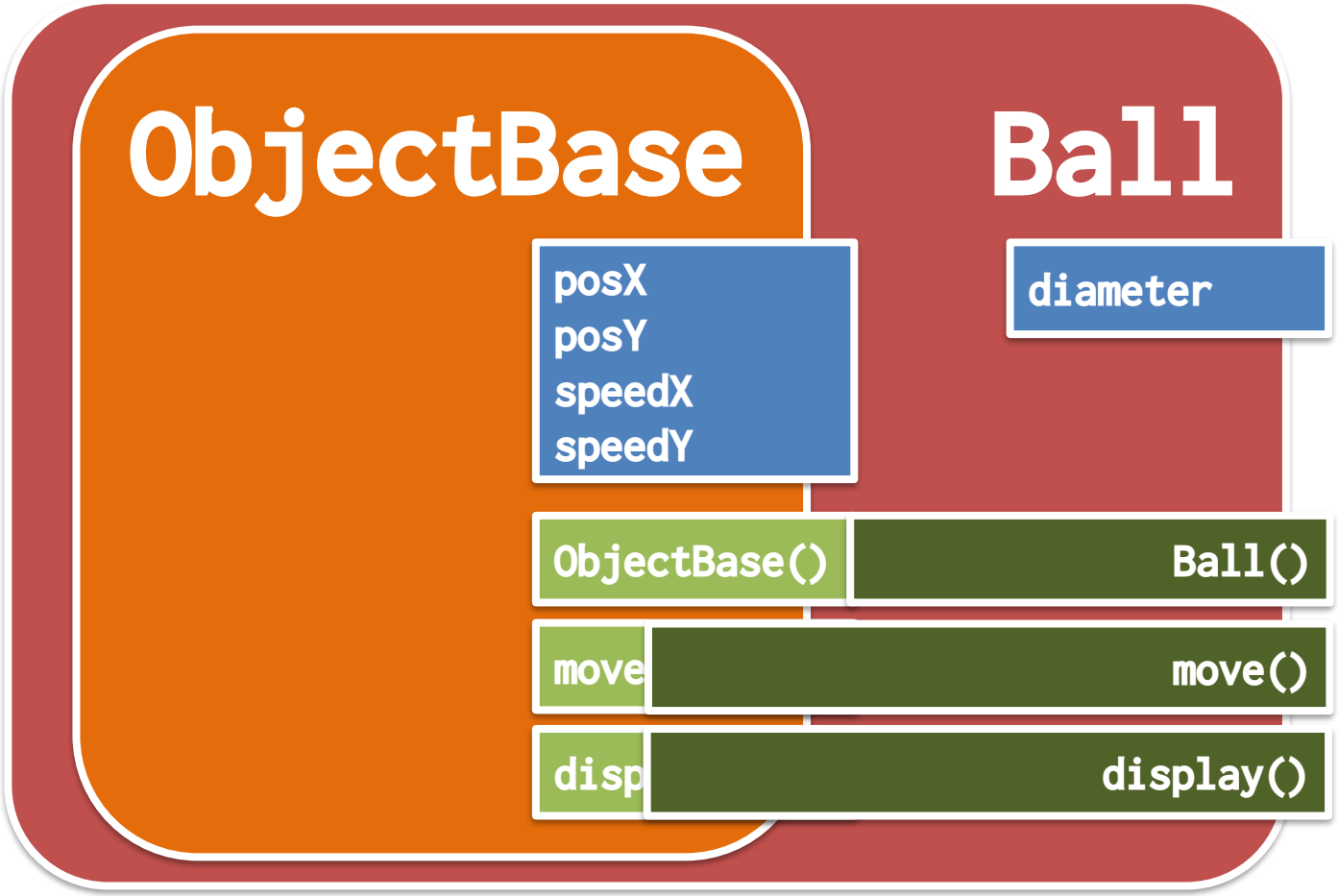
- アップキャスト
  - 親クラスで定義されたものに，継承されたクラスのインスタンスが代入されると，自動的に親の型に変換されること
- ちなみに，ダウンキャストもあるよ
  - ダウンキャストは安全じゃないよ！

継承先で変数追加したい

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室



- ObjectBaseを継承するけど，○だけはその直径（diameter）もランダムに決定したい



継承先で変数追加したい

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

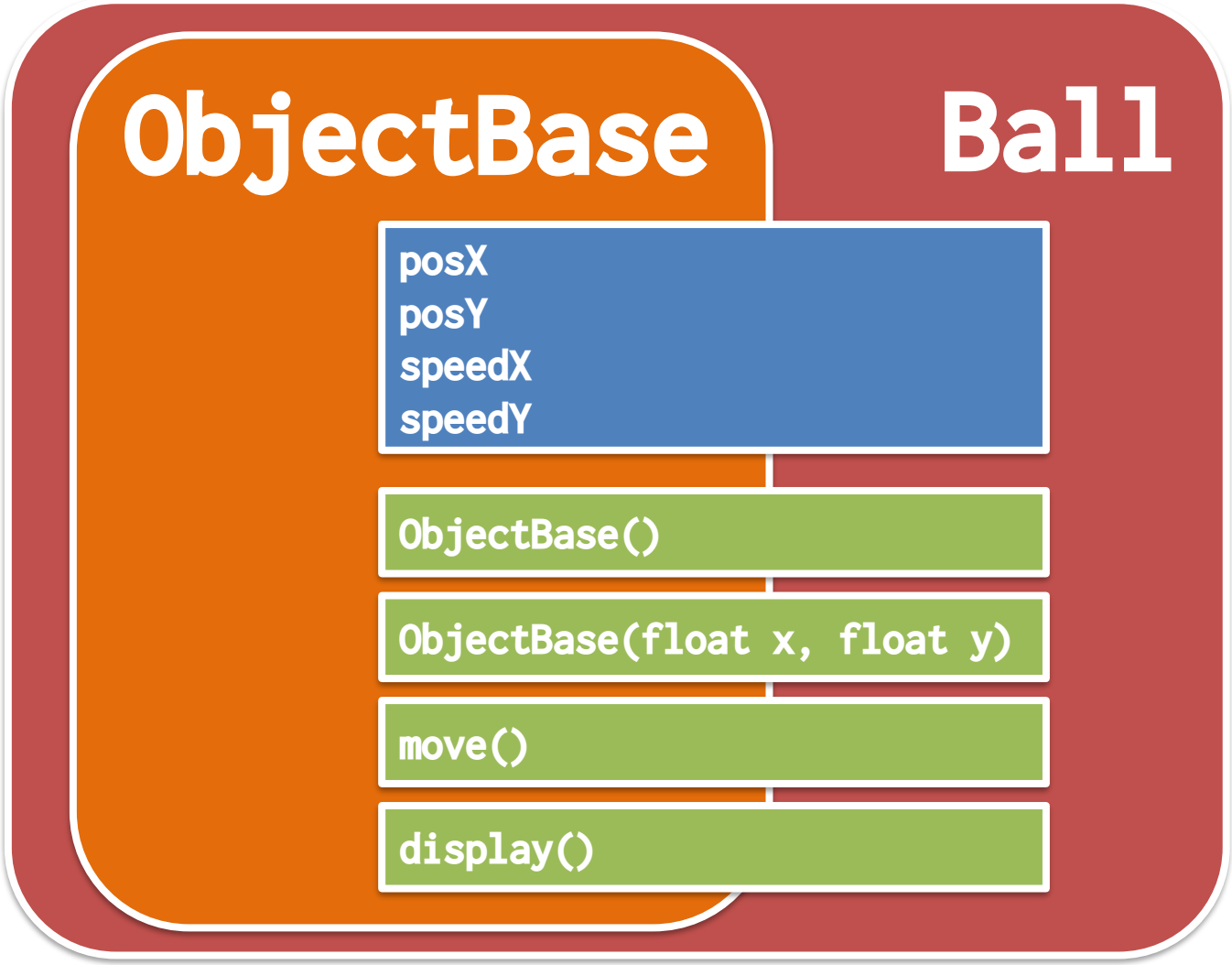
- Ballにインスタンス変数diameterを追加して，10-50の間でランダムに決定！
  - ObjectBaseのコンストラクタを呼び出してないのに動く！（コンストラクタはオーバーライドされない）

```
class Ball extends ObjectBase {
  int diameter;

  Ball(){
    diameter = (int)random(10, 50);
  }

  void display(){
    fill(255, 0, 0);
    ellipse(posX, posY, diameter, diameter);
  }
```

注意

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

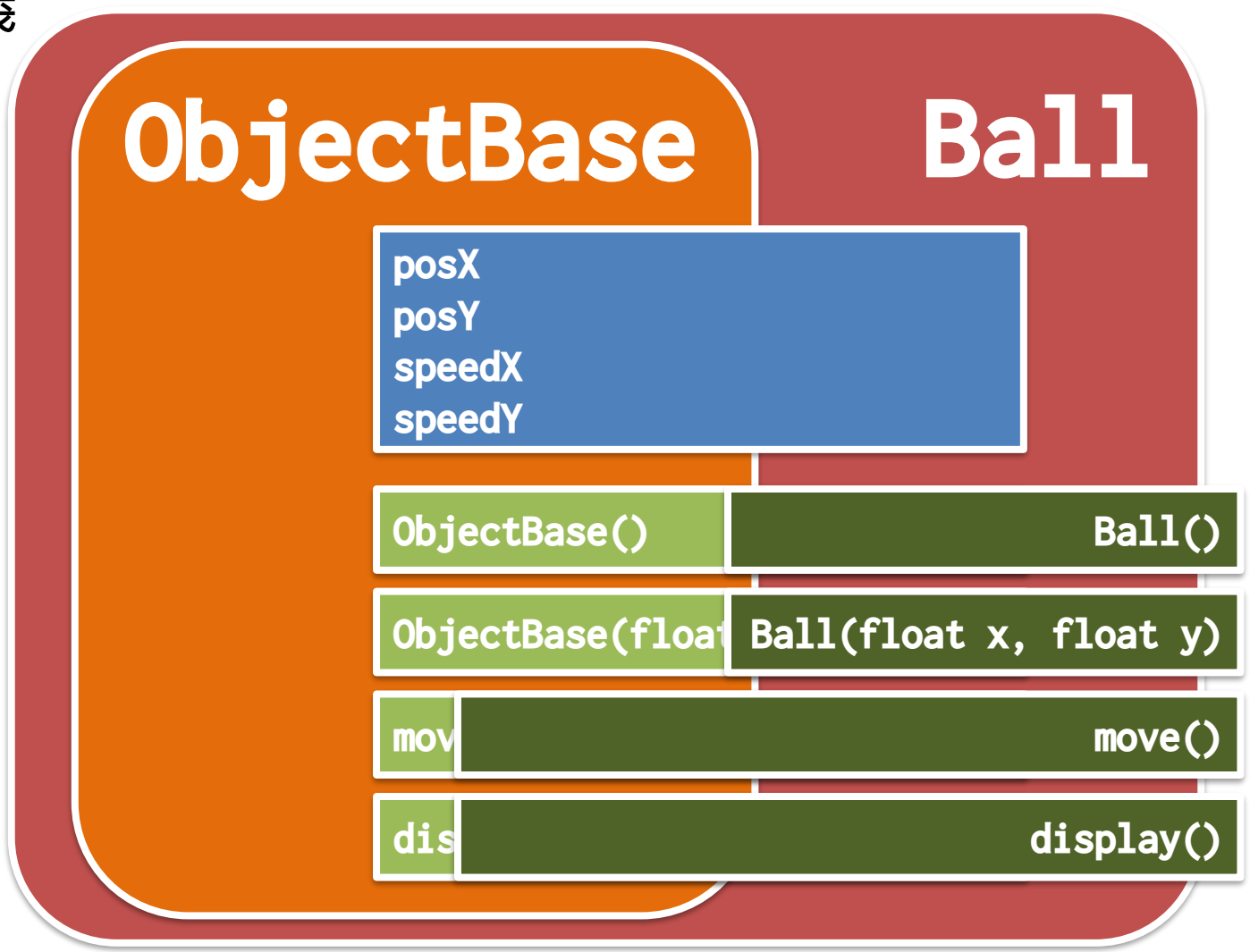
- コンストラクタに引数がある場合は要コンストラクタの定義



注意

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- コンストラクタに引数がある場合は要コンストラクタの定義



親コンストラクタの呼び出し

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 空っぽで同じ引数のコンストラクタを作る

```
class ObjectBase {
  float posX;
  float posY;
  float speedX;
  float speedY;
  ObjectBase(){
    posX = random(width);
    posY = random(height);
    speedX = random(-5, 5);
    speedY = random(-5, 5);
  }
  ObjectBase(float x, float y){
    posX = x;
    posY = y;
    speedX = random(-5, 5);
    speedY = random(-5, 5);
  }
```

```
class Ball extends ObjectBase {

  Ball(){
  }

  Ball(float x, float y){
    super(x, y);
  }
```

課題3-1:basic_CharaClass

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 配布するCharacterBaseクラスを継承し，CharacterBaseクラス内のcenterX，centerYを利用してその位置に自身のキャラクタを描画するようにせよ
  - 継承したクラスではdisplayActiveとdisplayInactiveをオーバーライドせよ
  - moveメソッドをオーバーライドして特殊な動き方をするように変更してもかまわないが，画面の外に出ていかないようにせよ
- なお配布プログラムではCharacterBaseクラスのキャラクタ？のみを描画しているが，自分で作成したクラス（継承したクラス）のキャラクタも一緒に表示するようにせよ
- クラスの名前は名前と番号が含まれるようにし、そのクラスのみ別ファイルとして作成せよ
  - （例）X-3-34 中村の場合、クラス名を「ExNakamura334」とする
- 来週配布して利用します

課題3-2: basic_boundAll3

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 講義中に扱ったObjectBaseクラスを継承し，赤色の○が動き回るBallクラスと，緑色の△が動き回るTriangleクラス，青色の□が動き回るSquareクラスを作成せよ
- ただし，○は上下左右の壁で跳ね返り，△は上下左右で逆から出てくるように，□は左右で跳ね返り，上下では逆から出てくるようにせよ
- また，この継承したクラスを利用して800x600のウインドウ内を100個の赤色○と，60個の緑色△と，80個の青色□を描画せよ

課題3-3: basic_clickRandom

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 800x600のウインドウ内をクリックするたびに，そのクリックした場所にランダムに赤色の○か，青色の△か，カラフルな□が生成され，○と△と□が動き回るプログラムを作成せよ
  - ただし，コンストラクタの中ではmouseX，mouseYという変数を使わないようにせよ
  - ○は上下左右の壁で跳ね返り，△は上下左右で逆から出てくるように，□は左右で跳ね返り，上下では逆から出てくるようにせよ（速度は-5～5の実数値）
  - 赤色の○は，生成されたときにランダムに5～60の間で直径が決まるようにせよ
  - カラフルな□は，生成されたときにランダムに色が決まるようにせよ

宿題3-1: hw_GenomeAnalyzer

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 以下の条件を満たすGenomeAnalyzerクラスを作成し，次のページで示すプログラムで動作させよ
  - コンストラクタの引数として塩基配列を文字列として与え，インスタンス変数でその塩基配列情報を保持するようにせよ
  - 返り値が実数値のcalcRatioA(), calcRatioT(), calcRatioG(), calcRatioC()というインスタンスメソッドを作成し，それぞれA,T,G,Cのそれぞれの出現率を返すようにせよ
    - わかる人は，引数をchar型にし，メソッドはcalcRatio()だけでもよい
  - 返り値が整数のlength()というインスンタンスメソッドを作成し，塩基配列の長さを返すようにせよ
  - 引数として文字列パターンを与え，返り値としてその文字列が塩基配列内で現れる回数を整数値で返すcountPatternメソッドを作成せよ．ただし，「AAAA」は「AAA」が2回とカウントするようにせよ
    - 引数プログラム内でそれぞれcountPattern("AAA"), countPattern("TTT"), countPattern("GGG"), countPattern("CCC")のそれぞれの出現回数を出力するようにせよ

宿題3-1: hw_GenomeAnalyzer

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室


```
GenomeAnalyzer analyzer;

void setup() {
  size(400, 300);
  // hw_Covid19Genome の塩基配列を利用せよ↓
  analyzer = new GenomeAnalyzer("すんごく長い塩基配列");

  println("塩基配列の長さは" + analyzer.length() + "文字");

  println("Aの出現率", analyzer.calcRatioA());
  println("Gの出現率", analyzer.calcRatioG());
  println("Tの出現率", analyzer.calcRatioT());
  println("Cの出現率", analyzer.calcRatioC());

  println("AAAの出現回数", analyzer.countPattern("AAA"));
  println("GGGの出現回数", analyzer.countPattern("GGG"));
  println("TTTの出現回数", analyzer.countPattern("TTT"));
  println("CCCの出現回数", analyzer.countPattern("CCC"));
}
```

宿題3-2: hw_JankenGuriko

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- グリコというゲーム（グーで勝つと+3，チョキで勝つと+5，パーで勝つと+6の点が入る）がある
  - チョキで+6が一般的だがここでは+5とする
- 毎回じゃんけんを繰り返し，10000点を先に獲得したほうが勝ちになる
- 配布する hw_JankenGuriko 内のJankenAgentクラスを継承し，自身のエージェントを作成せよ
- battleメソッドに自身のエージェントと，敵のエージェントを引数として与え，10000点になるまで戦わせて勝ちましょう！
  - 少なくとも JankenAgent自身，AgentNakamura334には勝ちましょう
  - できれば，知り合いのAgentももらって，勝ちましょう！！

宿題3-3: hw_CalcVector

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- 2つのベクトルをメソッドでセットし，様々な計算を可能とするVectorOperationクラスを作成し，次ページのプログラムで動作を確認せよ
  - 引数を実数値の配列としてベクトルを与え，インスタンス変数としてベクトルを配列として管理するようにする setVectorA(), setVectorB()という2つのメソッドを作成せよ
  - 返り値が実数のcalcVectorMagnitudeA(), calcVectorMagnitudeB()メソッドを作成し，それぞれのベクトルの大きさを求めて返すようにせよ
  - 返り値が実数のcalcInnerProduct()メソッドを作成し，内部にもつ2つのベクトルの内積を求めて返すようにせよ
  - 返り値が実数のcalcCosSimilarity()メソッドを作成し，内部に持つ2つのベクトルのコサイン類似度を計算して返すようにせよ．2つのベクトルの類似度は，次ページ以降に示すコサイン類似度を利用することで求めることができる
  - 返り値が実数のcalcArea()メソッドを作成し，内部の2つのベクトルから面積を求め，返すようにせよ．2つのベクトルからなる三角形の面積は，次ページ以降の式で求めることが可能である

宿題3-3: hw_CalcVector

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- setVectorAとsetVectorBで指定されるベクトルの次元（配列の要素数）は，同じであると決めつけて良い

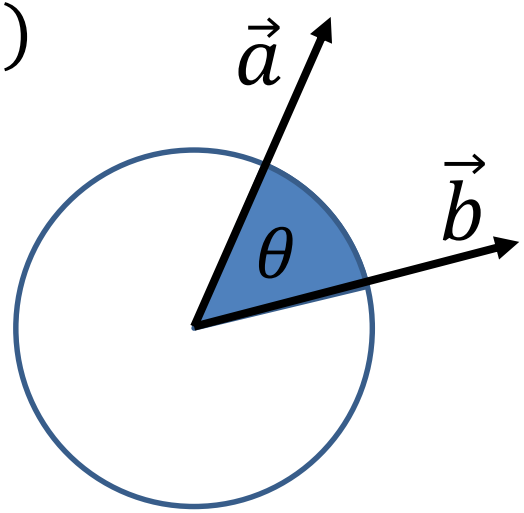
```
void setup() {
  VectorOperation vo = new VectorOperation();
  float[] vectorA = {10, 0, 1, 9};
  float[] vectorB = {8, 1, 1, 4};
  vo.setVectorA(vectorA);
  vo.setVectorB(vectorB);

  println("Aの大きさ", vo.calcVectorMagnitudeA());
  println("Bの大きさ", vo.calcVectorMagnitudeB());

  println("内積", vo.calcInnerProduct());
  println("コサイン類似度", vo.calcCosSimilarity());
  println("面積", vo.calcArea());
}
```

コサイン類似度

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- ベクトル $\vec{a}$ と $\vec{b}$ のなす角度を利用した類似度

$$\vec{a} = (a_1, a_2) \text{ と } \vec{b} = (b_1, b_2)$$



- ベクトルの内積の計算は...

$$\vec{a} \cdot \vec{b} = |\vec{a}||\vec{b}|cos\theta$$

$$cos\theta = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|} = \frac{a_1 \cdot b_1 + a_2 \cdot b_2}{\sqrt{a_1^2 + a_2^2} \cdot \sqrt{b_1^2 + b_2^2}}$$

コサイン類似度：3次元の場合

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- ベクトル $\vec{a}$ と $\vec{b}$ のなす角度を利用した類似度

$$\vec{a} = (a_1, a_2, a_3), \vec{b} = (b_1, b_2, b_3)$$

$$cos\theta = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|} = \frac{a_1 \cdot b_1 + a_2 \cdot b_2 + a_3 \cdot b_3}{\sqrt{a_1^2 + a_2^2 + a_3^2} \cdot \sqrt{b_1^2 + b_2^2 + b_3^2}}$$

三角形の面積

明治大学総合数理学部
先端メディアサイエンス学科
中村研究室

- ベクトル $\vec{a}$ と $\vec{b}$ のからなる三角形の面積Sは下記の式で求めることができる

$$S = \frac{1}{2}\sqrt{|\vec{a}|^2\left|\vec{b}\right|^2 - (\vec{a}\cdot\vec{b})^2}$$

  - 三次元の場合はこんな感じ

$$\vec{a} = (a_1, a_2, a_3), \vec{b} = (b_1, b_2, b_3)$$

$$S = \frac{1}{2}\sqrt{|\vec{a}|^2\left|\vec{b}\right|^2 - (\vec{a}\cdot\vec{b})^2}$$

$$= \frac{1}{2}\sqrt{(a_1^2 + a_2^2 + a_3^2)(b_1^2 + b_2^2 + b_3^2) - (a_1\cdot b_1 + a_2\cdot b_2 + a_3\cdot b_3)^2}$$