プログラミング入門 2

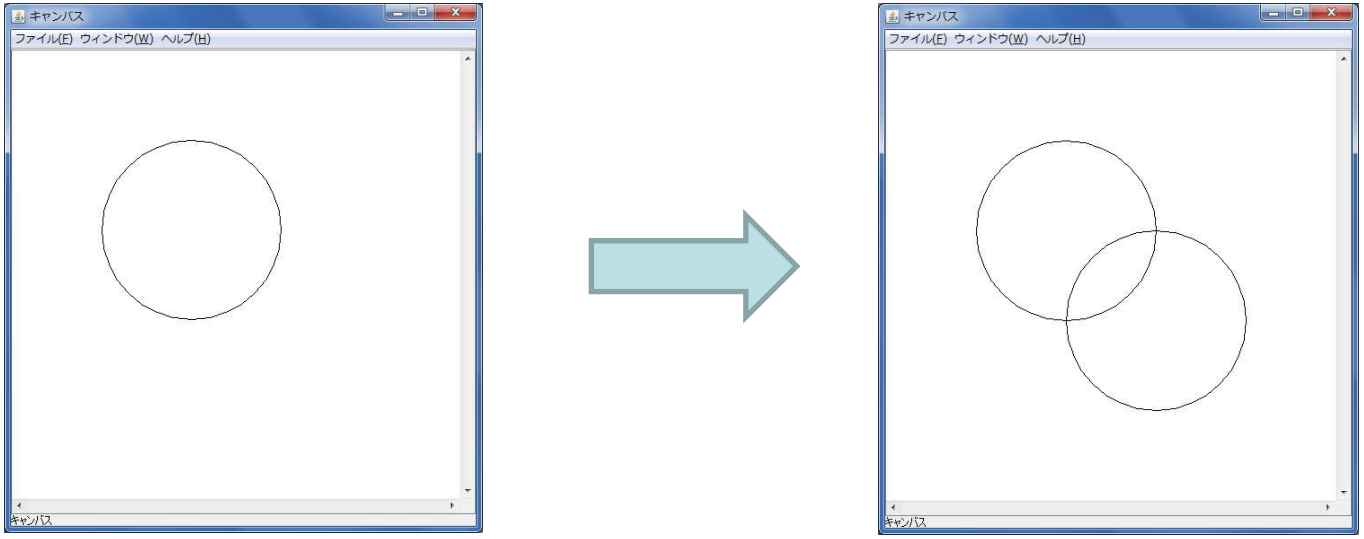
第4回　クラスとインスタンス(2)
クラスとインスタンスの詳細

テーマ：クラスとインスタンス（2）

- インスタンス生成とその利用（詳細）
- インスタンス（実体）と参照
- メソッドの参照呼び
- インスタンスを生成するメソッド
- インスタンスを要素に持つ配列

本日の主な題材

- 2つの MyCircle 変数にインスタンスを代入して、インスタンスフィールドを変更してみよ (例題11, 例題12)



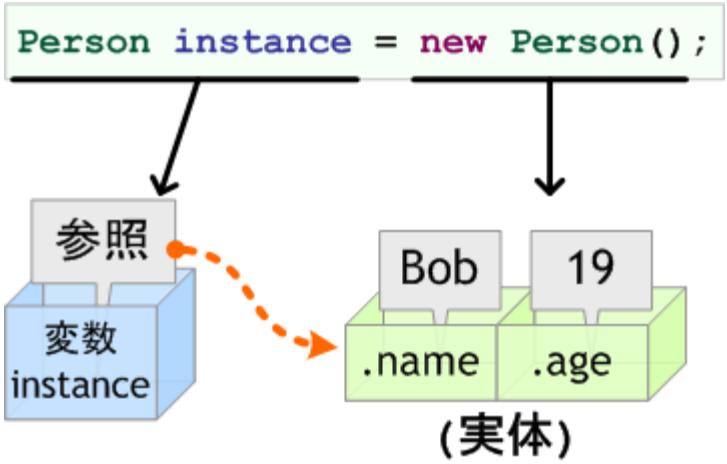
テーマ：クラスとインスタンス（2）

インスタンス生成とその利用（詳細）

- インスタンス（実体）と参照
- メソッドの参照呼び
- インスタンスを生成するメソッド
- インスタンスを要素に持つ配列

インスタンスの実体と参照

- 「new」という命令でインスタンスを生成すると、コンピューター上のどこかに実体を生成



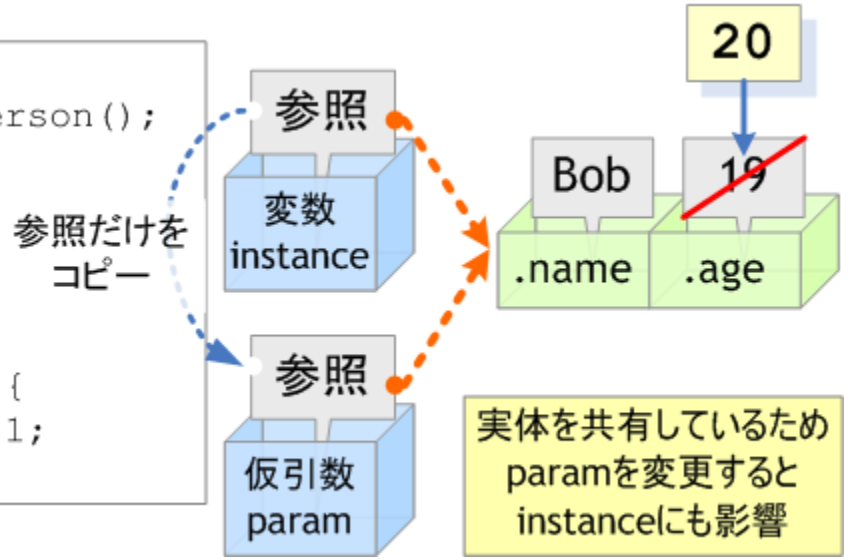
変数が記憶するのは
インスタンスの実体への参照だけ

配列と同様の仕組み

副作用 (インスタンス)

- modifyメソッドを起動する際に、インスタンスの参照だけをコピーして仮引数に渡す
  - 起動先で実体を変更すると、起動元にも影響

```
void start() {
  Person instance = new Person();
  ...
  modify(instance);
  ...
}

void modify(Person param) {
  param.age = param.age + 1;
}
```

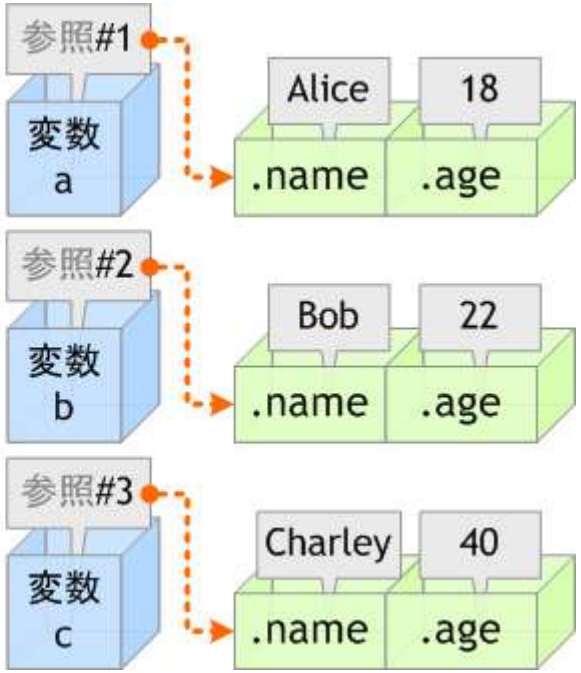


整数や実数と、配列や
インスタンスの違い

「new」による実体の生成

- 「new」でコンピューター上に実体を1つ生成
  - 複数の実体が必要なら複数回の「new」を処理

```
Person a = new Person();
a.name = "Alice";
a.age = 18;

Person b = new Person();
b.name = "Bob";
b.age = 22;

Person c = new Person();
c.name = "Charley";
c.age = 40;
```



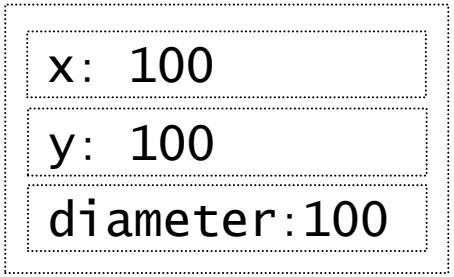
どの参照がどの実体を指すかを
正しく理解することが大切

インスタンスとその利用（詳細）

次のクラスを考える。

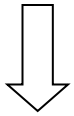
```
public clsas MyCircle {
  int x=100;
  int y=100;
  int diameter=100;
}
```

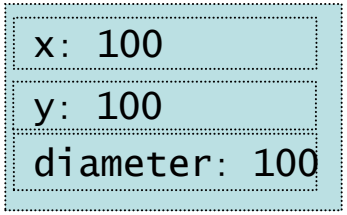
クラスMyCircle



このクラスのインスタンスを作成するためには、
次のような命令を用いた。

```
MyCircle c0 = new MyCircle();
```





これは、次の2行の作業をまとめて記述している。

```
MyCircle c0;
c0 = new MyCircle();
```

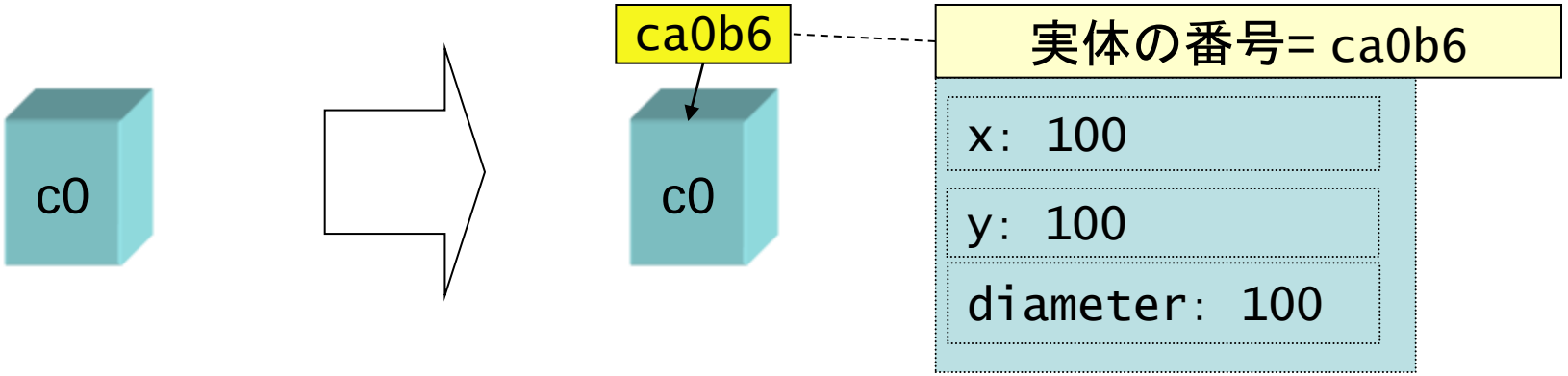
1行目：MyCircle型の変数c0を宣言。
2行目：MyCircleクラスのインスタンスを生成しc0に代入。

実体と参照

```
MyCircle c0;
c0 = new MyCircle();
```

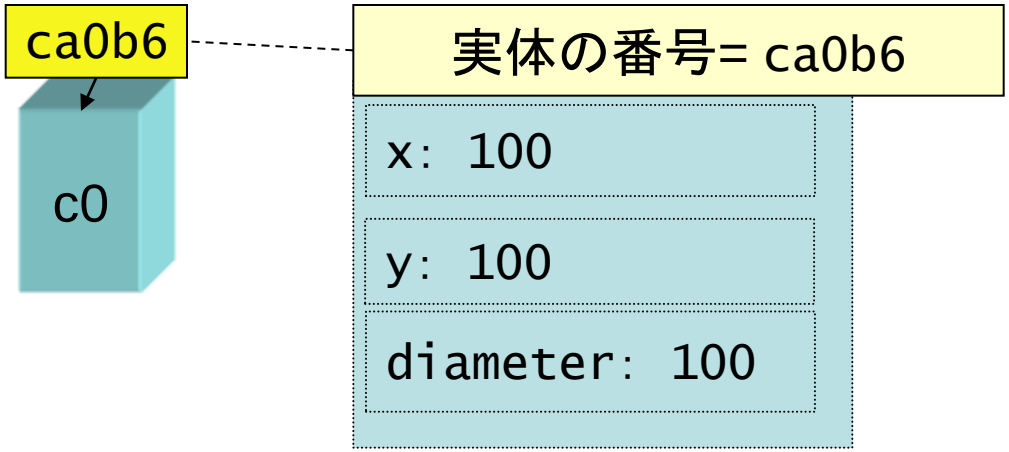
1行目：MyCircle型の変数c0を宣言。
2行目：MyCircleクラスのインスタンスを生成しc0に代入。

上記は、より正確には、
1行目：MyCircle型の変数c0を宣言。
2行目：(1) MyCircleクラスのインスタンス（実体）を生成し
　　(2) その「参照」をc0に保持させる。



実体と参照

「参照」は、プログラミング入門1の第10回で説明したものと同じである。先の例のc0は、インスタンスそのもの（実体）を保持するのではなく、インスタンスへの参照（＝インスタンスの住所、id番号）を保持する。



例：下記の場合、作成されたインスタンスはid番号「ca0b6」を持つ。
変数c0には、このid番号が格納されている。

```
MyCircle c0 = new MyCircle();
System.out.println(c0);
```

⇨ `j2.lesson04.MyCircle@ca0b6`

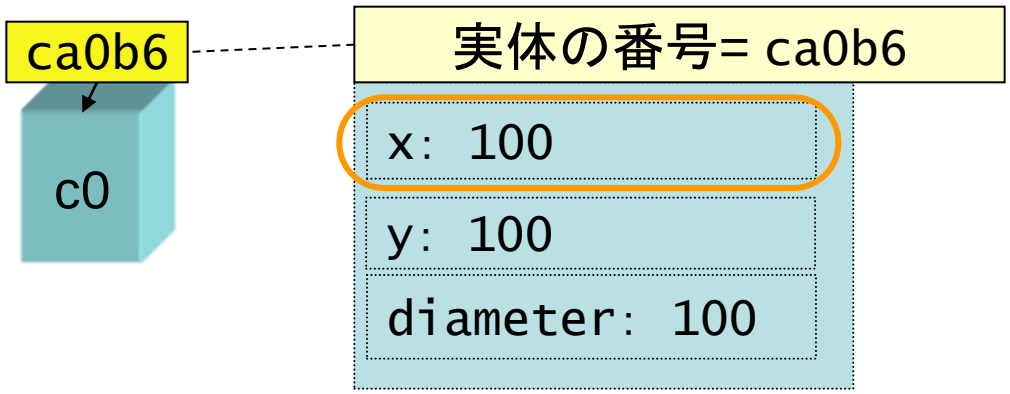
このように「参照」を保持する変数は、参照型（Reference type）変数と呼ばれる。なお、int型変数などは基本型（Primitive type）の変数と呼ばれる。

インスタンスへのアクセス

インスタンスへのアクセス（＝書き込みと読み込み）は、実際には参照を用いて間接的に行う。

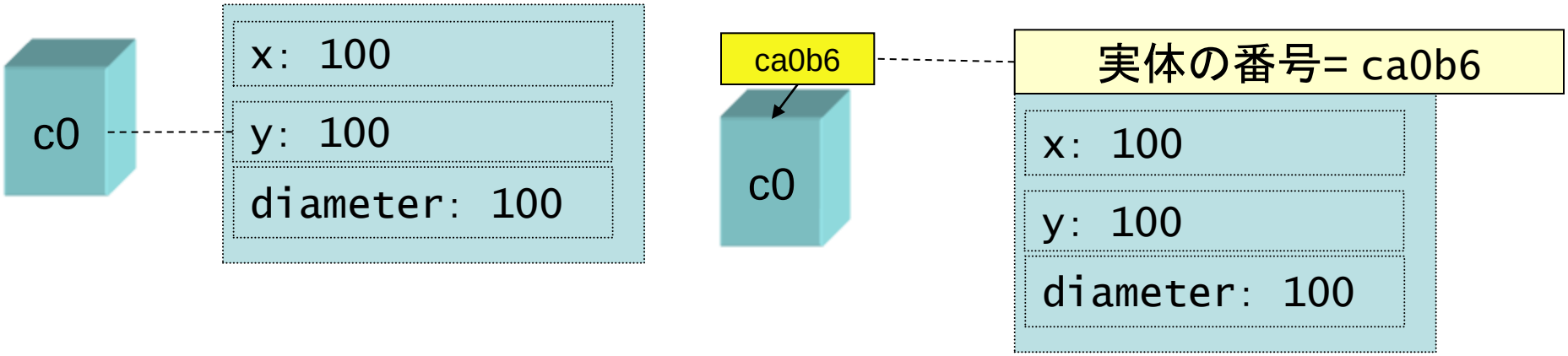
```
MyCircle c0;
c0 = new MyCircle();
System.out.println(c0.x)
```

3行目は、「インスタンスc0のインスタンス変数x」の内容を表示する」命令
であるが、より正確な表現では、
　「『変数c0が保持する「参照」が示す実体（すなわちインスタンス）』
　　のインスタンス変数x」の内容を表示する命令
となる。



まとめ

参照型変数への代入は、変数をインスタンスと「結びつける」と理解しておくのがよい。（左図）。リンクの役割を果たすのが「参照」である。

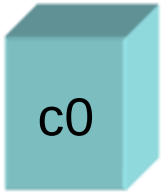



```
MyCircle c0 = new MyCircle();
```

なお「参照」か「実体」の違いを意識していなくても、不都合は起きない場合も多い。ただし、本日のテーマで扱う、参照型変数を用いた代入やメソッド呼び出しでは、基本型の時とは微妙な違いが生じることがあるので（特にインスタンスへの書き込みの際に）、意識する必要がある。

補足：定数null (ナル、ヌル)

```
MyCircle c0 = null;
System.out.println(c0);
System.out.println(c0.x);
```



参照型の変数は、nullという定数値を持つことができる。
基本的には、変数がどのインスタンスも結びついていないことを表す。
この状態で、値nullを保持する変数で、インスタンスにアクセスしようとすると、エラーがおこる。

```
Exception in thread "main"
java.lang.NullPointerException
at j2.lesson04.Sample1.main(Sample1.java:…)
```

テーマ：クラスとインスタンス(2)

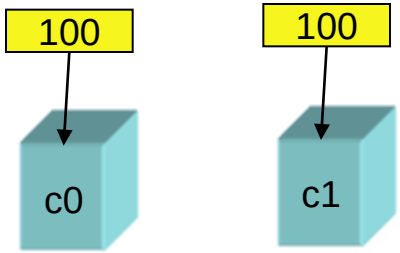
- インスタンス生成とその利用（詳細）
- インスタンス（実体）と参照
- メソッドの参照呼び
- インスタンスを生成するメソッド
- インスタンスを要素に持つ配列

インスタンス(実体)と参照

前項で述べたように、クラスから生成したインスタンスを利用する方法は、実際には、インスタンスへの「参照」を用いるので、数値型などの基本型のデータの扱いとは少し異なる場合がある。
(なお、配列データも、参照を通して間接的に利用するため、下記で説明する考え方は、配列の「実体」と「参照」の関係と同じである。プログラミング入門1第10回の説明も参考にせよ。)

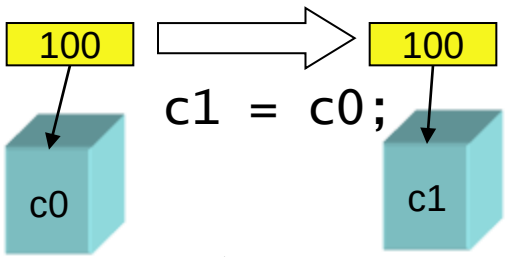
---

(1)
```
int c0 = 100;
int c1 = 100;
```



基本型変数への代入の例

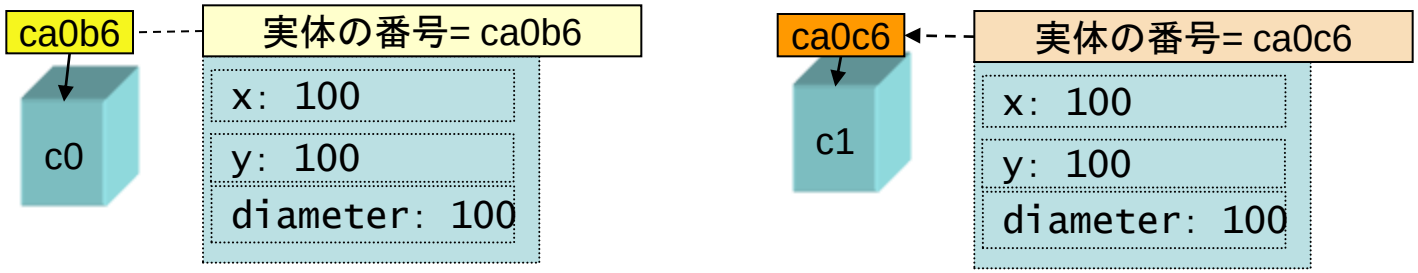
(2)
```
int c0 = 100;
int c1 = c0;
```



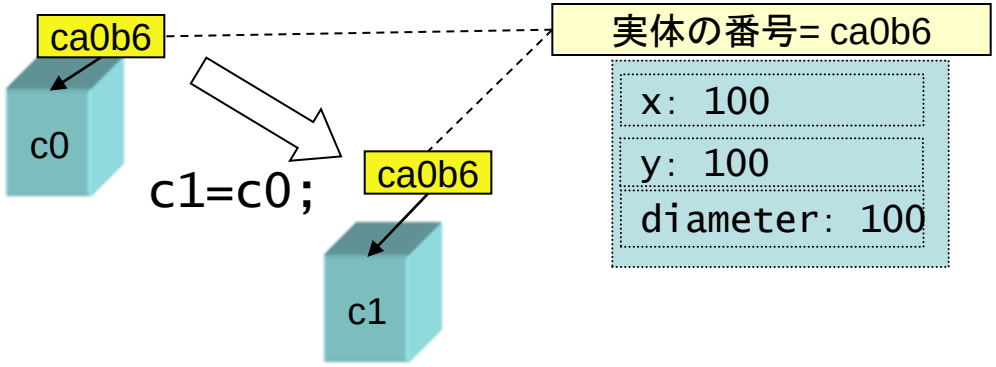
c0が保持する100というデータがコピーされ、c1のデータとなる。
結果的に(1)(2)は同じ。

参照型変数への
代入の例

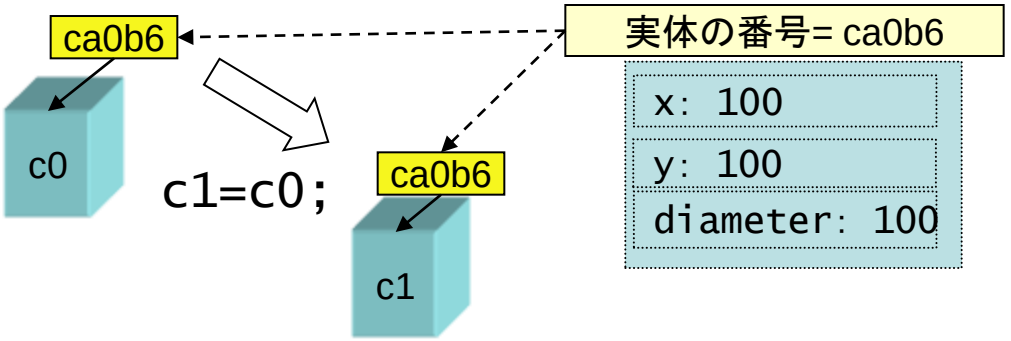
```
(3)
MyCircle c0 = new MyCircle();
MyCircle c1 = new MyCircle();
```




```
(4)
MyCircle c0 = new MyCircle();
MyCircle c1 = c0;
```



```
(4)
MyCircle c0 = new MyCircle();
MyCircle c1 = c0;
```



c0が保持する参照データがコピーされ、c1のデータとなる。

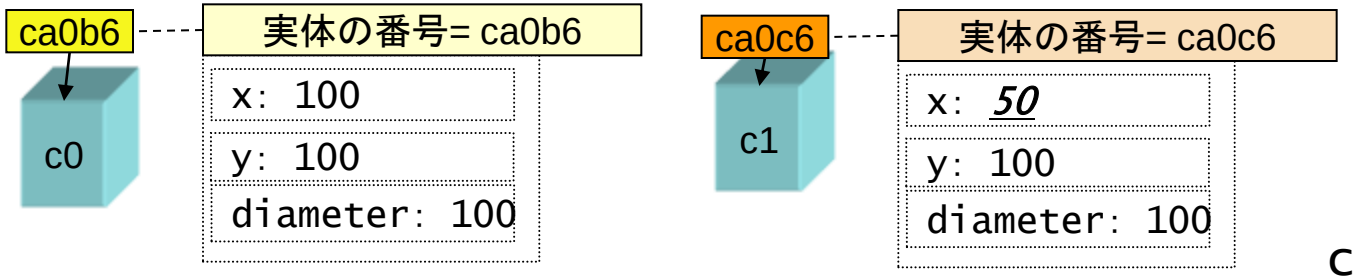
コピーされるのは「参照」だけであり、インスタンスそのものはコピーされていない。すなわち

- 前ページ(3)では、c0とc1がそれぞれ別個のインスタンスを示す異なる値の「参照」を持つ。
- (4)では、c0とc1とも同じ値の「参照」を持つ(＝同一のインスタンスを示す)。

参照のコピーが作られた場合、インスタンス変数へ値の書き込みを行う場合に注意が必要。
(3)(4)の両者で、c1.x = 50;
としたときに何が起こるか考えて見よ。(次ページスライド)

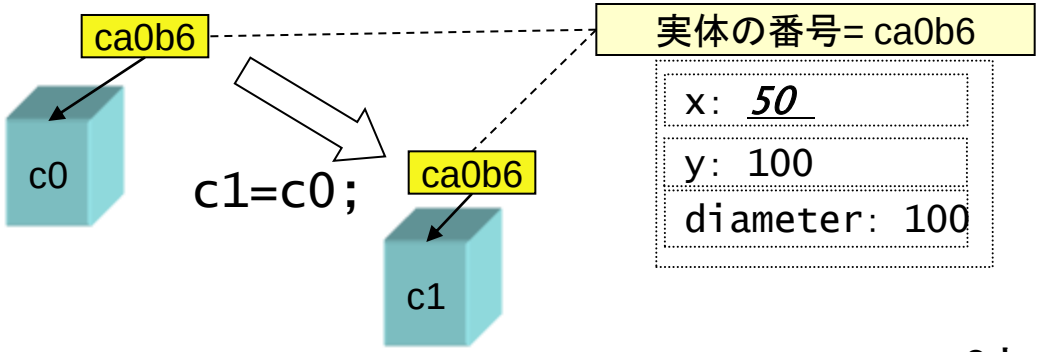
インスタンス変数の
内容を書き換える

```
(3)
MyCircle c0 = new MyCircle();
MyCircle c1 = new MyCircle();
c1.x = 50;
```



c1.x は50
c0.x は100のまま

```
(4)
MyCircle c0 = new MyCircle();
MyCircle c1 = c0;
c1.x = 50;
```



c1.x は 50
c0.x も 50に変わる

c0とc1とも同じ値の「参照」を持つ
（＝同一のインスタンスを示す）。

テーマ：クラスとインスタンス(2)

- インスタンス生成とその利用（詳細）
- インスタンス（実体）と参照
- メソッドの参照呼び
- インスタンスを生成するメソッド
- インスタンスを要素に持つ配列

メソッドの参照呼び（メソッド引数の参照渡し）

プログラムの実行中のメソッド呼び出しにおける、引数の受け渡しは、次のように行われる。(プログラミング入門1第4回)

(1) メソッドに渡したい値を計算し、
(2) これをメソッドの引数変数に代入して渡して
(3) 呼び出し先で利用する。

ここで、呼び出そうとするメソッドの引数の型が参照型（つまり、クラス名）である場合は、メソッドに渡される値は、インスタンスへの「参照」である。

インスタンスそのものが渡されないことに注意（次ページスライド）

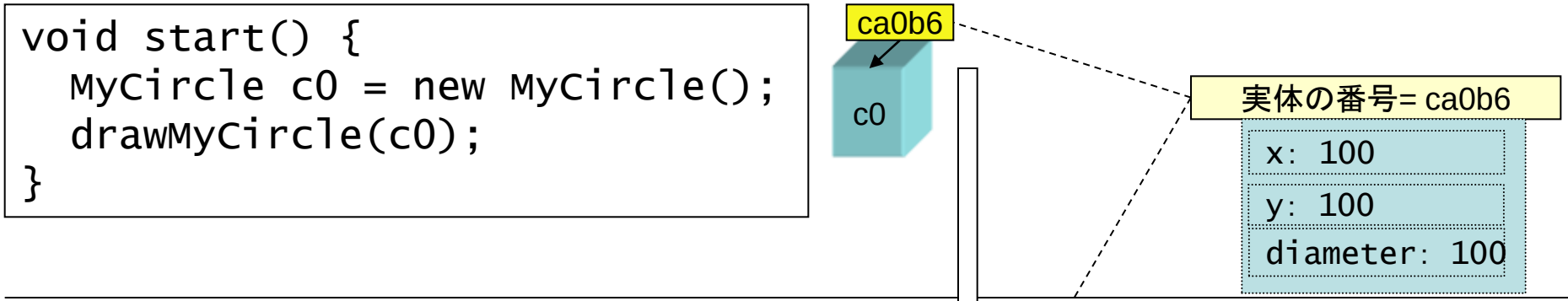
```
void start() {
  MyCircle c0 = new MyCircle();
  drawMyCircle(c0);
}
```

ca0b6

```
void
drawMyCircle(MyCircle c){
   Canvas.drawCircle(c.x, c.y,
                     c.diameter)
}
```

参照呼び(引数の参照渡し)

```
void start() {
  MyCircle c0 = new MyCircle();
  drawMyCircle(c0);
}
```




```
void
drawMyCircle(MyCircle c){
   Canvas.drawCircle(c.x, c.y,
                     c.diameter)
}
```

呼び出されたメソッド側には、コピーされた参照が渡され、それを通して、
インスタンスにアクセスすることが可能となる。

テーマ：クラスとインスタンス(2)

- インスタンス生成とその利用（詳細）
- インスタンス（実体）と参照
- メソッドの参照呼び
- インスタンスを生成するメソッド
- インスタンスを要素に持つ配列

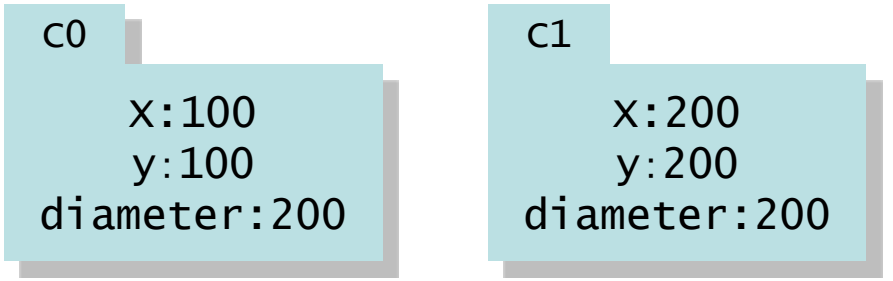
インスタンスを生成するメソッド(1)

複合データを作る方法は、大雑把には次の過程を経る。
(1) インスタンスを生成する。
(2) インスタンス変数に持たせたい値(初期値)を代入する。

ここでは、この(1)(2)を一つのメソッドとしてまとめる方法を紹介する。以下の例では、 MyCircle クラスのインスタンスを生成し、初期値を設定するメソッド createMyCircleを作成する。

```
MyCircle c0 = new MyCircle();
c0.x = 100;
c0.y = 100;
c0.diameter = 200;

MyCircle c1 = new MyCircle();
c1.x = 200;
c1.y = 200;
c1.diameter = 200;
```

```
MyCircle c0 = createMyCircle(100, 100, 200);
MyCircle c1 = createMyCircle(200, 200, 200);
```



■インスタンスを生成するメソッド
例題11 より

問題：

(a) MyCircleクラスのインスタンスを2つ作成せよ。ただし、同一の座標,直径を持つようにせよ。

(b) 次にこれらの円をキャンバス上に表示させよ。

返り値の型	メソッド名(引数)	機能
MyCircle	createMyCircle(int x, int y, int diameter)	MyCircleクラスのインスタンス変数の値がそれぞれx,y,diameterであるMyCircleインスタンスを作成し、返す。
void	drawMyCircle(MyCircle c)	キャンバスに引数のMyCircleインスタンスcを描画する。

（クラス名： Ex11TwoCircles）

例題21

```java
package j2.lesson04;

import gpjava.Canvas;

public class Ex11TwoCircles {
    public static void main(String[] args) {
        new Ex11TwoCircles().start();
    }

    void start() {
        MyCircle c0 = createMyCircle(100, 100, 200);
        MyCircle c1 = createMyCircle(100, 100, 200);

        Canvas.show();
        drawMyCircle(c0);
        drawMyCircle(c1);
    }

    MyCircle createMyCircle(int x, int y, int diameter) {
        MyCircle newObj = new MyCircle();
        newObj.x = x;
        newObj.y = y;
        newObj.diameter = diameter;
        return newObj;
    }

    void drawMyCircle(MyCircle c) {
        Canvas.drawOval(c.x, c.y, c.diameter, c.diameter);
    }
}
```

```java
package j2.lesson04;

public class MyCircle {
    int x;
    int y;
    int diameter;
}
```

インスタンスを生成するメソッド(2)

メソッドの定義の方法
（0） 複合データに含まれる一つ一つのデータを、引数で受け取るように、引数の型を決める。返り値の型は、作成したいインスタンスのクラス名である。（前回も少し触れたが、クラス名は、インスタンスの型名の役割も果たす。）

```
MyCircle createMyCircle(int x, int y, int diameter) {
    MyCircle newObj = new MyCircle();
    newObj.x = x;
    newObj.y = y;
    newObj.diameter = diameter;
    return newObj;
}
```

メソッド内では、
（1） インスタンスを生成する。
（2） 引数で渡されたデータをインスタンス変数に代入する。
（3） インスタンスを呼び出し元に返す。

インスタンスを生成するメソッド(3)

呼び出し方法

　複合データに含まれる一つ一つのデータを、メソッドの引数に渡してやればよい。それを束ねた複合データが返ってくる。（引数の順番は注意が必要である）

```
void start() {
    MyCircle c0 = createMyCircle(100, 100, 200);
    MyCircle c1 = createMyCircle(100, 100, 200);

    Canvas.show();
    drawMyCircle(c0);
    drawMyCircle(c1);
}

MyCircle createMyCircle(int x, int y, int diameter) {
    MyCircle newObj = new MyCircle();
    newObj.x = x;
    newObj.y = y;
    newObj.diameter = diameter;
    return newObj;
}
```

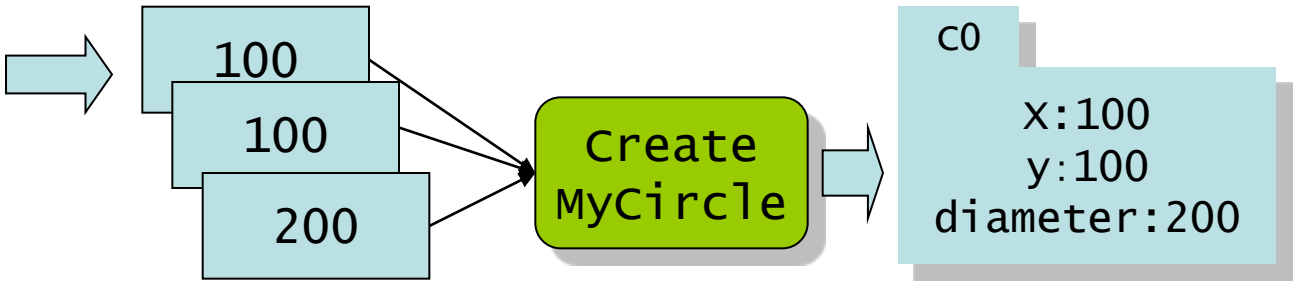
インスタンスを生成するメソッド（まとめ）

このような形でインスタンスを生成させるメソッドは、それを呼び出すことで、インスタンス（製品）を生産することができるので、ファクトリー（工場）メソッドとも呼ばれる。

データとデータを結びつけて、複合データを作成するというプロセスを一つの式（メソッド呼び出し式）で表現できるため、プログラムが分かりやすくなる。

```
MyCircle c0 = createMyCircle(100, 100, 200);
```

```
MyCircle createMyCircle(int x, int y, int diameter) {
    MyCircle newObj = new MyCircle();
    newObj.x = x;
    newObj.y = y;
    newObj.diameter = diameter;
    return newObj;
}
```



テーマ：クラスとインスタンス(2)

- インスタンス生成とその利用（詳細）
- インスタンス（実体）と参照
- メソッドの参照呼び
- インスタンスを生成するメソッド
- インスタンスを要素に持つ配列

インスタンスを要素に持つ配列(1)

同じ型のデータをまとめて保持する機構に「配列」があった。ここで扱うのは、同じクラスから作成した複数のインスタンスを配列を用いてまとめる方法である。

```
void start() {
    Island island0 = createIsland("沖縄島", 1206.49);
    Island island1 = createIsland("淡路島", 592.17);
    Island island2 = createIsland("伊豆大島", 91.06);
    Island island3 = createIsland("与那国島", 28.91);
}
```

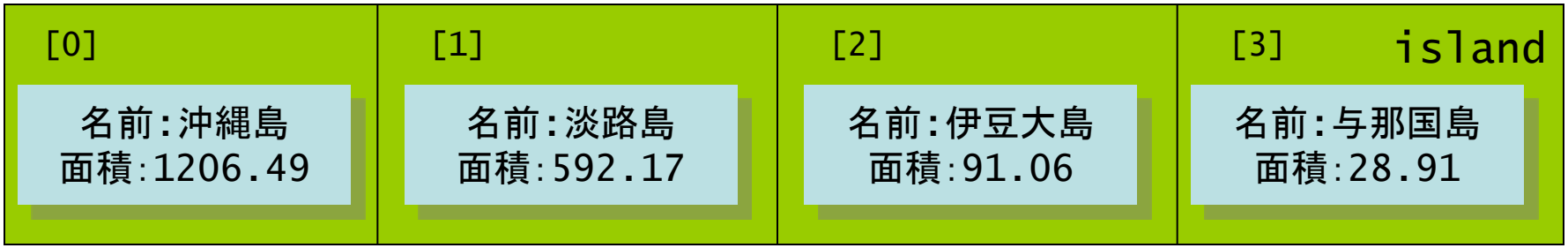
```
void start() {
    int numIsland = 4;
    Island [] island = new Island [numIsland];
    island[0] = createIsland("沖縄島", 1206.49);
    island[1] = createIsland("淡路島", 592.17);
    island[2] = createIsland("伊豆大島", 91.06);
    island[3] = createIsland("与那国島", 28.91);
}
```

■インスタンスを要素に持つ配列

例題41より

問題：クラスIslandのインスタンスを4個作成するプログラムを作成せよ。ただし、それぞれのインスタンスはIsland型の配列に格納するようにせよ。

返り値の型	メソッド名(引数)	機能
Island	*createIsland(String name, double area)*	*各インスタンス変数の値がそれぞれnameとareaであるIslandのインスタンスを作成し、返す。（前と同じ）*



（クラス名： Ex41Island）

例題41

```
1  package j2.lesson04;
2
3  public class Ex41Island {
4      public static void main(String[] args) {
5          new Ex41Island().start();
6      }
7
8      void start() {
9          int numIsland = 4;
10         Island [] island = new Island [numIsland];
11         island[0] = createIsland("沖縄島", 1206.49);
12         island[1] = createIsland("淡路島", 592.17);
13         island[2] = createIsland("伊豆大島", 91.06);
14         island[3] = createIsland("与那国島", 28.91);
15     }
16
17     Island createIsland(String name, double area){
18         Island newObj = new Island();
19         newObj.name = name;
20         newObj.area = area;
21         return newObj;
22     }
23
24 }
```

インスタンスを要素に持つ配列(2)

生成したインスタンスは、ある一つの事柄に関する情報を記録した一枚の情報カード（あるいはデータシート）と似ている。

配列は、同じフォーマットを持つ大量の情報カードを集めて束ね、保持しておくためのデータの貯蓄庫の役割を果たす。また、一つ一つの情報カードを簡単に参照することができる。

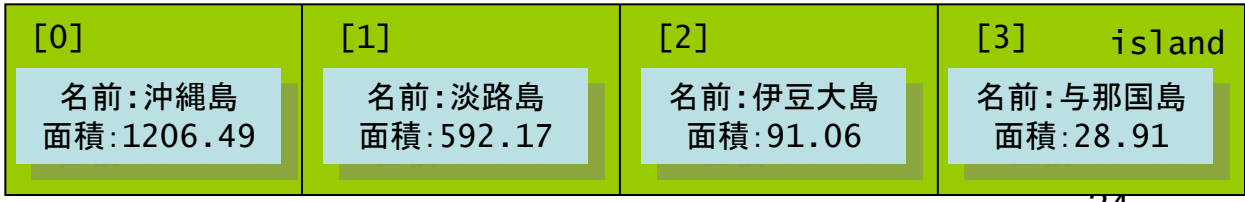
```
void start() {
    int numIsland = 4;
    Island [] island = new Island [numIsland];
    island[0] = createIsland("沖縄島", 1206.49);
    island[1] = createIsland("淡路島", 592.17);
    island[2] = createIsland("伊豆大島", 91.06);
    island[3] = createIsland("与那国島", 28.91);
}
```

インスタンスを要素に持つ配列(3)

このように集めたデータは、次のような一覧表（テーブル）の形で表すと分かりやすい。すなわち、一覧表の中で、インスタンスは、1行分のデータを保持する。これをレコードと呼ぶことがある。

島データ（island）

名前	面積
沖縄島	1206.49
淡路島	592.17
伊豆大島	91.06
与那国島	28.91



■例題42

問題：（例題41)の拡張）作成したIslandクラスのインスタンスの内容を表示せよ。

返り値の型	メソッド名(引数)	機能
Island	*createIsland(String name, double area)*	*各インスタンス変数の値がnameとareaであるIslandのインスタンスを作成し、返す。（前と同じ）*
void	*islandData(Island i, int row)*	*引数iの各インスタンス変数の値の文字列表現をSpreadsheetの row 行に表示する。*
void	*header()*	*Spreadsheet のヘッダ行を表示する。*

※既存のEx41Islandを編集する

（クラス名： Ex42Island）

例題42

```java
package j2.lesson04;

import gpjava.Spreadsheet;

public class Ex42Island {
    public static void main(String[] args) {
        new Ex42Island().start();
    }

    void start() {
        int numIslands = 4;
        Island [] island = new Island [numIslands];
        island[0] = createIsland("沖縄島", 1206.49);
        island[1] = createIsland("淡路島", 592.17);
        island[2] = createIsland("伊豆大島", 91.06);
        island[3] = createIsland("与那国島", 28.91);

        Spreadsheet.show();
        header();
        for (int i = 0; i < numIslands; i++) {
            islandData(island[i], i + 1);
        }
    }

    void header() {
        Spreadsheet.setString(0, 0, "名前");
        Spreadsheet.setString(0, 1, "面積");
    }

    void islandData(Island i, int row) {
        Spreadsheet.setString(row, 0, i.name);
        Spreadsheet.setString(row, 1, i.area + "km2");
    }

    Island createIsland(String name, double area){
        Island newObj = new Island();
        newObj.name = name;
        newObj.area = area;
        return newObj;
    }
}
```

例題42

```
1  package j2.lesson04;
2
3  import gpjava.Spreadsheet;
4
5  public class Ex42Island {
6      public static void main(String[] args) {
7          new Ex42Island().start();
8      }
9
10     void start() {
11         int numIslands = 4;
12         Island [] island = new Island [numIslands];
13         island[0] = createIsland("沖縄島", 1206.49);
14         island[1] = createIsland("淡路島", 592.17);
15         island[2] = createIsland("伊豆大島", 91.06);
16         island[3] = createIsland("与那国島", 28.91);
17
18         Spreadsheet.show();
19         header();
20         for (int i = 0; i < numIslands; i++) {
21             islandData(island[i], i + 1);
22         }
23     }
24
```

配列を用いることで、for文のインデックスiを用いて配列islandに格納したインスタンスを参照している。

なお、配列の要素数に変更があったとしても、（例えば、登録したいデータが増えたとしても）、この部分を変更する必要は無い。

```
islandData(island[0], 1);
islandData(island[1], 2);
islandData(island[2], 3);
islandData(island[3], 4);
```

まとめ：クラスとインスタンス（2）

- インスタンス生成とその利用（詳細）
- インスタンス（実体）と参照
- メソッドの参照呼び
- インスタンスを生成するメソッド
- インスタンスを要素に持つ配列

本日の例題と問題

- MyCircle のインスタンス作成例
  - Ex10, Ex11, Ex12, Q10, Q11
- MyCircle の膨張・収縮の例
  - Ex21, Ex22, Ex23*, Q20, Q21
- Island のインスタンス作成例
  - Ex30, Ex31, Ex32, Ex33*, Ex34, Q30, Q31, Q32, Q33*
- インスタンスを格納する配列の作成例
  - Ex41, Ex42, Ex43, Q12, Q22, Q34, Q35*, (Q41*)

(Ex:例題, Q:問題, *は少し手間のかかる問題)

各自に適した順番で解けばよいが、上記の順番が自然な流れとなるよう構成されている。

例題集

パッケージ「j2.lesson04」を作成する。

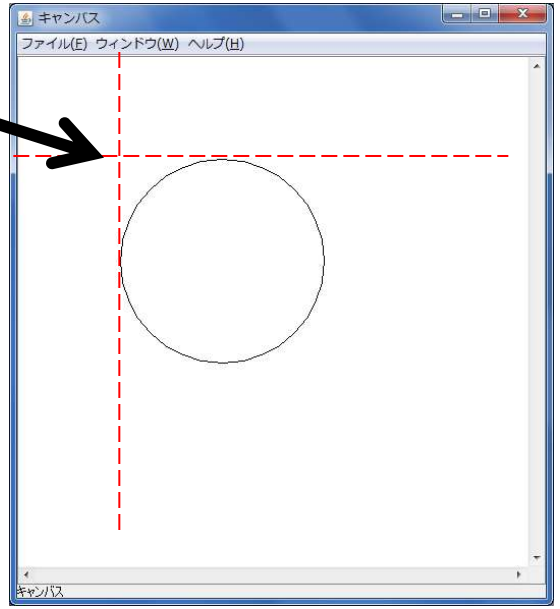
パッケージやクラスの作成，実行の仕方の説明は省略する。
作り方を忘れた場合は過去のスライドや
http://java2010.cis.k.hosei.ac.jp/01/material-01/
を参考にせよ

■例題10

問題：次のクラスMyCircleクラスを作成せよ。

インスタンス変数	初期値	説明
int　x	無し	x座標(※)
int　y	無し	y座標(※)
int　diameter	無し	円の直径

(x,y)

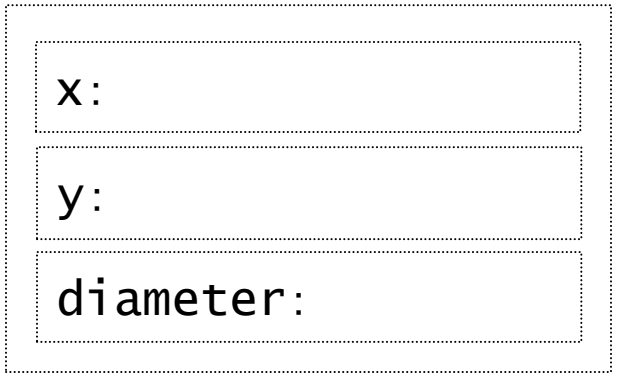


(クラス名： MyCircle)

例題10

```
1  package j2.lesson04;
2
3  public class MyCircle {
4      int x;
5      int y;
6      int diameter;
7  }
```

クラスMyCircle
初期値は無し



■例題11(1)

問題：

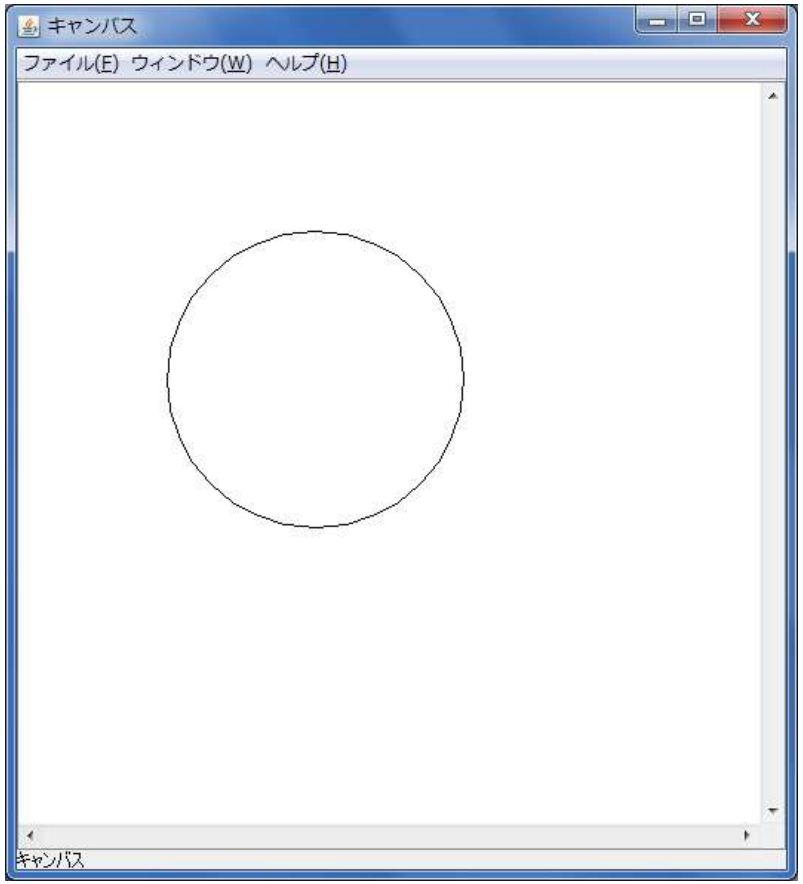
(a) MyCircleクラスのインスタンスを2つ作成せよ。ただし、同一の座標,直径を持つようにせよ。

(b) 次にこれらの円をキャンバス上に表示させよ。

返り値の型	メソッド名(引数)	機能
MyCircle	createMyCircle(int x, int y, int diameter)	MyCircleクラスのインスタンス変数の値がそれぞれx,y,diameterであるMyCircleインスタンスを作成し、返す。
void	drawMyCircle(MyCircle c)	キャンバスに引数のMyCircleインスタンスcを描画する。

（クラス名： Ex11TwoCircles）

実行例

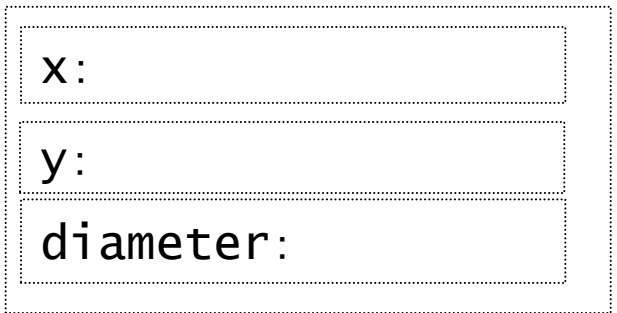


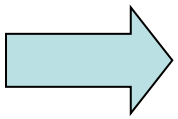
例題11(1)

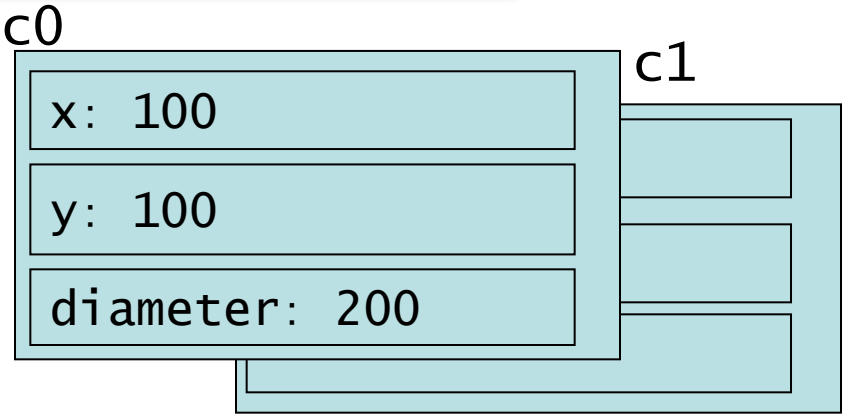
```java
package j2.lesson04;

import gpjava.Canvas;

public class Ex11TwoCircles {
    public static void main(String[] args) {
        new Ex11TwoCircles().start();
    }

    void start() {
        MyCircle c0 = createMyCircle(100, 100, 200);
        MyCircle c1 = createMyCircle(100, 100, 200);

        Canvas.show();
        drawMyCircle(c0);
        drawMyCircle(c1);
    }

    MyCircle createMyCircle(int x, int y, int diameter) {
        MyCircle newObj = new MyCircle();
        newObj.x = x;
        newObj.y = y;
        newObj.diameter = diameter;
        return newObj;
    }

    void drawMyCircle(MyCircle c) {
        Canvas.drawOval(c.x, c.y, c.diameter, c.diameter);
    }
}
```

2つのインスタンスを作成

クラスMyCircle
初期値は無し







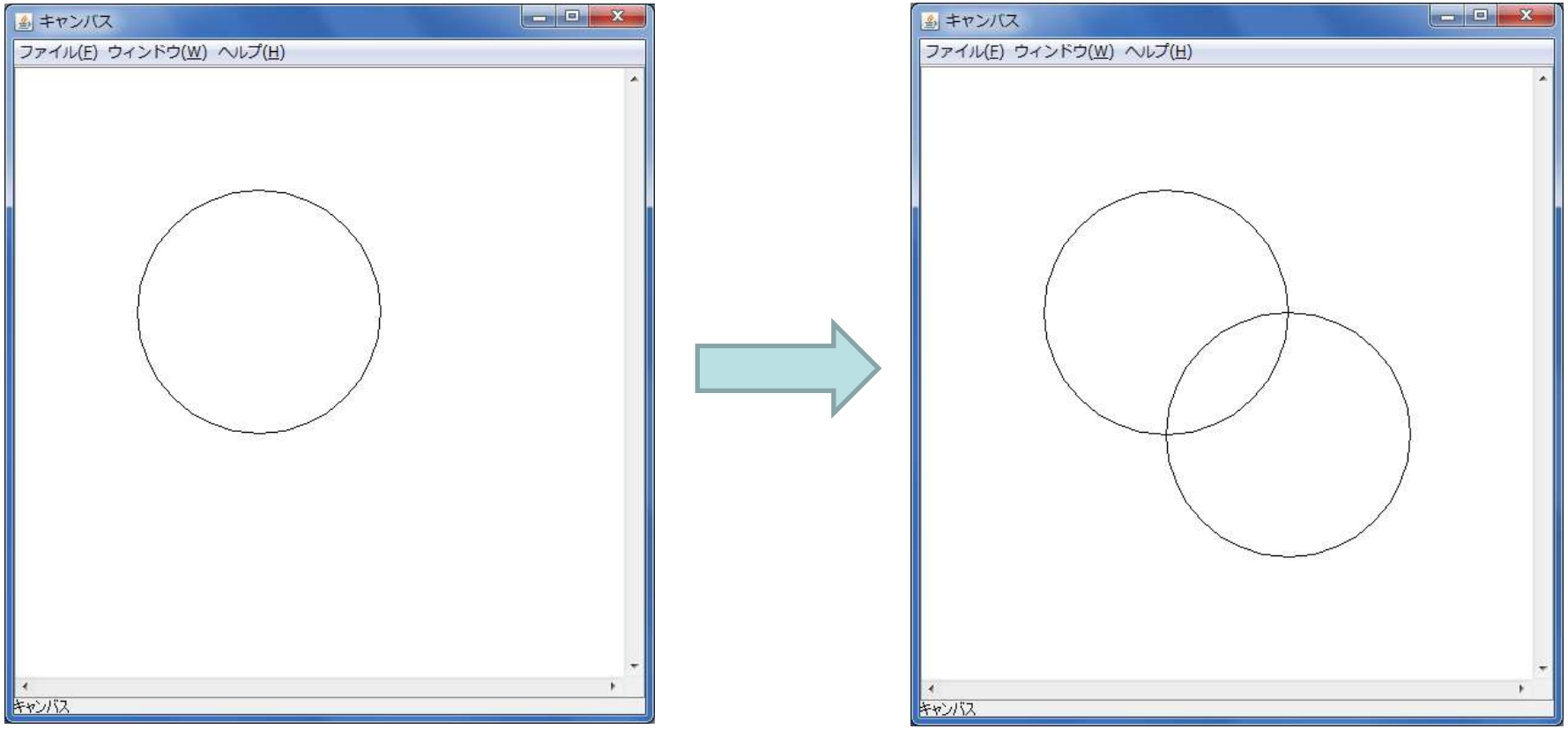
■例題11(2)

問題：例題11(1)に拡張を加える。
(a) 一定の時間ののち、キャンバスの内容を一度消す。
(b) 片方のインスタンスの座標を変更せよ。
(c) 二つの円を描画しなおせ。

返り値の型	メソッド名(引数)	機能
MyCircle	createMyCircle(int x, int y, int diameter)	MyCircleクラスのインスタンス変数の値がそれぞれx,y,diameterであるMyCircleインスタンスを作成し、返す。
void	drawMyCircle(MyCircle c)	キャンバスに引数のMyCircleインスタンスcを描画する。

※既存のEx11TwoCirclesを編集する

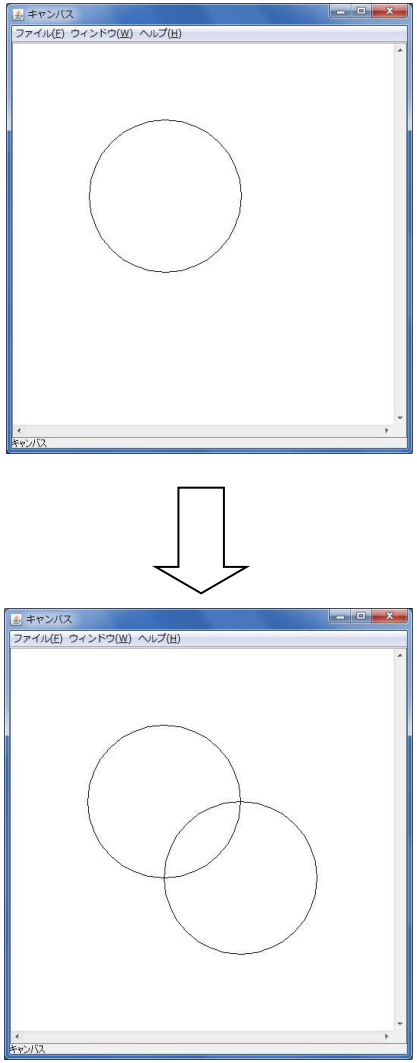
実行例


キャンバス
ファイル(F) ウィンドウ(W) ヘルプ(H)
キャンバス
キャンバス
ファイル(F) ウィンドウ(W) ヘルプ(H)
キャンバス

例題11(2)

```java
package j2.lesson04;

import gpjava.Canvas;

public class Ex11TwoCircles_2 {
    public static void main(String[] args) {
        new Ex11TwoCircles_2().start();
    }

    void start() {
        MyCircle c0 = createMyCircle(100, 100, 200);
        MyCircle c1 = createMyCircle(100, 100, 200);

        Canvas.show();
        drawMyCircle(c0);
        drawMyCircle(c1);
        Canvas.waitForCountdown(1500);
        Canvas.clear();
        c1.x = 200;
        c1.y = 200;
        drawMyCircle(c0);
        drawMyCircle(c1);
    }

    MyCircle createMyCircle(int x, int y, int diameter) {
        MyCircle newObj = new MyCircle();
        newObj.x = x;
        newObj.y = y;
        newObj.diameter = diameter;
        return newObj;
    }

    void drawMyCircle(MyCircle c) {
        Canvas.drawOval(c.x, c.y, c.diameter, c.diameter);
    }
}
```

c1のみを変更



■例題12(1)

問題:

(a) MyCircleクラスのインスタンスを1つ作成せよ。(インスタンスを参照する変数名はc0)

(b) このインスタンスを別の変数名c1で参照させよ。

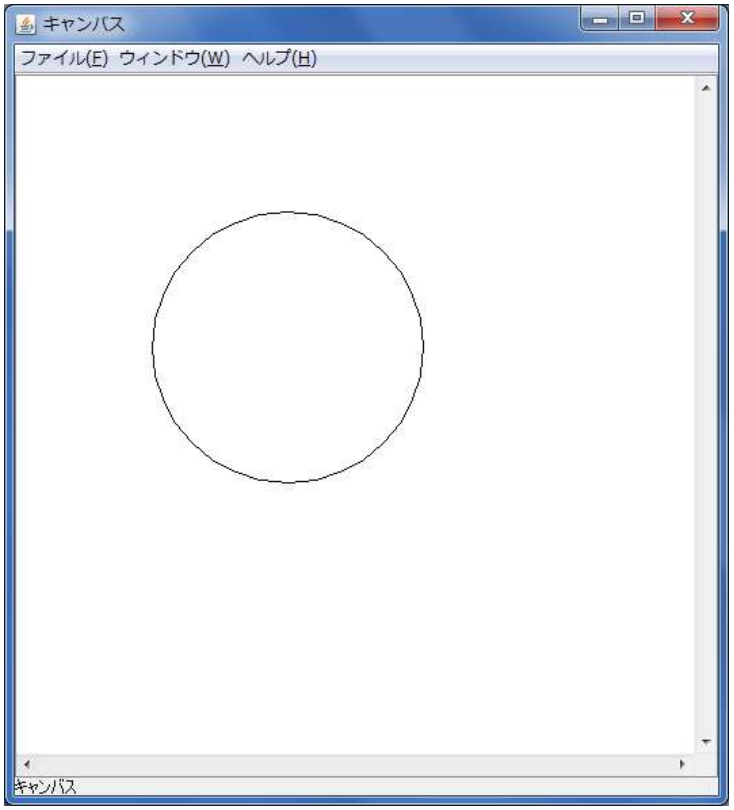
（インスタンスが二つの名前を持つ）

(c) 変数c0,c1の両方に対して、メソッドdrawMyCircleを適用せよ。

返り値の型	メソッド名(引数)	機能
MyCircle	createMyCircle(int x, int y, int diameter)	MyCircleクラスのインスタンス変数の値がそれぞれx,y,diameterであるMyCircleインスタンスを作成し、返す。
void	drawMyCircle(MyCircle c)	キャンバスに引数のMyCircleインスタンスcを描画する。

（クラス名： Ex12Circle）

実行例



例題12(1)

```
1  package j2.lesson04;
2
3  import gpjava.Canvas;
4
5  public class Ex12Circle {
6      public static void main(String[] args) {
7          new Ex12Circle().start();
8      }
9
10     void start() {
11         MyCircle c0 = createMyCircle(100, 100, 200);
12         MyCircle c1 = c0;
13
14         Canvas.show();
15         drawMyCircle(c0);
16         drawMyCircle(c1);
17     }
18
19     // Ex11と同じ
20     MyCircle createMyCircle(int x, int y, int diameter) {..
27
28     // Ex11と同じ
29     void drawMyCircle(MyCircle c) {..
32 }
```

インスタンスは一つ

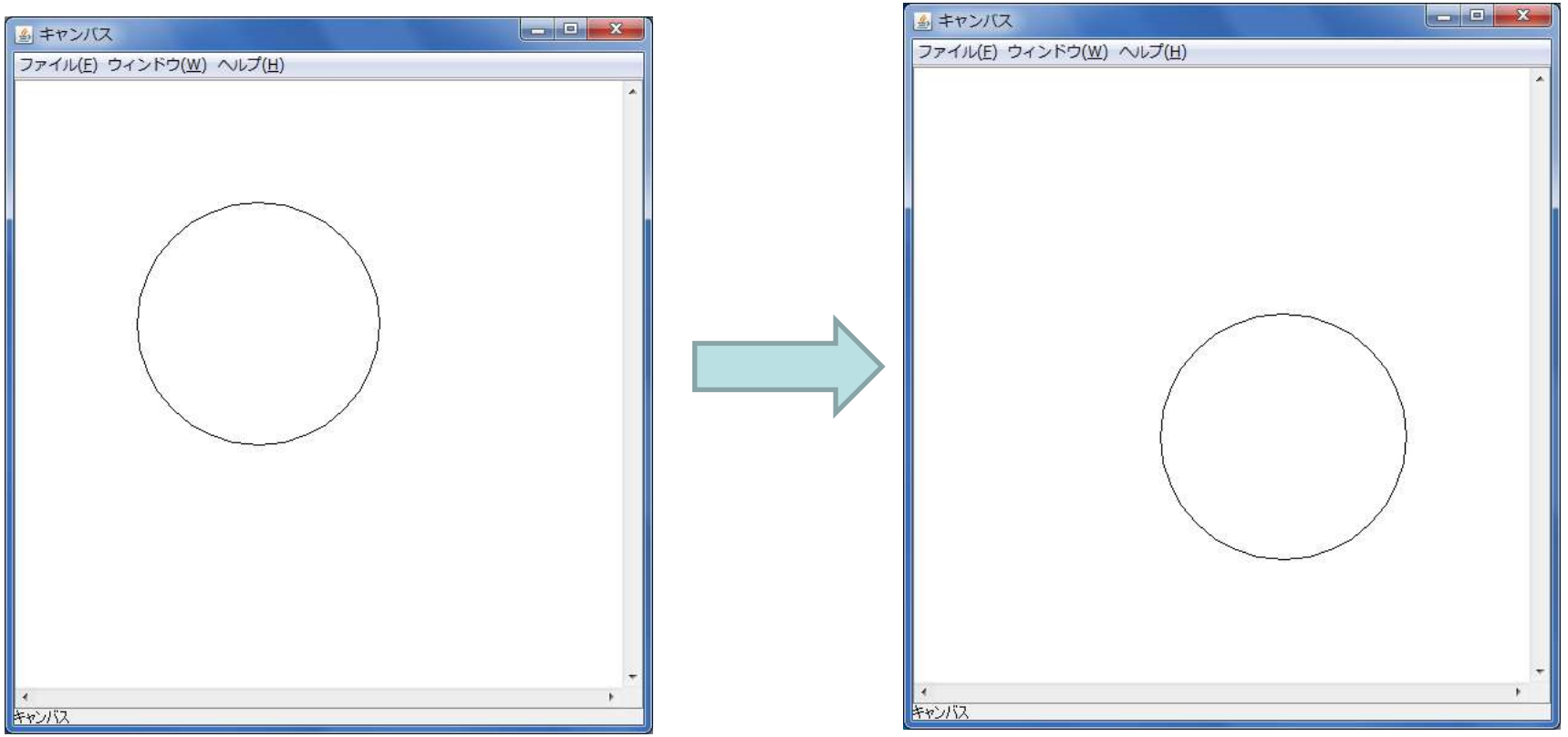
■例題12(2)

問題：例題12(1)に拡張を加える。
(a) 一定の時間ののち、キャンバスの内容を一度消す
(b) c1の座標を変更せよ。
(c) 二つの円を描画しなおせ。
上のプログラムを実行した場合、キャンバスにどのような絵が描画されるか予想せよ。

返り値の型	メソッド名(引数)	機能
MyCircle	createMyCircle(int x, int y, int diameter)	MyCircleクラスのインスタンス変数の値がそれぞれx,y,diameterであるMyCircleインスタンスを作成し、返す。
void	drawMyCircle(MyCircle c)	キャンバスに引数のMyCircleインスタンスcを描画する。

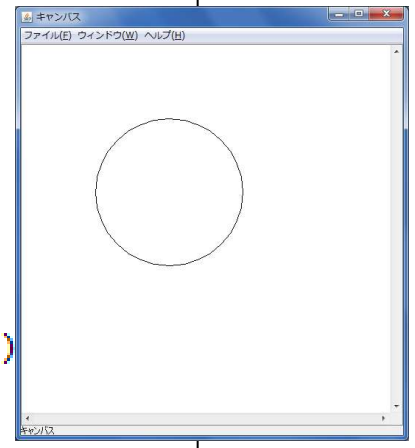
※既存のEx12Circleを編集する

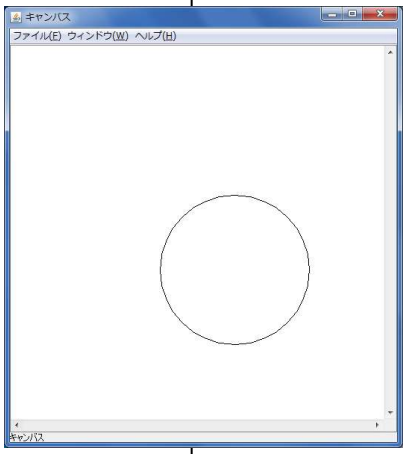
実行例


キャンバス
ファイル(F) ウィンドウ(W) ヘルプ(H)
キャンバス
キャンバス
ファイル(F) ウィンドウ(W) ヘルプ(H)
キャンバス

例題12(2)

```
1  package j2.lesson04;
2
3  import gpjava.Canvas;
4
5  public class Ex12Circle {
6      public static void main(String[] args) {
7          new Ex12Circle().start();
8      }
9
10     void start() {
11         MyCircle c0 = createMyCircle(100, 100, 200);
12         MyCircle c1 = c0;
13
14         Canvas.show();
15         drawMyCircle(c0);
16         drawMyCircle(c1);
17
18         Canvas.waitForCountdown(1500);
19         Canvas.clear();
20
21         c1.x = 200;
22         c1.y = 200;
23
24         drawMyCircle(c0);
25         drawMyCircle(c1);
26     }
27
28     // Ex11と同じ
29     MyCircle createMyCircle(int x, int y, int diameter) {..
36
37     // Ex11と同じ
38     void drawMyCircle(MyCircle c) {..
41 }
```





c1のみを変更、
のように思えるが…

c0とc1は同じ実体を指しているので、例題11(2)のようにはならない

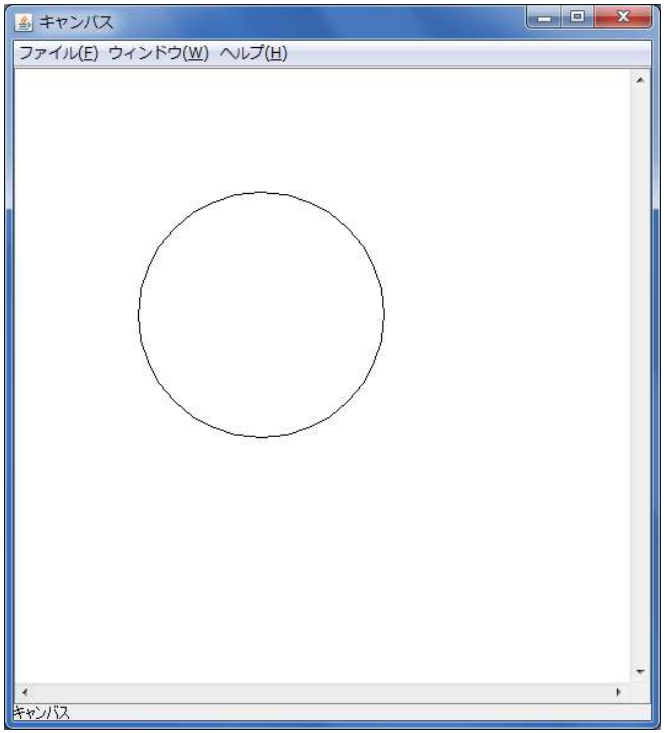
■例題21

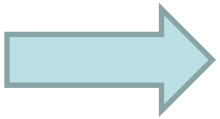
問題：円を膨張（収縮）させるメソッドを作成し、動作を確かめよ。

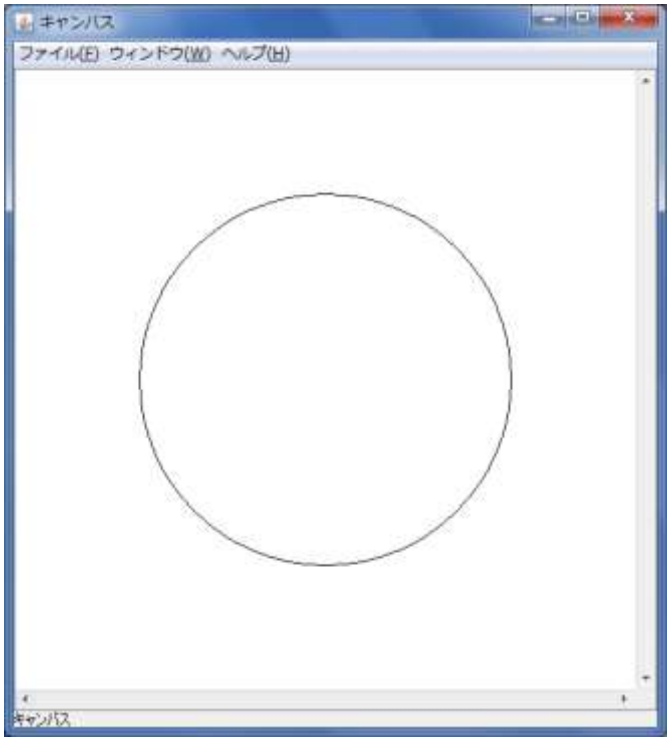
返り値の型	メソッド名(引数)	機能
MyCircle	createMyCircle(int x, int y, int diameter)	MyCircleクラスのインスタンス変数の値がそれぞれx,y,diameterであるMyCircleインスタンスを作成し、返す。
void	drawMyCircle(MyCircle c)	キャンバスに引数のMyCircleインスタンスcを描画する。
void	expandMyCircleBy(MyCircle c, double factor)	MyCircle cのインタスンス変数diameterの値をfactor倍する。

（クラス名： Ex21ExpandCircle）

実行例


キャンバス
ファイル(F) ウィンドウ(W) ヘルプ(H)
キャンバス




キャンバス
ファイル(F) ウィンドウ(W) ヘルプ(H)
キャンバス

例題21

```
1  package j2.lesson04;
2
3  import gpjava.Canvas;
4
5  public class Ex21ExpandCircle {
6      public static void main(String[] args) {
7          new Ex21ExpandCircle().start();
8      }
9
10     void start() {
11         MyCircle c0 = createMyCircle(100, 100, 200);
12
13         Canvas.show();
14         drawMyCircle(c0);
15
16         Canvas.waitForCountdown(1500);
17         Canvas.clear();
18         expandMyCircleBy(c0, 1.5);
19         drawMyCircle(c0);
20
21     }
22
23     MyCircle createMyCircle(int x, int y, int diameter) {
24         MyCircle newObj = new MyCircle();
25         newObj.x = x;
26         newObj.y = y;
27         newObj.diameter = diameter;
28         return newObj;
29     }
30
31     void drawMyCircle(MyCircle c) {
32         Canvas.drawOval(c.x, c.y, c.diameter, c.diameter);
33     }
34
35     void expandMyCircleBy(MyCircle c, double factor) {
36         c.diameter *= factor;
37     }
38 }
```

渡された参照を利用してインスタンスの直径を書き換えている。

■例題22(1)

問題:

(a) MyCircleクラスのインスタンスを2つ作成し、これをもとに円を描画せよ。

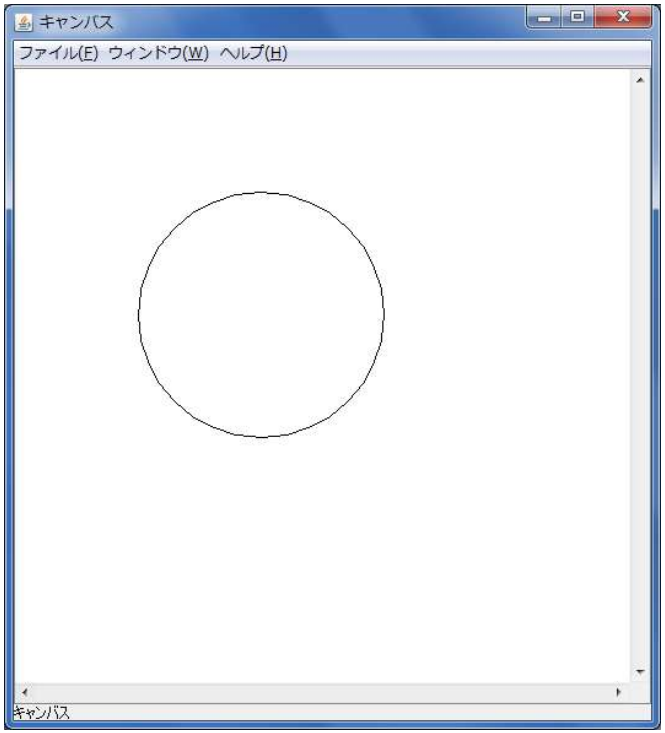
(b) 次に片方のインスタンスの大きさをメソッドexpandMyCircleByを用いて変更せよ。

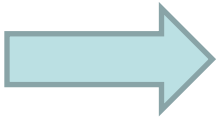
(c) キャンバスの内容を一度消した後に、二つの円を描画しなおせ。

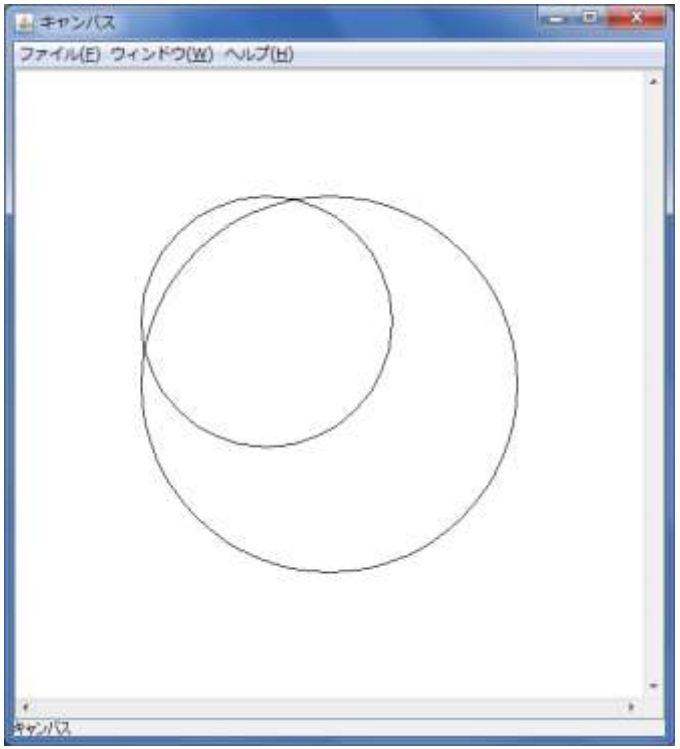
返り値の型	メソッド名(引数)	機能
MyCircle	createMyCircle(int x, int y, int diameter)	MyCircleクラスのインスタンス変数の値がそれぞれx,y,diameterであるMyCircleインスタンスを作成し、返す。
void	drawMyCircle(MyCircle c)	キャンバスに引数のMyCircleインスタンスcを描画する。
void	expandMyCircleBy(MyCircle c, double factor)	MyCircle cのインタスンス変数diameterの値をfactor倍する。

(クラス名: Ex22ExpandCircle)

実行例


キャンバス
ファイル(F) ウィンドウ(W) ヘルプ(H)
キャンバス

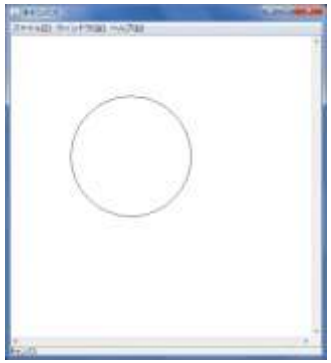


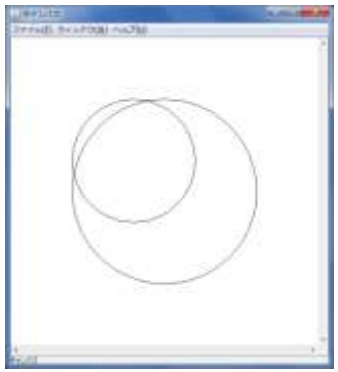

キャンバス
ファイル(F) ウィンドウ(W) ヘルプ(H)
キャンバス

例題22(1)

```
1  package j2.lesson04;
2
3  import gpjava.Canvas;
4
5  public class Ex22ExpandCircle {
6      public static void main(String[] args) {
7          new Ex22ExpandCircle().start();
8      }
9
10     void start() {
11         MyCircle c0 = createMyCircle(100, 100, 200);
12         MyCircle c1 = createMyCircle(100, 100, 200);
13
14         Canvas.show();
15         drawMyCircle(c0);
16         drawMyCircle(c1);
17
18         Canvas.waitForCountdown(1500);
19         Canvas.clear();
20         expandMyCircleBy(c1, 1.5);
21         drawMyCircle(c0);
22         drawMyCircle(c1);
23
24     }
25
26     MyCircle createMyCircle(int x, int y, int diameter) {
27         MyCircle newObj = new MyCircle();
28         newObj.x = x;
29         newObj.y = y;
30         newObj.diameter = diameter;
31         return newObj;
32     }
33
34     void drawMyCircle(MyCircle c) {
35         Canvas.drawOval(c.x, c.y, c.diameter, c.diameter);
36     }
37
38     void expandMyCircleBy(MyCircle c, double factor) {
39         c.diameter *= factor;
40     }
41 }
```

c1のみを変更





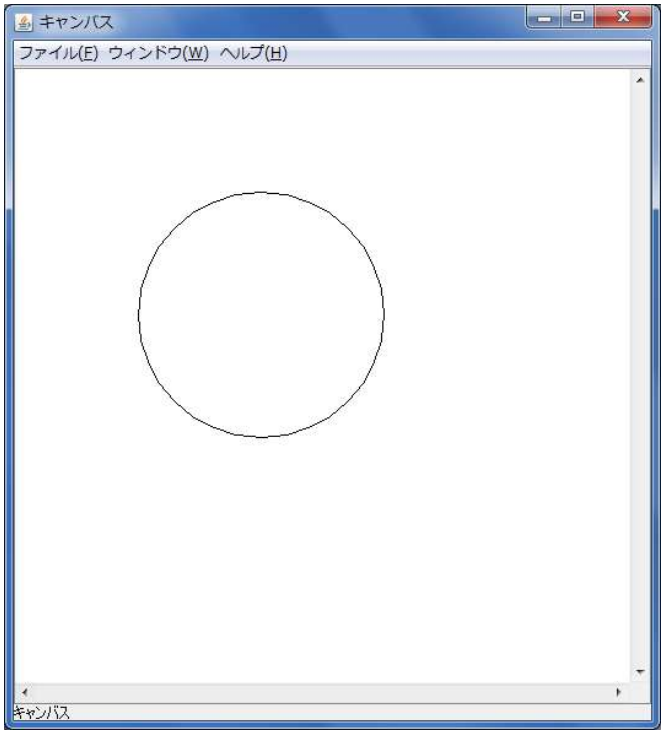
■例題22(2)

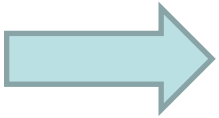
問題:(a) MyCircleクラスのインスタンスを1つ作成せよ。
(b) このインスタンスを別の変数名c1で参照させせよ。
（インスタンスが二つの名前を持つ）
(c) キャンバスに2つの円を描画せよ
(d) 次にc1のインスタンスの大きさをメソッド
expandMyCircleByを用いて変更せよ。
(e) キャンバスの内容を一度消した後に、2つの円を描画
しなおせ。

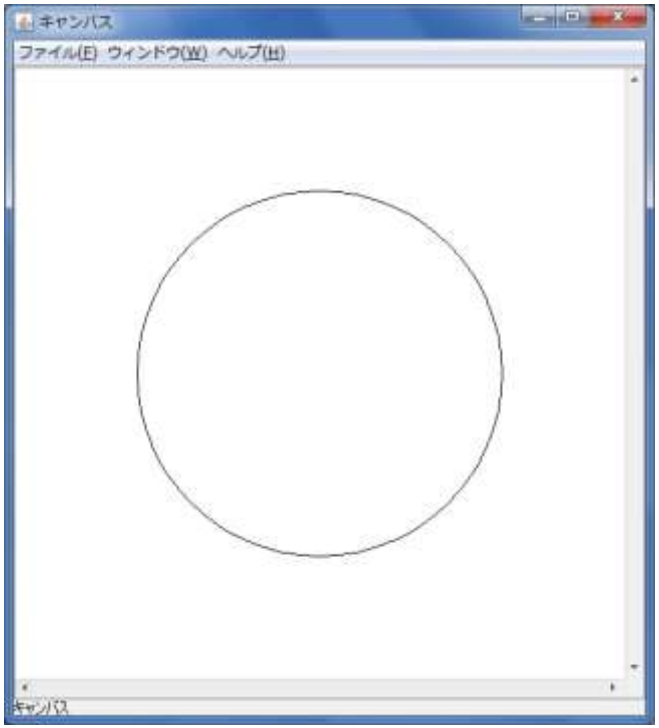
返り値の型	メソッド名(引数)	機能
MyCircle	createMyCircle(int x, int y, int diameter)	MyCircleクラスのインスタンス変数の値がそれぞれx,y,diameterであるMyCircleインスタンスを作成し、返す。
void	drawMyCircle(MyCircle c)	キャンバスに引数のMyCircleインスタンスcを描画する。
void	expandMyCircleBy(MyCircle c, double factor)	MyCircle cのインタスンス変数diameterの値をfactor倍する。

※既存のEx22ExpandCircleを編集する

実行例


キャンバス
ファイル(F) ウィンドウ(W) ヘルプ(H)
キャンバス

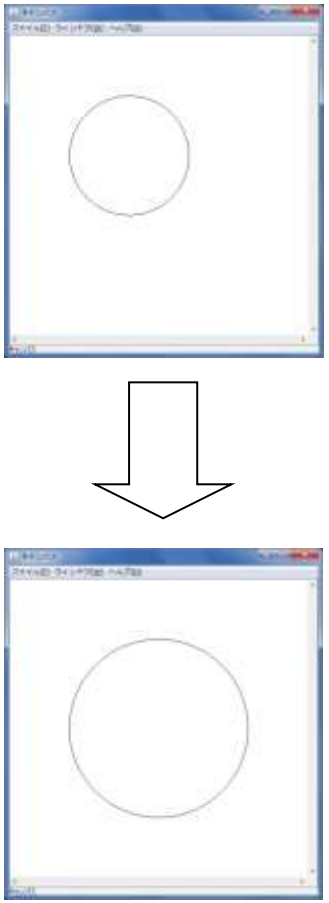



キャンバス
ファイル(F) ウィンドウ(W) ヘルプ(H)
キャンバス

例題22(2)

```
1  package j2.lesson04;
2
3  import gpjava.Canvas;
4
5  public class Ex22ExpandCircle_2 {
6      public static void main(String[] args) {
7          new Ex22ExpandCircle_2().start();
8      }
9
10     void start() {
11         MyCircle c0 = createMyCircle(100, 100, 200);
12         MyCircle c1 = c0;
13
14         Canvas.show();
15         drawMyCircle(c0);
16         drawMyCircle(c1);
17
18         Canvas.waitForCountdown(1500);
19         Canvas.clear();
20         expandMyCircleBy(c1, 1.5);
21         drawMyCircle(c0);
22         drawMyCircle(c1);
23
24     }
25
26     MyCircle createMyCircle(int x, int y, int diameter) {
27         MyCircle newObj = new MyCircle();
28         newObj.x = x;
29         newObj.y = y;
30         newObj.diameter = diameter;
31         return newObj;
32     }
33
34     void drawMyCircle(MyCircle c) {
35         Canvas.drawOval(c.x, c.y, c.diameter, c.diameter);
36     }
37
38     void expandMyCircleBy(MyCircle c, double factor) {
39         c.diameter *= factor;
40     }
41 }
```

c1のみを変更…
のように見えるが…



c0とc1は同じ実体を指しているので、例題22(1)のようにはならない

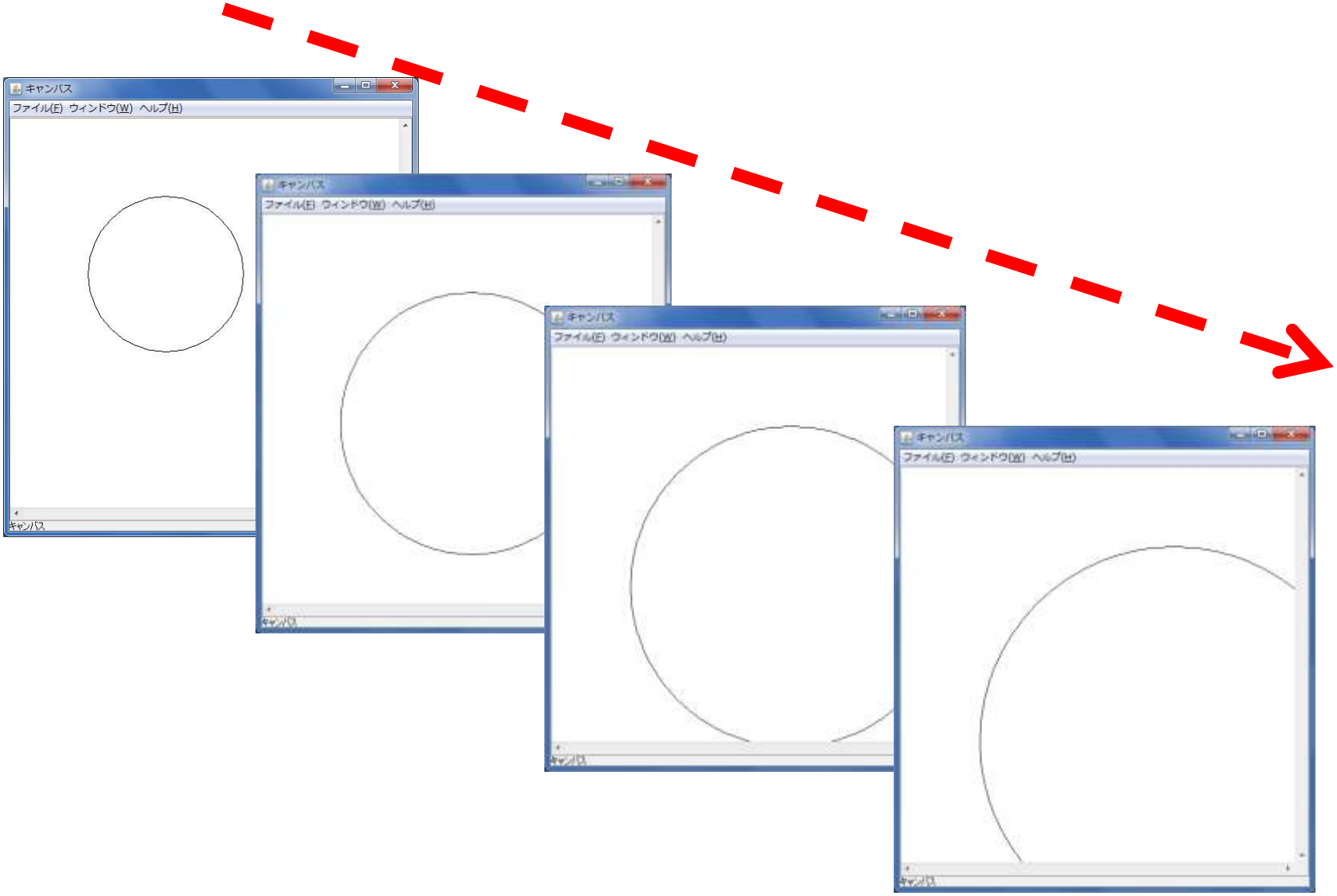
■例題23

問題：円が拡大していく様子をアニメーションで表現せよ。

返り値の型	メソッド名(引数)	機能
MyCircle	createMyCircle(int x, int y, int diameter)	MyCircleクラスのインスタンス変数の値がそれぞれx,y,diameterであるMyCircleインスタンスを作成し、返す。
void	drawMyCircle(MyCircle c)	キャンバスに引数のMyCircleインスタンスcを描画する。
void	expandMyCircleBy(MyCircle c, double factor)	MyCircle cのインタスンス変数diameterの値をfactor倍する。

（クラス名： Ex23ExpandingCircle）

実行例



例題23

```java
package j2.lesson04;

import gpjava.Canvas;

public class Ex23ExpandingCircle {
    public static void main(String[] args) {
        new Ex23ExpandingCircle().start();
    }

    void start() {
        MyCircle c0 = createMyCircle(100, 100, 200);

        Canvas.show();
        for(int i = 0; i < 20; i++) {
            Canvas.clear();
            drawMyCircle(c0);
            Canvas.waitForCountdown(200);
            expandMyCircleBy(c0, 1.05);
        }
    }

    MyCircle createMyCircle(int x, int y, int diameter) {
        MyCircle newObj = new MyCircle();
        newObj.x = x;
        newObj.y = y;
        newObj.diameter = diameter;
        return newObj;
    }

    void drawMyCircle(MyCircle c) {
        Canvas.drawOval(c.x, c.y, c.diameter, c.diameter);
    }

    void expandMyCircleBy(MyCircle c, double factor) {
        c.diameter *= factor;
    }
}
```

■例題30

問題：次のクラスIslandを作成せよ。

クラスIslandのインスタンスは次のインスタンス変数を持ち、「島」の情報を保持する。（初期値は指定しなくて良い）

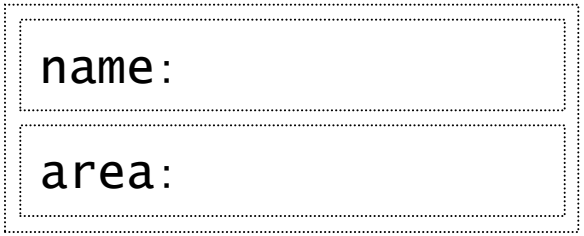
インスタンス変数	初期値	説明
String name	無し	島の名前
double area	無し	島の面積(km^2)

（クラス名： Island）

例題30

```
1  package j2.lesson04;
2
3  public class Island {
4      String name;  // 島の名前
5      double area; // 面積
6  }
```

クラスIsland
初期値は無し

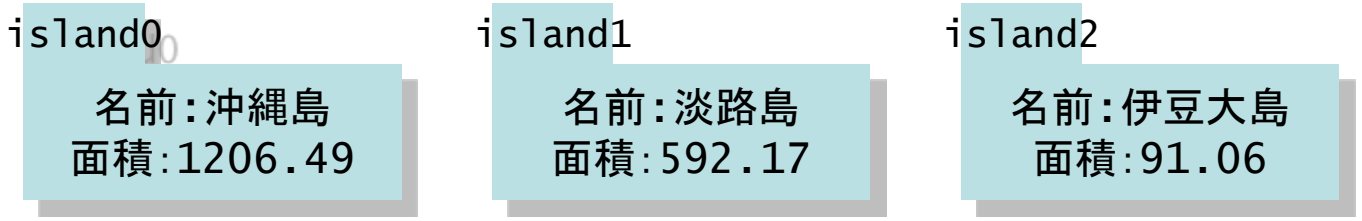


インスタンス変数	初期値	説明
String name	無し	島の名前
double area	無し	島の面積(km^2)

■例題31

問題：クラスIslandのインスタンスを3個作成するプログラムを作成せよ。ただし、インスタンスを生成する次のメソッドを作成し、これを利用せよ。

返り値の型	メソッド名(引数)	機能
Island	createIsland(String name, double area)	各インスタンス変数の値がそれぞれnameとareaであるIslandのインスタンスを作成し、返す。



（クラス名： Ex31Island）

例題31

```java
package j2.lesson04;

public class Ex31Island {
    public static void main(String[] args) {
        new Ex31Island().start();
    }

    void start() {
        Island island0 = createIsland("沖縄島", 1206.49);
        Island island1 = createIsland("淡路島", 592.17);
        Island island2 = createIsland("伊豆大島", 91.06);
    }

    Island createIsland(String name, double area) {
        Island newObj = new Island();
        newObj.name = name;
        newObj.area = area;
        return newObj;
    }
}
```

```java
package j2.lesson04;

public class Island {
    String name;  // 島の名前
    double area; // 面積
}
```

■例題32

問題：(例題31の拡張）作成したIslandクラスのインスタンスの内容を表示せよ。

返り値の型	メソッド名(引数)	機能
Island	*createIsland(String name, double area)*	*各インスタンス変数の値がnameとareaであるIslandのインスタンスを作成し、返す。（前のものと同じ）*
void	islandData(Island i, int row)	引数iの各インスタンス変数の値の文字列表現をSpreadsheetの row 行に表示する。
void	header()	Spreadsheet のヘッダ行を表示する。

※既存のEx31Islandを編集する

（クラス名： Ex32Island）

例題32

```
1  package j2.lesson04;
2
3  import gpjava.Spreadsheet;
4
5  public class Ex32Island {
6      public static void main(String[] args) {
7          new Ex32Island().start();
8      }
9
10     void start() {
11         Island island0 = createIsland("沖縄島", 1206.49);
12         Island island1 = createIsland("淡路島", 592.17);
13         Island island2 = createIsland("伊豆大島", 91.06);
14
15         Spreadsheet.show();
16         header();
17         islandData(island0, 1);
18         islandData(island1, 2);
19         islandData(island2, 3);
20     }
21
22     void header() {
23         Spreadsheet.setString(0, 0, "名前");
24         Spreadsheet.setString(0, 1, "面積");
25     }
26
27     void islandData(Island i, int row) {
28         Spreadsheet.setString(row, 0, i.name);
29         Spreadsheet.setString(row, 1, i.area + "km2");
30     }
31
32     Island createIsland(String name, double area){
33         Island newObj = new Island();
34         newObj.name = name;
35         newObj.area = area;
36         return newObj;
37     }
38 }
```

インスタンス生成メソッドを用いたことでmainメソッドが分かりやすくなった。

■例題33

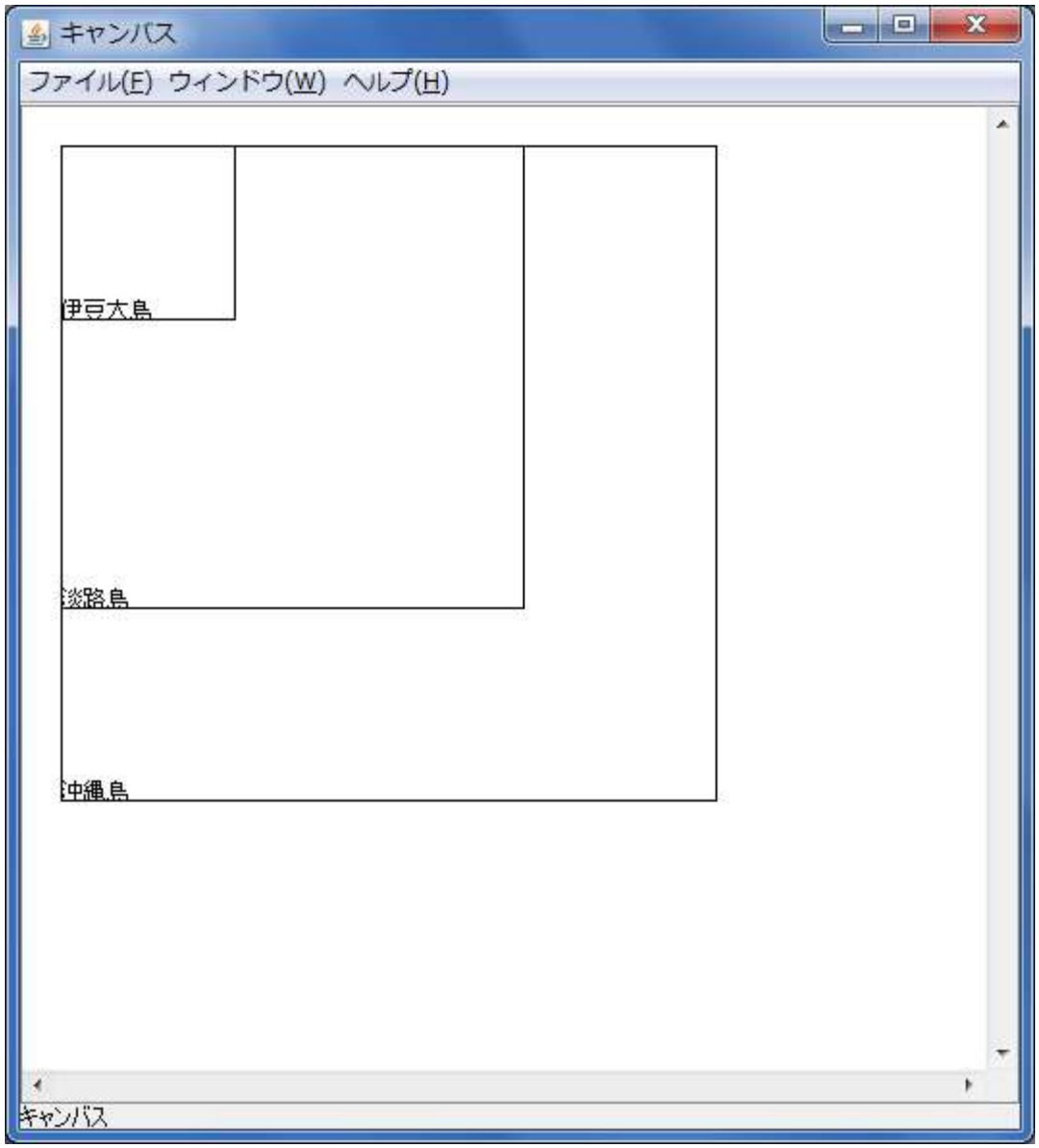
問題：(例題32の拡張）作成したIslandクラスのインスタンスの情報をもとにして、「島」の名前と、面積に比例する大きさの正方形で表示させよ。

返り値の型	メソッド名(引数)	機能
Island	*createIsland(String name, double area)*	*各インスタンス変数の値がnameとareaであるIslandのインスタンスを作成し、返す。（前のものと同じ）*
void	drawIslandSquare(Island island)	キャンバスに「島」の名前と、面積に比例する大きさの正方形を描画する。

※既存のEx32Islandを編集する

（クラス名： Ex33Island）

実行結果



例題33

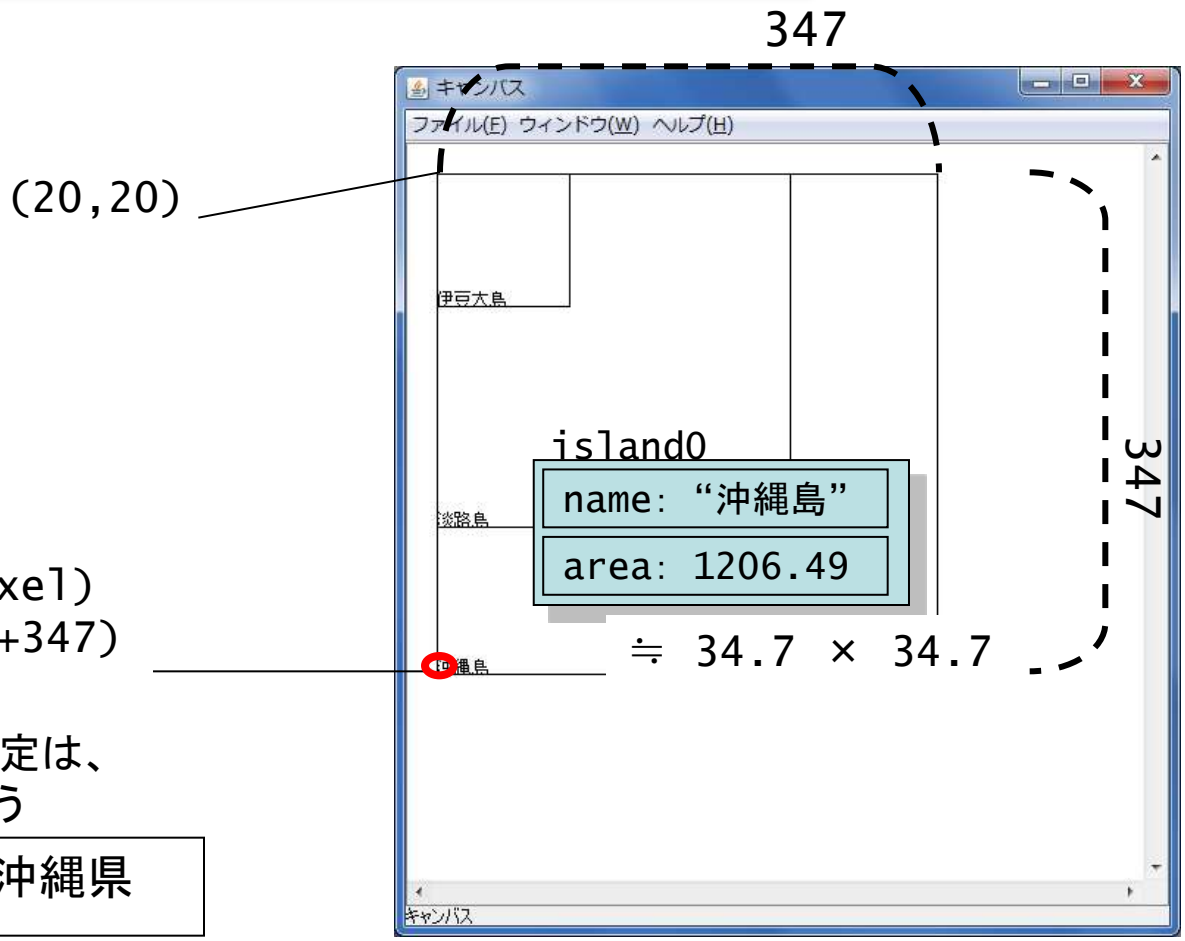
```
1  package j2.lesson04;
2
3  import gpjava.Canvas;
4
5  public class Ex33Island {
6      public static void main(String[] args) {
7          new Ex33Island().start();
8      }
9
10     void start() {
11         Island island0 = createIsland("沖縄島", 1206.49);
12         Island island1 = createIsland("淡路島", 592.17);
13         Island island2 = createIsland("伊豆大島", 91.06);
14
15         Canvas.show();
16         drawIslandSquare(island0);
17         drawIslandSquare(island1);
18         drawIslandSquare(island2);
19     }
20
21     Island createIsland(String name, double area){
22         Island newObj = new Island();
23         newObj.name = name;
24         newObj.area = area;
25         return newObj;
26     }
27
28     void drawIslandSquare(Island island) {
29         double length = Math.sqrt(island.area);
30         int pixel = (int)length * 10;
31         Canvas.drawRect(20, 20, pixel, pixel);
32         Canvas.drawString(20, 20+pixel, island.name);
33     }
34 }
```

例題33補足： 文字列を表示する機能 （drawString）

```
public static void drawIslandSquare(Island island) {
    double length = Math.sqrt(island.area);
    int pixel = (int)length * 10;
    Canvas.drawRect(20, 20, pixel, pixel);
    Canvas.drawString(20, 20+pixel, island.name);
}
```

Math.sqrt (1206.9) → 34.74….

pixel= 347



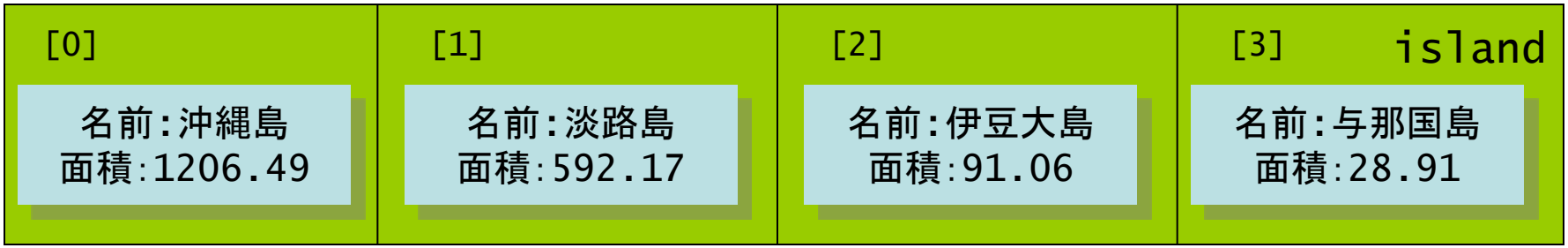
文字列の座標の指定は、
文字列の左下で行う

沖縄県

■例題41

問題：クラスIslandのインスタンスを4個作成するプログラムを作成せよ。ただし、それぞれのインスタンスはIsland型の配列に格納するようにせよ。

返り値の型	メソッド名(引数)	機能
Island	*createIsland(String name, double area)*	*各インスタンス変数の値がそれぞれnameとareaであるIslandのインスタンスを作成し、返す。（前と同じ）*



（クラス名： Ex41Island）

例題41

```
1  package j2.lesson04;
2
3  public class Ex41Island {
4      public static void main(String[] args) {
5          new Ex41Island().start();
6      }
7
8      void start() {
9          int numIsland = 4;
10         Island [] island = new Island [numIsland];
11         island[0] = createIsland("沖縄島", 1206.49);
12         island[1] = createIsland("淡路島", 592.17);
13         island[2] = createIsland("伊豆大島", 91.06);
14         island[3] = createIsland("与那国島", 28.91);
15     }
16
17     Island createIsland(String name, double area){
18         Island newObj = new Island();
19         newObj.name = name;
20         newObj.area = area;
21         return newObj;
22     }
23
24 }
```

■例題42

問題：（例題41)の拡張）作成したIslandクラスのインスタンスの内容を表示せよ。

返り値の型	メソッド名(引数)	機能
Island	*createIsland(String name, double area)*	*各インスタンス変数の値がnameとareaであるIslandのインスタンスを作成し、返す。（前と同じ）*
void	*islandData(Island i, int row)*	*引数iの各インスタンス変数の値の文字列表現をSpreadsheetの row 行に表示する。*
void	*header()*	*Spreadsheet のヘッダ行を表示する。*

※既存のEx41Islandを編集する

（クラス名： Ex42Island）

例題42

```
1  package j2.lesson04;
2
3  import gpjava.Spreadsheet;
4
5  public class Ex42Island {
6      public static void main(String[] args) {
7          new Ex42Island().start();
8      }
9
10     void start() {
11         int numIslands = 4;
12         Island [] island = new Island [numIslands];
13         island[0] = createIsland("沖縄島", 1206.49);
14         island[1] = createIsland("淡路島", 592.17);
15         island[2] = createIsland("伊豆大島", 91.06);
16         island[3] = createIsland("与那国島", 28.91);
17
18         Spreadsheet.show();
19         header();
20         for (int i = 0; i < numIslands; i++) {
21             islandData(island[i], i + 1);
22         }
23     }
24
25     void header() {
26         Spreadsheet.setString(0, 0, "名前");
27         Spreadsheet.setString(0, 1, "面積");
28     }
29
30     void islandData(Island i, int row) {
31         Spreadsheet.setString(row, 0, i.name);
32         Spreadsheet.setString(row, 1, i.area + "km2");
33     }
34
35     Island createIsland(String name, double area){
36         Island newObj = new Island();
37         newObj.name = name;
38         newObj.area = area;
39         return newObj;
40     }
41 }
```

例題42

```
1  package j2.lesson04;
2
3  import gpjava.Spreadsheet;
4
5  public class Ex42Island {
6      public static void main(String[] args) {
7          new Ex42Island().start();
8      }
9
10     void start() {
11         int numIslands = 4;
12         Island [] island = new Island [numIslands];
13         island[0] = createIsland("沖縄島", 1206.49);
14         island[1] = createIsland("淡路島", 592.17);
15         island[2] = createIsland("伊豆大島", 91.06);
16         island[3] = createIsland("与那国島", 28.91);
17
18         Spreadsheet.show();
19         header();
20         for (int i = 0; i < numIslands; i++) {
21             islandData(island[i], i + 1);
22         }
23     }
24
```

配列を用いることで、for文のインデックスiを用いて配列islandに格納したインスタンスを参照している。

なお、配列の要素数に変更があったとしても、（例えば、登録したいデータが増えたとしても）、この部分を変更する必要は無い。

```
islandData(island[0], 1);
islandData(island[1], 2);
islandData(island[2], 3);
islandData(island[3], 4);
```

■例題43

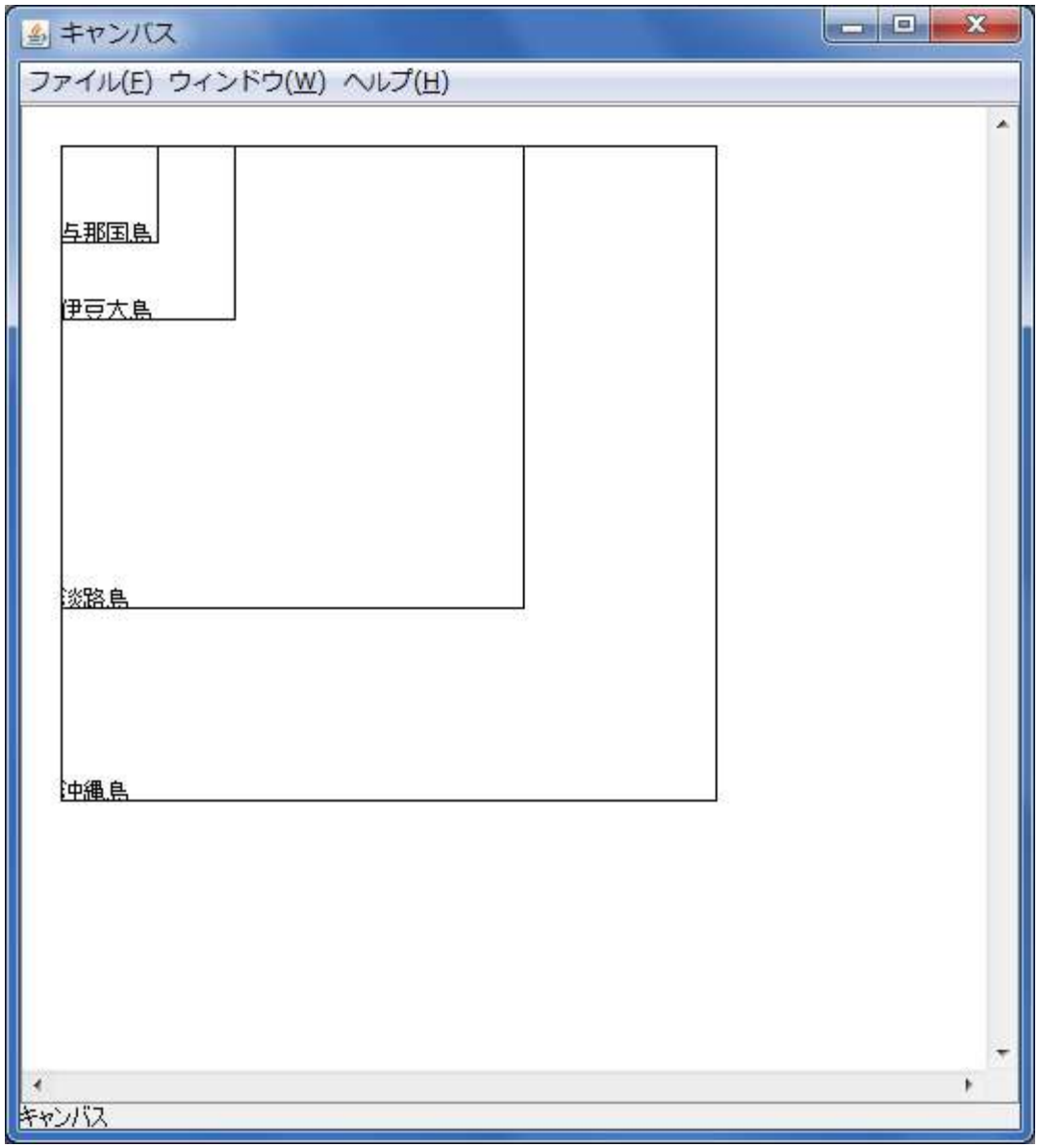
問題：(例題42の拡張）作成したIslandクラスのインスタンスの情報をもとにして、「島」の名前と、面積に比例する大きさの正方形で表示させよ。

返り値の型	メソッド名(引数)	機能
Island	*createIsland(String name, double area)*	*各インスタンス変数の値がnameとareaであるIslandのインスタンスを作成し、返す。（前のものと同じ）*
void	*drawIslandSquare(Island island)*	*キャンバスに「島」の名前と、面積に比例する大きさの正方形を描画する。（前のものと同じ）*

※既存のEx42Islandを編集する

（クラス名： Ex43Island）

実行結果



例題43

```
1  package j2.lesson04;
2
3  import gpjava.Canvas;
4
5  public class Ex43Island {
6      public static void main(String[] args) {
7          new Ex43Island().start();
8      }
9
10     void start() {
11         int numIslands = 4;
12         Island [] island = new Island [numIslands];
13         island[0] = createIsland("沖縄島", 1206.49);
14         island[1] = createIsland("淡路島", 592.17);
15         island[2] = createIsland("伊豆大島", 91.06);
16         island[3] = createIsland("与那国島", 28.91);
17
18         Canvas.show();
19         for (int i = 0; i < numIslands; i++) {
20             drawIslandSquare(island[i]);
21         }
22     }
23
24     Island createIsland(String name, double area){
25         Island newObj = new Island();
26         newObj.name = name;
27         newObj.area = area;
28         return newObj;
29     }
30
31     void drawIslandSquare(Island island) {
32         double length = Math.sqrt(island.area);
33         int pixel = (int)length * 10;
34         Canvas.drawRect(20, 20, pixel, pixel);
35         Canvas.drawString(20, 20+pixel, island.name);
36     }
37 }
```