

pi-9. Javaプログラミングにおけるクラス継承

トピックス：スーパークラス, サブクラス,
extends, super, 継承

URL: <https://www.kkaneko.jp/pro/pi/index.html>

(Java の基本, スライド資料とプログラム例)

金子邦彦



全体まとめ



- サブクラスのオブジェクトは、すべてスーパークラスに属する
- 継承：スーパークラスの属性とメソッドをサブクラスが受け継ぐ

但し、コンストラクタは受け継がない

- コンストラクタ以外のメソッドは、オーバーライドできる
- サブクラスで、スーパークラスにない属性やメソッドを追加できる

番号	項目
	復習
9-1	スーパークラス, サブクラス
9-2	スーパークラス, サブクラスの Java プログラム, 継承
9-3	メソッドのオーバーライド
9-4	サブクラスでの属性の追加

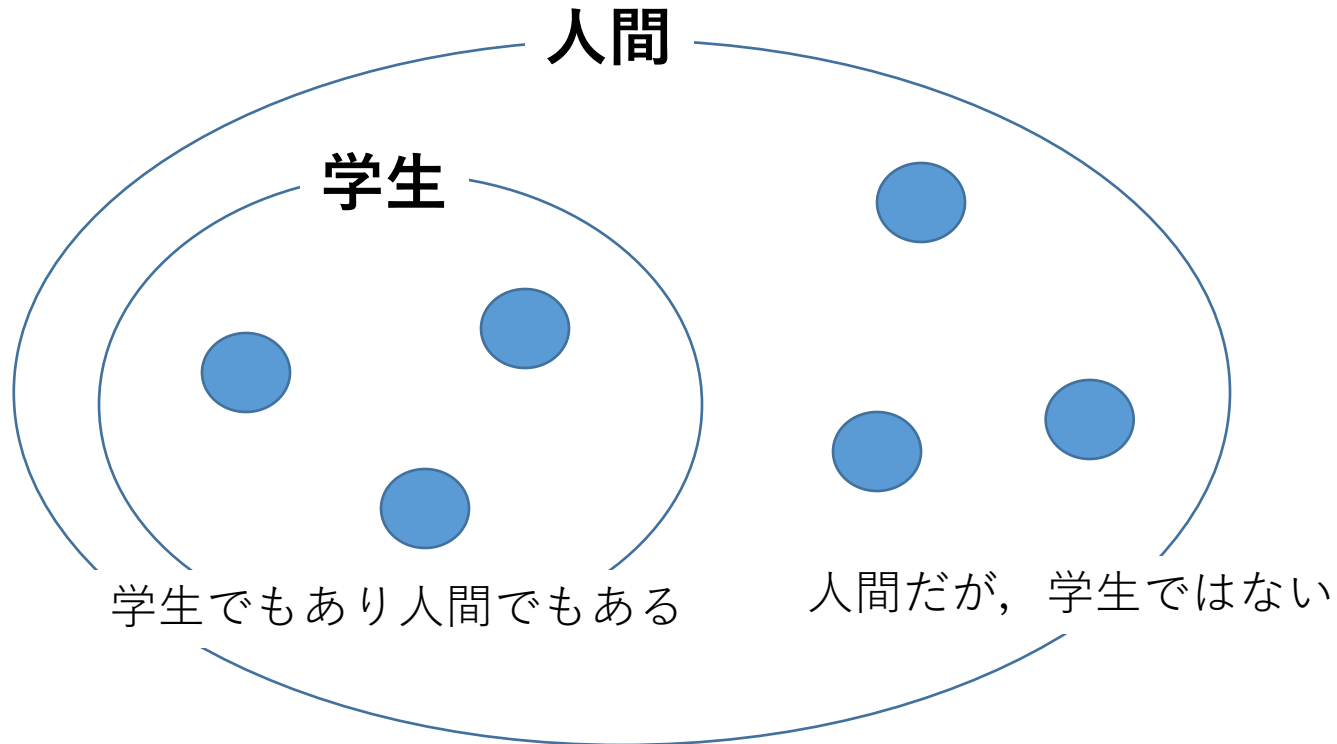
各自、資料を読み返したり、課題に取り組んだりも行う

この授業では、**Java** を用いて基礎を学び、マスターする

クラスとオブジェクト



クラスは、同じ種類のオブジェクトの集まりと考えることができる

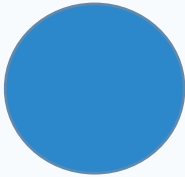


Java のクラス

クラス Ball



オブジェクト



半径 1, 場所 (8, 10)
色 blue

オブジェクト



半径 3, 場所 (2, 4)
色 green

```
1 class Ball {  
2     double x;  
3     double y;  
4     double r;  
5     String color;  
6     public Ball(double x, double y, double r, String color) {  
7         this.x = x;  
8         this.y = y;  
9         this.r = r;  
10        this.color = color;  
11    }  
12    public void printout() {  
13        System.out.println(this.x);  
14        System.out.println(this.y);  
15        System.out.println(this.r);  
16        System.out.println(this.color);  
17    }  
18 }
```

クラス定義のプログラム

```
Ball a = new Ball(8, 10, 1, "blue");  
Ball b = new Ball(2, 4, 3, "green");
```

オブジェクト生成のプログラム

```
a.printout();  
b.printout();
```

メソッドアクセスのプログラム

クラス定義のプログラム



```
1 class Ball { クラス名: Ball
2     double x;
3     double y;
4     double r;
5     String color;
6     public Ball(double x, double y, double r, String color) {
7         this.x = x;
8         this.y = y;
9         this.r = r;
10        this.color = color;
11    }
12    public void printout() {
13        System.out.println(this.x);
14        System.out.println(this.y);
15        System.out.println(this.r);
16        System.out.println(this.color);
17    }
18 }
```

4つの属性

メソッド: Ball

メソッド: printout

このクラス定義を使用した, **オブジェクト**の生成

a	8	10	1	"red"
b	2	4	3	"green"
	x	y	r	color

```
Ball a = new Ball(8, 10, 1, "blue");
Ball b = new Ball(2, 4, 3, "green");
```

メソッドアクセス



```
a.printout();  
b.printout();
```

Javaプログラム

a や b **オブジェクト**
printout() **メソッド**
間を「.」で区切っている

Java Tutor の起動



① ウェブブラウザを起動する

② **Java Tutor** を使いたいので, 次の URL を開く
<http://www.pythontutor.com/>

③ 「**Java**」をクリック ⇒ **編集画面が開く**

Learn Python, JavaScript, C, C++, and Java

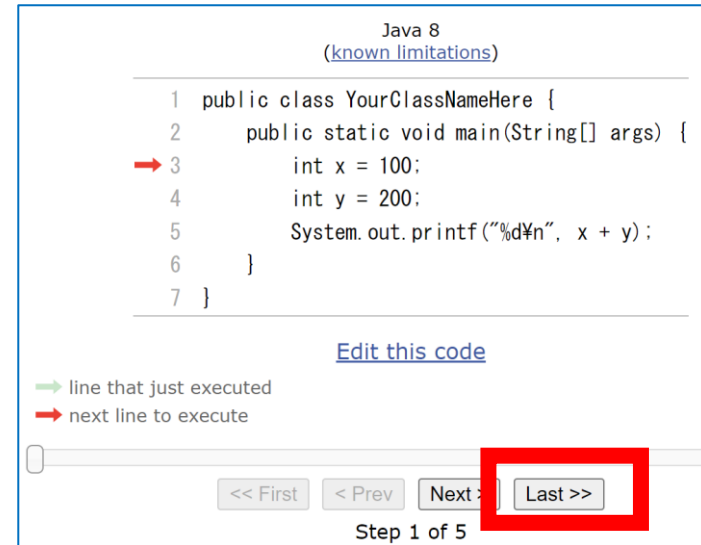
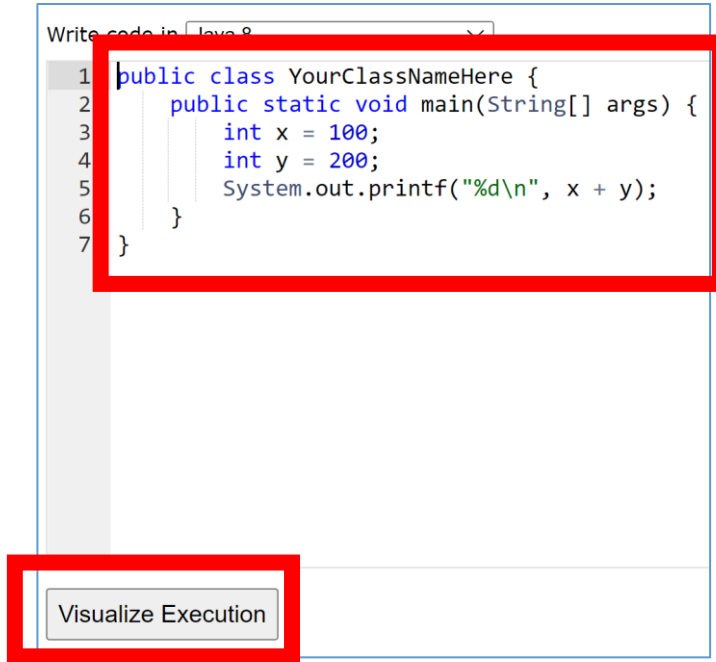
This tool helps you learn Python, JavaScript, C, C++, and Java programming by [visualizing code execution](#). You can use it to debug your homework assignments and as a supplement to online coding tutorials.

Start coding now in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

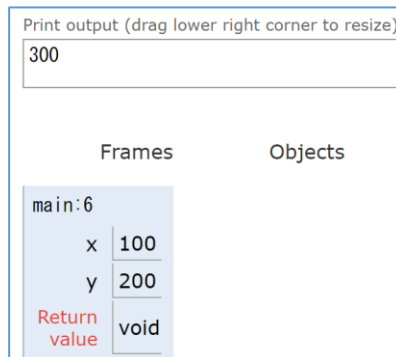
Over 15 million people in more than 180 countries have used Python Tutor to visualize over 200 million pieces of code. It is the most widely-used program visualization tool for computing education.

You can also embed these visualizations into any webpage. Here's an example showing recursion in Python:

Java Tutor でのプログラム実行手順



(1) 「**Visualize Execution**」をクリックして**実行画面**に切り替える



(2) 「**Last**」をクリック。



(3) **実行結果を確認**する。

(4) 「**Edit this code**」をクリックして**編集画面**に戻る

Java Tutor 使用上の注意点①



- ・実行画面で、次のような**赤の表示**が出ることがある →
無視してよい

過去の文法ミスに関する確認表示
邪魔なときは「**Close**」

Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

The screenshot displays the Python Tutor interface for Java 8. On the left, a code editor shows a Java class with a main method. Line 3, containing 'int x = 100;', is highlighted with a red arrow, indicating it is the next line to execute. Below the code editor, there are navigation buttons: '<< First', '< Prev', 'Next >', and 'Last >>'. A progress bar at the bottom indicates 'Step 1 of 3'. On the right side, there are tabs for 'Frames' and 'Objects'. The 'Frames' tab shows 'main:3'. Below this, an error message is displayed: 'You just fixed the following error:' followed by a code snippet with a red 'x' icon on line 3. The error message is 'Error: ';' expected'. Below the error message, there is a text input field for feedback and two buttons: 'Submit' and 'Close'. The 'Close' button is highlighted with a red box. At the bottom right of the error message box, there is a link: 'hide all of these pop-ups'.

```
Java 8  
(known limitations)  
1 public class YourClassNameHere {  
2     public static void main(String[] args) {  
→ 3         int x = 100;  
4     }  
5 }
```

[Edit this code](#)

→ line that just executed
→ next line to execute

<< First < Prev Next > Last >>

Step 1 of 3

[Customize visualization](#)

Frames Objects

main:3

You just fixed the following error:

```
1 public class YourClassNameHere {  
2     public static void main(String[] args) {  
3         int x = 100  
4     }  
5 }
```

Error: ';' expected

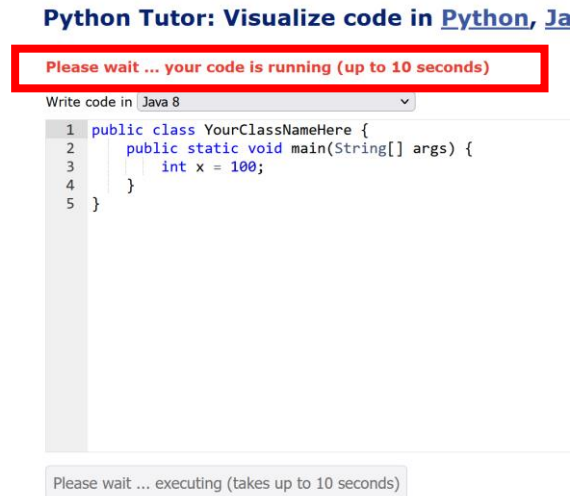
Please help us improve this tool with your feedback.
What misunderstanding do you think caused this error?

Submit Close [hide all of these pop-ups](#)

Java Tutor 使用上の注意点②



「please wait ... executing」のとき， 10秒ほど待つ。



→ 混雑しているときは， 「Server Busy・・・」
というメッセージが出ることがある。

混雑している． 少し（数秒から数十秒）待つと自動で表示が変わる（変わらない場合には，操作をもう一度行ってみる）

9-1. スーパークラス, サブクラス

スーパークラスとサブクラス



正方形は、長方形である
(正方形ではない長方形もある)

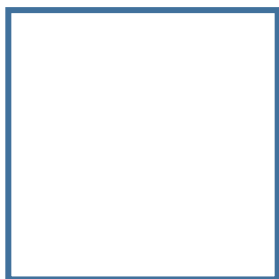


長方形 : スーパークラス
正方形 : サブクラス

長方形と正方形



幅 1, 高さ 1,
場所 (0, 3)



幅 2, 高さ 2,
場所 (1, 1)



幅 1, 高さ 2,
場所 (4, 8)



幅 2, 高さ 1,
場所 (8, 10)



幅 3, 高さ 1,
場所 (7, 5)

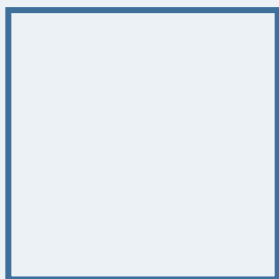
長方形と正方形



長方形であり、
正方形でもある



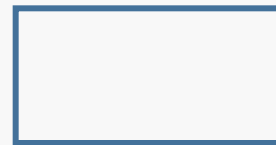
幅 1, 高さ 1,
場所 (0, 3)



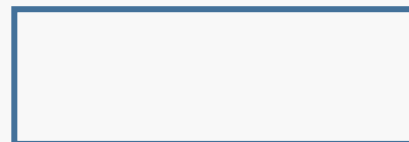
幅 2, 高さ 2,
場所 (1, 1)



幅 1, 高さ 2,
場所 (4, 8)



幅 2, 高さ 1,
場所 (8, 10)



幅 3, 高さ 1,
場所 (7, 5)

長方形であるが、
正方形ではない

スーパークラスとサブクラス

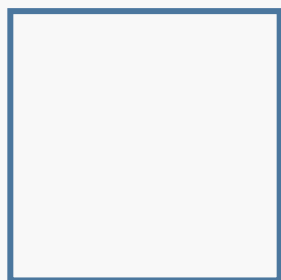


「長方形」の
クラス

「正方形」の
クラス



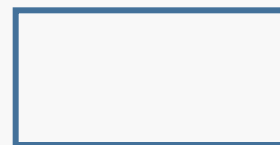
幅 1, 高さ 1,
場所 (0, 3)



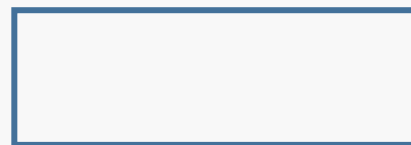
幅 2, 高さ 2,
場所 (1, 1)



幅 1, 高さ 2,
場所 (4, 8)



幅 2, 高さ 1,
場所 (8, 10)



幅 3, 高さ 1

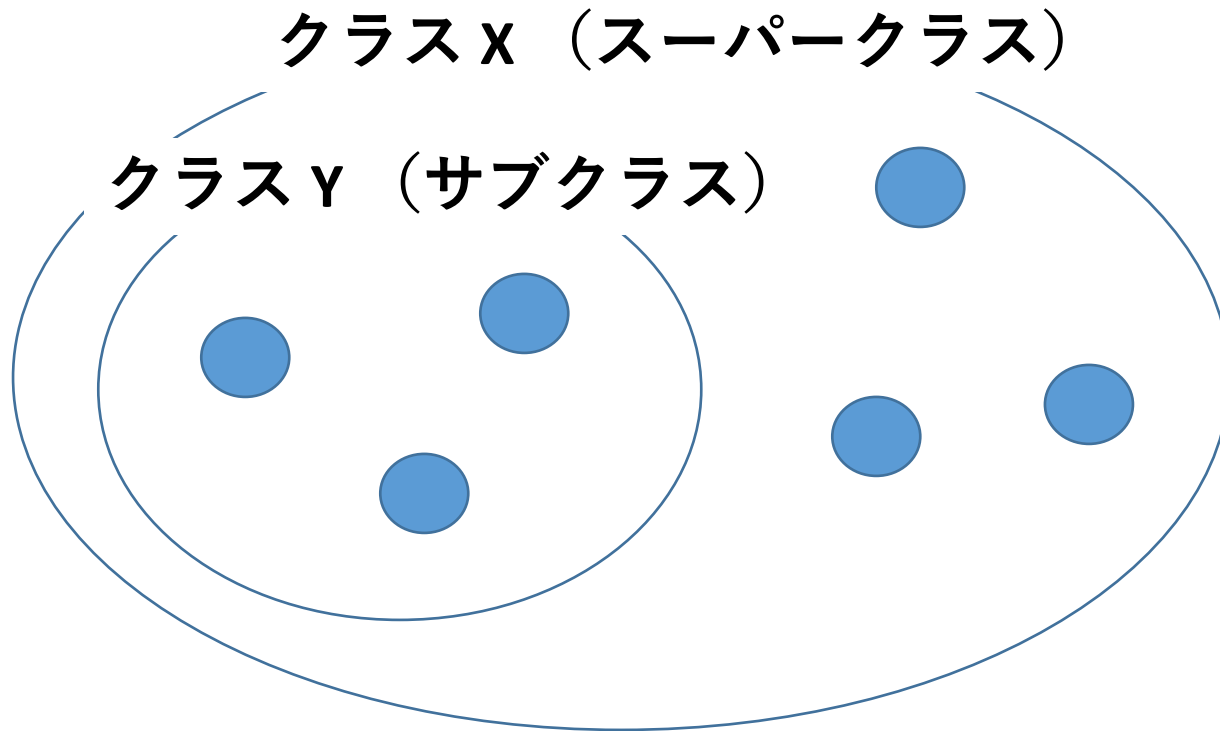
「正方形」ではない
長方形もある

「正方形」はすべて長方形
である

スーパークラス, サブクラス



サブクラスの**オブジェクト**は, すべて**スーパー**
クラスに属する



スーパークラス, サブクラスの例



例①

くだもの

スーパークラス

りんご

サブクラス

例②

住民

スーパークラス

世帯主

サブクラス

例③

図形

スーパークラス

円

サブクラス

9-2. スーパークラス, サブクラスの Java プログラム, 継承

Java でのクラス定義



- class Y **extends** X : クラス Y の**スーパークラス**はXであることを指定
- **super** : **スーパークラス**のコンストラクタの呼び出し

```
class X {  
    . . .  
    public X( . . . ) {  
        . . .  
    }  
    . . .  
}
```

クラス定義

```
class Y extends X {  
    . . .  
    public Y( . . . ) {  
        super( . . . )  
        . . .  
    }  
    . . .  
}
```

クラス定義

Y のスーパークラスが
X である場合



「長方形」のクラス定義の例

クラス名: Rectangle

```
1 class Rectangle {
2     double x;
3     double y;
4     double width;
5     double height;
6     public Rectangle(double x, double y, double width, double height) {
7         this.x = x;
8         this.y = y;
9         this.width = width;
10        this.height = height;
11    }
12    public void printout() {
13        System.out.println(this.x);
14        System.out.println(this.y);
15        System.out.println(this.width);
16        System.out.println(this.height);
17    }
18 }
```

4つの属性

メソッド: Rectangle

メソッド: printout

このクラス定義を使用した, **オブジェクト**の生成

a	4	8	1	2
b	8	10	2	1
	x	y	width	height

```
Rectangle a = new Rectangle(4, 8, 1, 2);
Rectangle b = new Rectangle(8, 10, 2, 1);
```

継承



- **継承**：スーパークラスの**属性**と**メソッド**を**サブクラス**が**受け継ぐ**
- 但し，コンストラクタは**受け継がない**
(コンストラクタは必ず定義する必要がある)

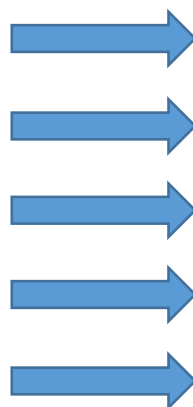
長方形 (Rectangle)

属性とメソッド

x
y
width
height
printout

スーパークラス

継承



正方形 (Square)

属性とメソッド

x
y
width
height
printout

サブクラス

「正方形」のクラス定義の例



クラス名: Square スーパークラス: Rectangle

メソッド: Square

```
19 class Square extends Rectangle {
20     public Square(double x, double y, double size) {
21         super(x, y, size, size);
22     }
23 }
```

このクラス定義を使用した, **オブジェクト**の生成

c	0	3	1	1
d	1	1	2	2
	x	y	width	height

```
Square c = new Square(0, 3, 1);
Square d = new Square(1, 1, 2);
```

Java プログラム

キーワード



extends
super

スーパークラスの指定
スーパークラスの**コンストラクタ**の
呼び出し

演習

資料 : 26 ~ 28

【トピックス】

- スーパークラス
- サブクラス
- extends
- super

① Java Tutor のエディタで次のプログラムを入れる



```
1  class Rectangle {
2      double x;
3      double y;
4      double width;
5      double height;
6      public Rectangle(double x, double y, double width, double height) {
7          this.x = x;
8          this.y = y;
9          this.width = width;
10         this.height = height;
11     }
12     public void printout() {
13         System.out.println(this.x);
14         System.out.println(this.y);
15         System.out.println(this.width);
16         System.out.println(this.height);
17     }
18 }
```

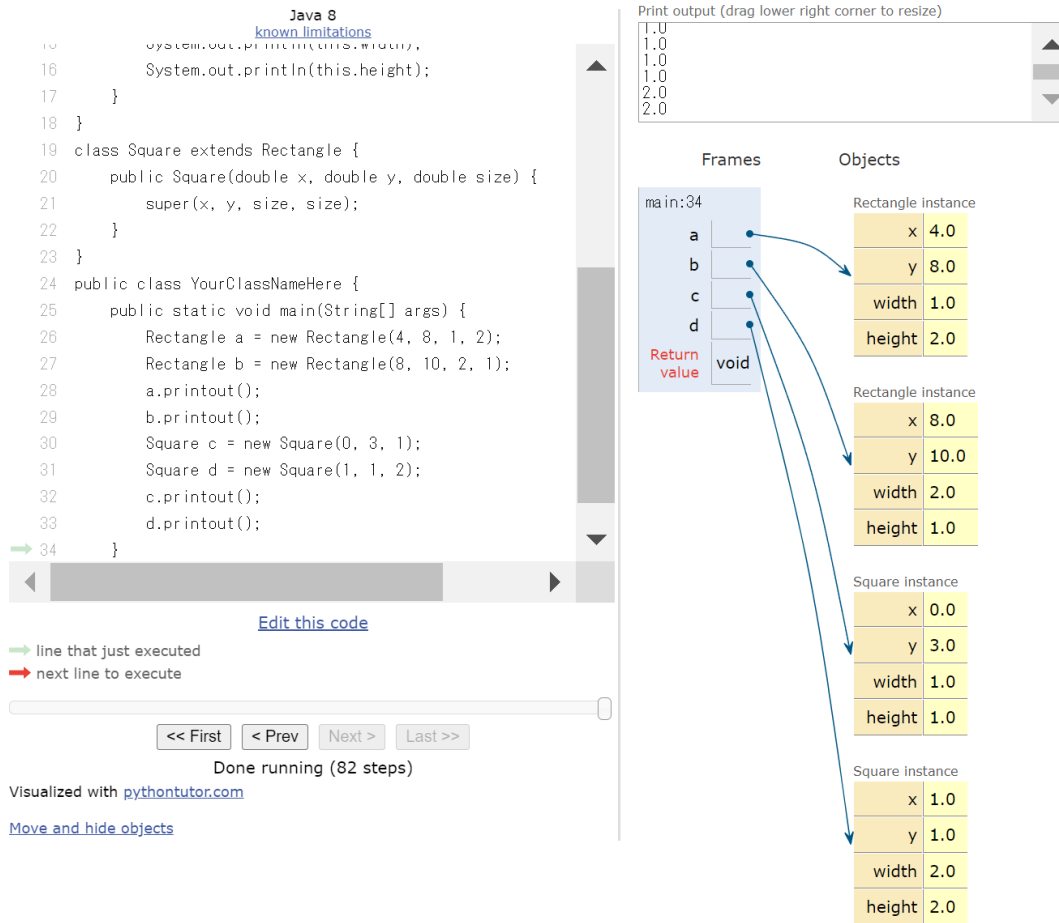
② 引き続き、次のプログラムを入れる



```
19 class Square extends Rectangle {
20     public Square(double x, double y, double size) {
21         super(x, y, size, size);
22     }
23 }
24 public class YourClassNameHere {
25     public static void main(String[] args) {
26         Rectangle a = new Rectangle(4, 8, 1, 2);
27         Rectangle b = new Rectangle(8, 10, 2, 1);
28         a.printout();
29         b.printout();
30         Square c = new Square(0, 3, 1);
31         Square d = new Square(1, 1, 2);
32         c.printout();
33         d.printout();
34     }
35 }
```

③ 実行し，結果を確認する

(あとで使うので、プログラムを消さないこと)



The screenshot shows the Python Tutor interface. On the left, the code editor displays a Java-like code snippet (though the title says 'Java 8', the code is Python). The code defines a `Square` class that extends `Rectangle` and creates four objects: `a`, `b`, `c`, and `d`. The execution is paused at line 34. On the right, the 'Print output' window shows the output of the `printout` method calls. Below that, the 'Frames' and 'Objects' panels are visible. The 'Frames' panel shows the `main` frame with variables `a`, `b`, `c`, and `d`. The 'Objects' panel shows three instances: a `Rectangle` instance (x: 4.0, y: 8.0, width: 1.0, height: 2.0), another `Rectangle` instance (x: 8.0, y: 10.0, width: 2.0, height: 1.0), and a `Square` instance (x: 0.0, y: 3.0, width: 1.0, height: 1.0). Arrows indicate the mapping from the variables in the frames to the objects in the objects panel.

4つのオブジェクト
a, b, c, d が生成される

「**Visual Execution**」をクリック．そして「**Last**」をクリック．結果を確認．
「**Edit this code**」をクリックすると，エディタの画面に戻る

9-3. メソッドのオーバーライド

メソッドのオーバーライド



- コンストラクタ以外のメソッドは, 受け継がずにオーバーライドできる

オーバーライドは, 別の定義を行うこと

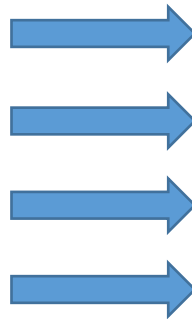
長方形 (Rectangle)

属性とメソッド

x
y
width
height
printout

スーパークラス

継承



正方形 (Square)

属性とメソッド

x
y
width
height
printout ← **オーバーライド**

サブクラス

「正方形」のクラス定義の例

クラス名: Square スーパークラス: Rectangle

```
19 class Square extends Rectangle {
20     public Square(double x, double y, double size) {
21         super(x, y, size, size);
22     }
23     public void printout() {
24         System.out.println(this.x);
25         System.out.println(this.y);
26         System.out.println(this.width);
27     }
28 }
```

メソッド: Square

メソッド: printout

- メソッド printout のオーバーライド
- スーパークラスのメソッド printout とは別の定義を行う。

演習

資料 : 33 ~ 34

【トピックス】

- ・メソッドのオーバーライド

① Java Tutor のエディタで次のプログラムを **加える**

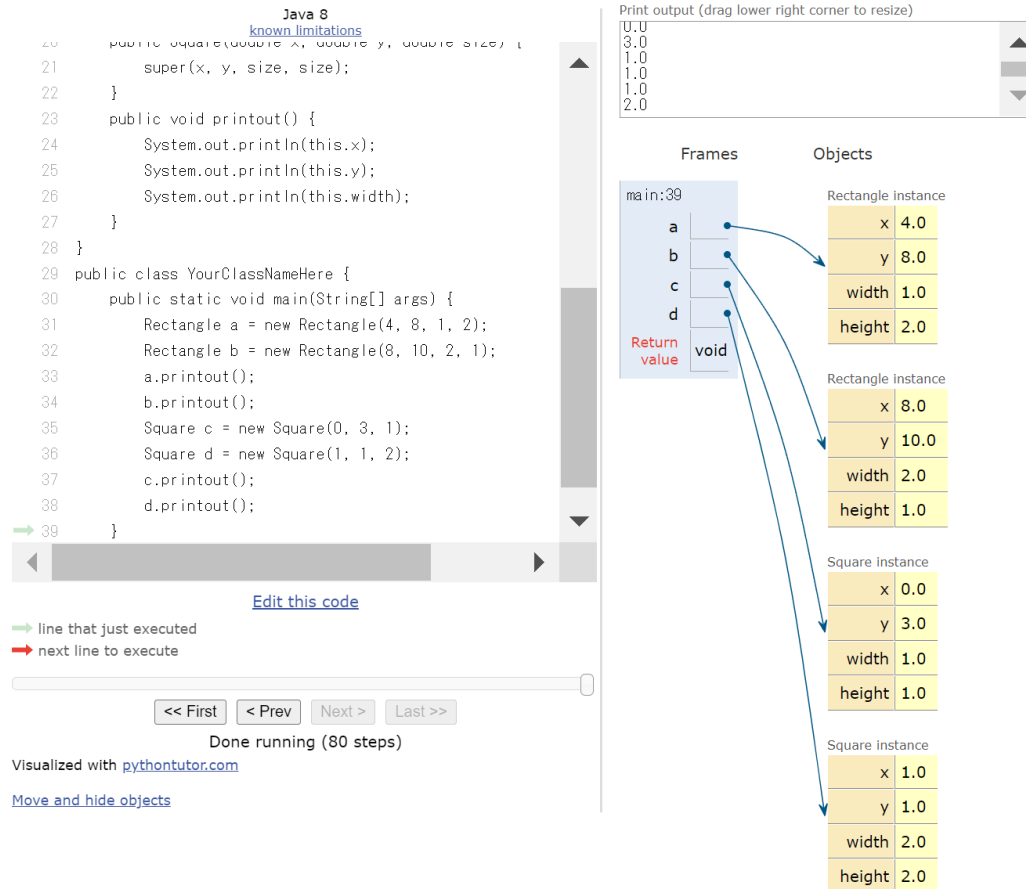


```
1 class Rectangle {
2     double x;
3     double y;
4     double width;
5     double height;
6     public Rectangle(double x, double y, double width, double height) {
7         this.x = x;
8         this.y = y;
9         this.width = width;
10        this.height = height;
11    }
12    public void printout() {
13        System.out.println(this.x);
14        System.out.println(this.y);
15        System.out.println(this.width);
16        System.out.println(this.height);
17    }
18 }
19 class Square extends Rectangle {
20     public Square(double x, double y, double size) {
21         super(x, y, size, size);
22     }
23     public void printout() {
24         System.out.println(this.x);
25         System.out.println(this.y);
26         System.out.println(this.width);
27     }
28 }
29 public class YourClassNameHere {
30     public static void main(String[] args) {
31         Rectangle a = new Rectangle(4, 8, 1, 2);
32         Rectangle b = new Rectangle(8, 10, 2, 1);
33         a.printout();
34         b.printout();
35         Square c = new Square(0, 3, 1);
36         Square d = new Square(1, 1, 2);
37         c.printout();
38         d.printout();
39     }
40 }
```

```
public void printout() {
    System.out.println(this.x);
    System.out.println(this.y);
    System.out.println(this.width);
}
```

② 実行し，結果を確認する

(あとで使うので、プログラムを消さないこと)



Print output (drag lower right corner to resize)

0.0
3.0
1.0
1.0
1.0
2.0

Frames

main:39
a
b
c
d
Return value
void

Objects

Rectangle instance

x	4.0
y	8.0
width	1.0
height	2.0

Rectangle instance

x	8.0
y	10.0
width	2.0
height	1.0

Square instance

x	0.0
y	3.0
width	1.0
height	1.0

Square instance

x	1.0
y	1.0
width	2.0
height	2.0

Visualized with pythontutor.com

[Move and hide objects](#)

表示が変わる

4つのオブジェクト
a, b, c, d が生成される

「**Visual Execution**」をクリック。そして「**Last**」をクリック。結果を確認。
「**Edit this code**」をクリックすると，エディタの画面に戻る

9-4. サブクラスでの属性の追加

サブクラスでの属性、メソッドの追加

- サブクラスで、スーパークラスにない属性やメソッドを追加できる

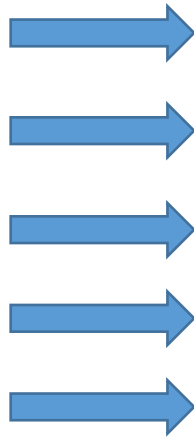
長方形 (Rectangle)

属性とメソッド

x
y
width
height
printout

スーパークラス

継承



色付きの長方形 (ColorRectangle)

属性 とメソッド

x
y
width
height
printout

color **サブクラスで追加された属性**

サブクラス

「色付きの長方形」のクラス定義の例

クラス名: ColorRectangle スーパークラス: Rectangle

メソッド: Square

```
29 class ColorRectangle extends Rectangle {
30     String color;
31     public ColorRectangle(double x, double y, double width, double height, String color) {
32         super(x, y, width, height);
33         this.color = color;
34     }
35     public void printout() {
36         System.out.println(this.x);
37         System.out.println(this.y);
38         System.out.println(this.width);
39         System.out.println(this.height);
40         System.out.println(this.color);
41     }
42 }
```

メソッド: printout

- サブクラスで, 属性 color を追加

演習

資料 : 39 ~ 41

【トピックス】

- ・ サブクラスでの属性の追加

① Java Tutor のエディタで次のプログラムを **加える**



```
1 class Rectangle {
2     double x;
3     double y;
4     double width;
5     double height;
6     public Rectangle(double x, double y, double width, double height) {
7         this.x = x;
8         this.y = y;
9         this.width = width;
10        this.height = height;
11    }
12    public void printout() {
13        System.out.println(this.x);
14        System.out.println(this.y);
15        System.out.println(this.width);
16        System.out.println(this.height);
17    }
18 }
19 class Square extends Rectangle {
20     public Square(double x, double y, double size) {
21         super(x, y, size, size);
22     }
23     public void printout() {
24         System.out.println(this.x);
25         System.out.println(this.y);
26         System.out.println(this.width);
27     }
28 }
29 class ColorRectangle extends Rectangle {
30     String color;
31     public ColorRectangle(double x, double y, double width, double height, String color) {
32         super(x, y, width, height);
33         this.color = color;
34     }
35     public void printout() {
36         System.out.println(this.x);
37         System.out.println(this.y);
38         System.out.println(this.width);
39         System.out.println(this.height);
40         System.out.println(this.color);
41     }
42 }
43 public class YourClassNameHere {
44     public static void main(String[] args) {
45         Rectangle a = new Rectangle(4, 8, 1, 2);
46         Rectangle b = new Rectangle(8, 10, 2, 1);
47         a.printout();
48         b.printout();
49         Square c = new Square(0, 3, 1);
50         Square d = new Square(1, 1, 2);
51         c.printout();
52     }
53 }
```



```
29 class ColorRectangle extends Rectangle {
30     String color;
31     public ColorRectangle(double x, double y, double width, double height, String color) {
32         super(x, y, width, height);
33         this.color = color;
34     }
35     public void printout() {
36         System.out.println(this.x);
37         System.out.println(this.y);
38         System.out.println(this.width);
39         System.out.println(this.height);
40         System.out.println(this.color);
41     }
42 }
```

② 引き続き、次のプログラムを加える

```
1 class Rectangle {
2     double x;
3     double y;
4     double width;
5     double height;
6     public Rectangle(double x, double y, double width, double height) {
7         this.x = x;
8         this.y = y;
9         this.width = width;
10        this.height = height;
11    }
12    public void printout() {
13        System.out.println(this.x);
14        System.out.println(this.y);
15        System.out.println(this.width);
16        System.out.println(this.height);
17    }
18 }
19 class Square extends Rectangle {
20     public Square(double x, double y, double size) {
21         super(x, y, size, size);
22     }
23     public void printout() {
24         System.out.println(this.x);
25         System.out.println(this.y);
26         System.out.println(this.width);
27     }
28 }
29 class ColorRectangle extends Rectangle {
30     String color;
31     public ColorRectangle(double x, double y, double width, double height, double height, String color) {
32         super(x, y, width, height);
33         this.color = color;
34     }
35     public void printout() {
36         System.out.println(this.x);
37         System.out.println(this.y);
38         System.out.println(this.width);
39         System.out.println(this.height);
40         System.out.println(this.color);
41     }
42 }
43 public class YourClassNameHere {
44     public static void main(String[] args) {
45         Rectangle a = new Rectangle(4, 8, 1, 2);
46         Rectangle b = new Rectangle(0, 10, 2, 1);
47         a.printout();
48         b.printout();
49         Square c = new Square(0, 3, 1);
50         Square d = new Square(1, 1, 2);
51         c.printout();
52         d.printout();
53         ColorRectangle e = new ColorRectangle(0, 0, 1, 4, "Red");
54         e.printout();
55     }
56 }
```



```
ColorRectangle e = new ColorRectangle(0, 0, 1, 4, "Red");
e.printout();
```


③ 実行し，結果を確認する

Java 8
known limitations

```

38     System.out.println(this.width);
39     System.out.println(this.height);
40     System.out.println(this.color);
41 }
42 }
43 public class YourClassNameHere {
44     public static void main(String[] args) {
45         Rectangle a = new Rectangle(4, 8, 1, 2);
46         Rectangle b = new Rectangle(8, 10, 2, 1);
47         a.printout();
48         b.printout();
49         Square c = new Square(0, 3, 1);
50         Square d = new Square(1, 1, 2);
51         c.printout();
52         d.printout();
53         ColorRectangle e = new ColorRectangle(0, 0, 1, 4
54         e.printout();
55     }
56 }

```

Print output (drag lower right corner to resize)

```

2.0
0.0
0.0
1.0
4.0
Red

```

Frames

main:55

a

b

c

d

e

Return value

void

Objects

Rectangle Instance

x	4.0
y	8.0
width	1.0
height	2.0

Rectangle instance

x	8.0
y	10.0
width	2.0
height	1.0

Square Instance

x	0.0
y	3.0
width	1.0
height	1.0

Square Instance

x	1.0
y	1.0
width	2.0
height	2.0

ColorRectangle Instance

color	"Red"
x	0.0
y	0.0
width	1.0
height	4.0

Visualized with pythontutor.com

[Move and hide objects](#)

5つのオブジェクト
a, b, c, d, e が生成される

「**Visual Execution**」をクリック。そして「**Last**」をクリック。結果を確認。
「**Edit this code**」をクリックすると，エディタの画面に戻る

- サブクラスのオブジェクトは、すべてスーパークラスに属する
- 継承：スーパークラスの属性とメソッドをサブクラスが受け継ぐ

但し、コンストラクタは受け継がない

- コンストラクタ以外のメソッドは、オーバーライドできる
- サブクラスで、スーパークラスにない属性やメソッドを追加できる

関連ページ

- **Java プログラミング入門**

GDB online を使用

<https://www.kkaneko.jp/pro/ji/index.html>

- **Java の基本**

Java Tutor, GDB online を使用

<https://www.kkaneko.jp/pro/pi/index.html>

- **Java プログラム例**

<https://www.kkaneko.jp/pro/java/index.html>

ソースコード (9-2)



```
class Rectangle {
    double x;
    double y;
    double width;
    double height;
    public Rectangle(double x, double y, double width, double height) {
        this.x = x;
        this.y = y;
        this.width = width;
        this.height = height;
    }
    public void printout() {
        System.out.println(this.x);
        System.out.println(this.y);
        System.out.println(this.width);
        System.out.println(this.height);
    }
}
class Square extends Rectangle {
    public Square(double x, double y, double size) {
        super(x, y, size, size);
    }
}
public class YourClassNameHere {
    public static void main(String[] args) {
        Rectangle a = new Rectangle(4, 8, 1, 2);
        Rectangle b = new Rectangle(8, 10, 2, 1);
        a.printout();
        b.printout();
        Square c = new Square(0, 3, 1);
        Square d = new Square(1, 1, 2);
        c.printout();
        d.printout();
    }
}
```

ソースコード (9-3)



```
class Rectangle {
    double x;
    double y;
    double width;
    double height;
    public Rectangle(double x, double y, double width, double height) {
        this.x = x;
        this.y = y;
        this.width = width;
        this.height = height;
    }
    public void printout() {
        System.out.println(this.x);
        System.out.println(this.y);
        System.out.println(this.width);
        System.out.println(this.height);
    }
}

class Square extends Rectangle {
    public Square(double x, double y, double size) {
        super(x, y, size, size);
    }
    public void printout() {
        System.out.println(this.x);
        System.out.println(this.y);
        System.out.println(this.width);
    }
}

public class YourClassNameHere {
    public static void main(String[] args) {
        Rectangle a = new Rectangle(4, 8, 1, 2);
        Rectangle b = new Rectangle(8, 10, 2, 1);
        a.printout();
        b.printout();
        Square c = new Square(0, 3, 1);
        Square d = new Square(1, 1, 2);
        c.printout();
        d.printout();
    }
}
```

ソースコード (9-4)



```
class Rectangle {
    double x;
    double y;
    double width;
    double height;
    public Rectangle(double x, double y, double width, double height) {
        this.x = x;
        this.y = y;
        this.width = width;
        this.height = height;
    }
    public void printout() {
        System.out.println(this.x);
        System.out.println(this.y);
        System.out.println(this.width);
        System.out.println(this.height);
    }
}
class Square extends Rectangle {
    public Square(double x, double y, double size) {
        super(x, y, size, size);
    }
    public void printout() {
        System.out.println(this.x);
        System.out.println(this.y);
        System.out.println(this.width);
    }
}
class ColorRectangle extends Rectangle {
    String color;
    public ColorRectangle(double x, double y, double width, double height, String color) {
        super(x, y, width, height);
        this.color = color;
    }
    public void printout() {
        System.out.println(this.x);
        System.out.println(this.y);
        System.out.println(this.width);
        System.out.println(this.height);
        System.out.println(this.color);
    }
}
public class YourClassNameHere {
    public static void main(String[] args) {
        Rectangle a = new Rectangle(4, 8, 1, 2);
        Rectangle b = new Rectangle(8, 10, 2, 1);
        a.printout();
        b.printout();
        Square c = new Square(0, 3, 1);
        Square d = new Square(1, 1, 2);
        c.printout();
        d.printout();
        ColorRectangle e = new ColorRectangle(0, 0, 1, 4, "Red");
        e.printout();
    }
}
```