プログラミング入門2

第14回　String, まとめ

テーマ: 等価性とStringクラス

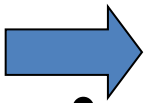
- インスタンスの等価性の評価
- String クラス
- まとめ

先週の復習: 連想記憶 HashMap

- java.util.HashMap として提供される
- キーを指定して、キーと関連付けれられたデータを検索できるライブラリ
- (初期化)　　HashMap map = new HashMap();
- (登録)　　map.put(“キー”, “値”);
- (検索)　　String value
  = (Striing)map.get(“キー”);
- (キー一覧)　　Set keys = map.keySet();

本日の課題： キーの等価性は、どのように判断されるのだろうか？

テーマ: 等価性と Stringクラス

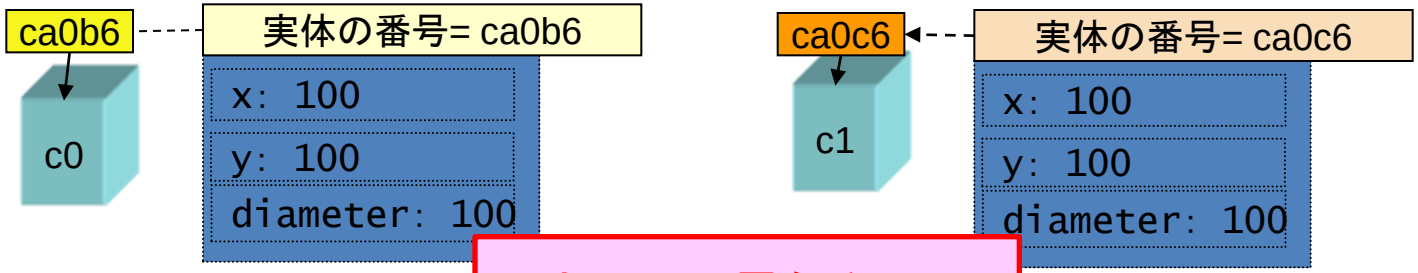


- インスタンスの等価性の評価
- String クラス
- まとめ

第4回資料に加筆

参照型変数への
代入の例

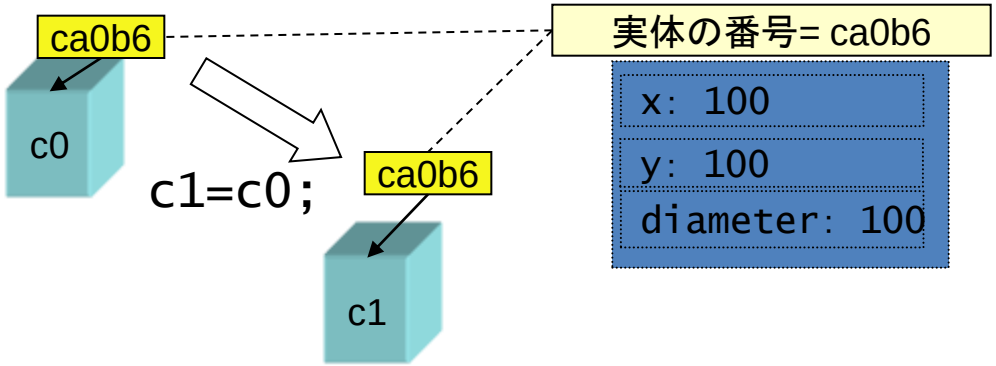
```
(3)
MyCircle c0 = new MyCircle();
MyCircle c1 = new MyCircle();
```



c0 と c1 は、異なるインスタンスが格納されているので、== は不成立

しかし、良く見ると、2つのインスタンスの中身のインスタンス変数は等しい

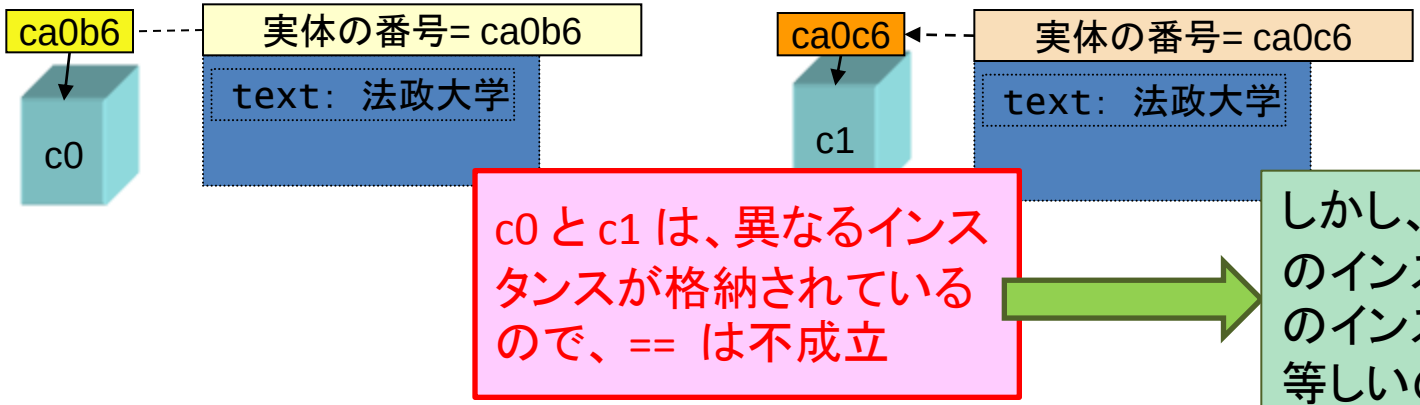
```
(4)
MyCircle c0 = new MyCircle();
MyCircle c1 = c0;
```



c0 と c1 は、全く同じインスタンスを指しているので、== が成立

文字列の比較（String インスタンスの比較）

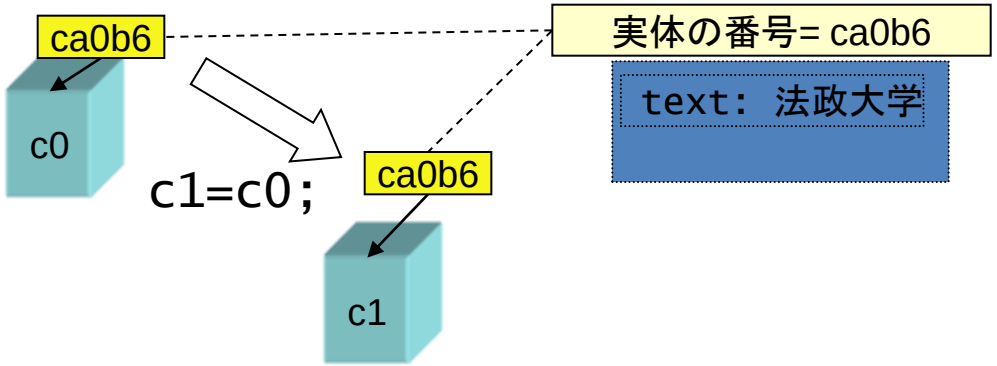
```
(1)
String c0 = ”法政大学”;
String c1 = “法政” + ”大学”;
```



c0 と c1 は、異なるインスタンスが格納されているので、== は不成立

しかし、良く見ると、2つのインスタンスの中身のインスタンス変数は等しいのでequals()は成立

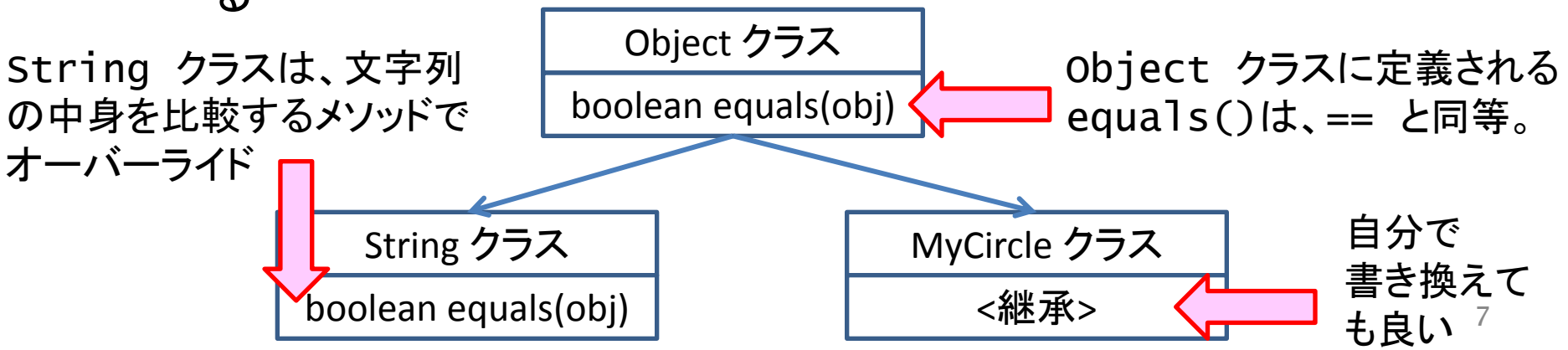
```
(2)
String c0 = ”法政大学”;
String c1 = c0;
```



c0 と c1 は、全く同じインスタンスを指しているので、== が成立

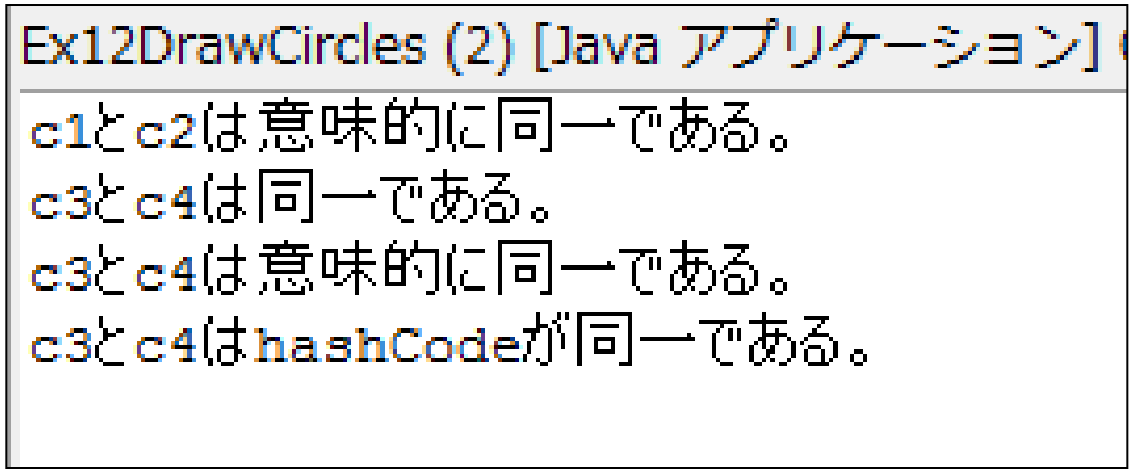
equals() メソッドの意味

- equals メソッドは、インスタンスの実体の同一性を判定しているのではなく、ある意味的な側面での等価性を評価するメソッドである。
  - Object クラスに、最も一般的な形の equals メソッドが定義されていて、それ以外のクラスは、この equals メソッドを継承するか、オーバーライドして定義しなおすかを選択できる。
  - HashMap のキーの等価性は、equals() を使って判断される



■例題13　- 独自のequals() の実装例

問題:MyCircle クラスに、x座標、y座標、直径が等しいインスタンスの時に true を返すような equals メソッドを定義して、動作を確認せよ。



動作確認クラス: Ex12DrawCircles

例題13 MyCircle

```
1  package j2.lesson14;
2
3  import gpjava.Canvas;
4
5  public class MyCircle {
6      int x;
7      int y;
8      int diameter;
9
10     public MyCircle(int x, int y, int d) {..
15
16     public String toString() {..
24
25     public void draw() {..
28
29     // Ex13
30     public boolean equals(Object obj) {
31         if(obj instanceof MyCircle) {
32             MyCircle c = (MyCircle)obj;
33             if(c.x == this.x && c.y == this.y && c.diameter == this.diameter) {
34                 return true;
35             } else {
36                 return false;
37             }
38         } else {
39             return false;
40         }
41     }
42 }
```

equalsメソッドの引数はObject型であることに注意。

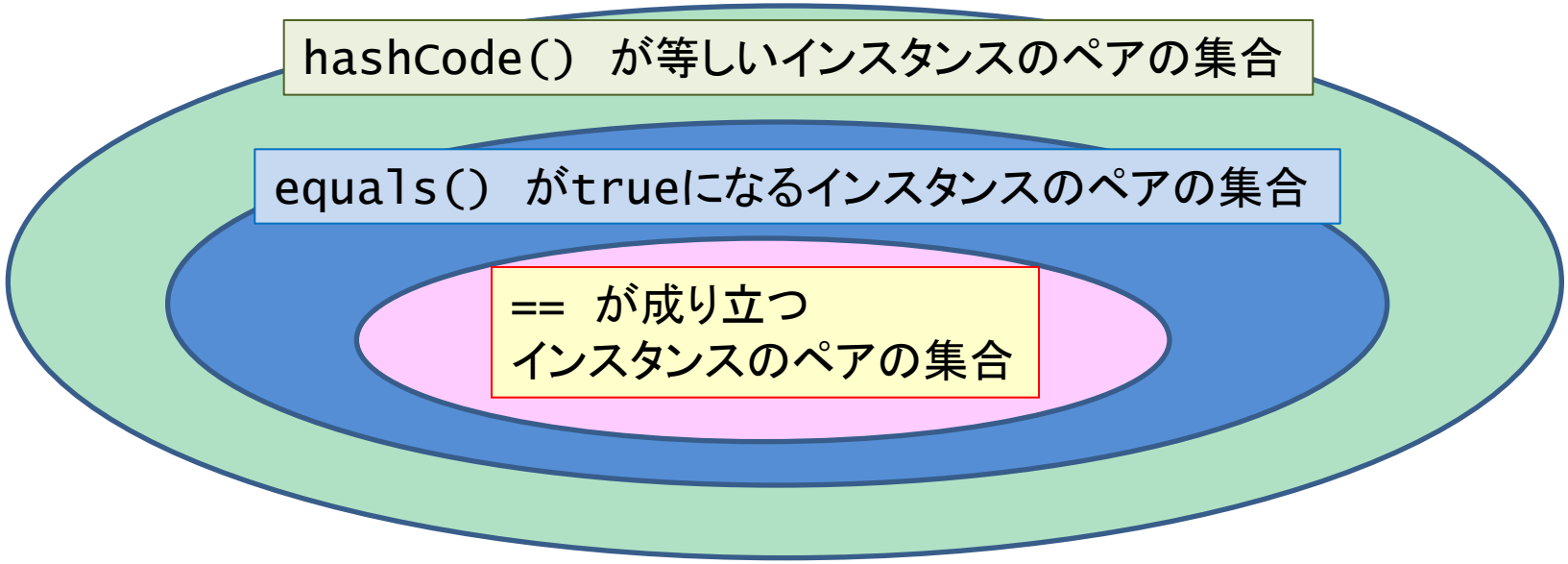
== と equals()の違い

- == は、インスタンスの実体の参照番号を直接比較して、同じ実体を参照している時に true、そうでない時に false を返す。
  - 参照番号の比較だけなので、非常に高速
- equals() は、インスタンスの中身を比較して、意味的な等価性を判断し、 true か false を返す。
  - 例えば、String の場合は文字列の中の全ての文字を一つずつ比較するので、非常に低速

hashCode() メソッド

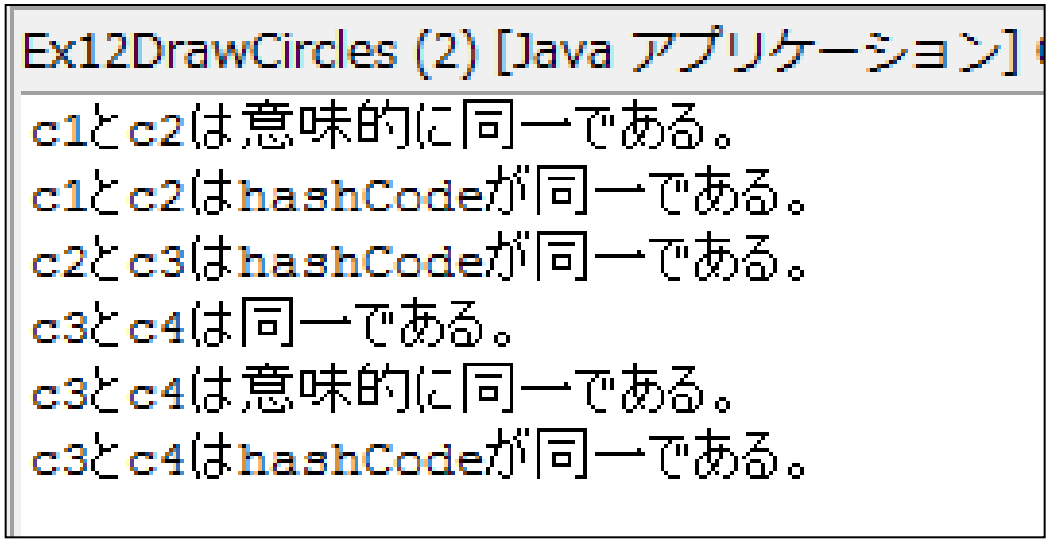
- Java では、あるクラスにequals() メソッドを定義する時には、必ず hashCode() メソッドを定義する必要がある。
  - hashCode() メソッドはインスタンスごとに計算された整数値を返す。
- そして、equals() が true を返す2つのインスタンスは、必ず hashCode()の返す整数が等しくなくてはいけない。

equals() の比較が遅い時には、まずhashCode()を比較して、これが等しい時だけ equals() による詳細な比較を行えば良い！！



■例題14

問題:MyCircle クラスに、x座標、y座標、直径の和を
hashCode() とするメソッドを作成し、動作を確認せよ。



動作確認クラス： Ex12DrawCircles

例題14 MyCircle

```
1  package j2.lesson14;
2
3  import gpjava.Canvas;
4
5  public class MyCircle {
6      int x;
7      int y;
8      int diameter;
9
10     public MyCircle(int x, int y, int d) {..
15
16     public String toString() {..
24
25     public void draw() {..
28
29     // Ex13
30     public boolean equals(Object obj) {
31         if(obj instanceof MyCircle) {
32             MyCircle c = (MyCircle)obj;
33             if(c.x == this.x && c.y == this.y && c.diameter == this.diameter) {
34                 return true;
35             } else {
36                 return false;
37             }
38         } else {
39             return false;
40         }
41     }
42
43     // Ex14
44     public int hashCode() {
45         return x + y + diameter;
46     }
47 }
```

簡単かつ、equals() が成立する時に同じ値になる関数であることが条件

テーマ: 等価性と Stringクラス

- インスタンスの等価性の評価
- String クラス
- まとめ

String クラス

- Java は、文字列を表現するために String クラスのインスタンスを用いる。文字列は、オブジェクトである。
- String s = “法政”; と実行する時に、内部では String s = new String(“法政”); のようなコードが実行されていると考えて良い。
- String クラスには、文字列を処理するための有用なインスタンスメソッドが多数用意されている。

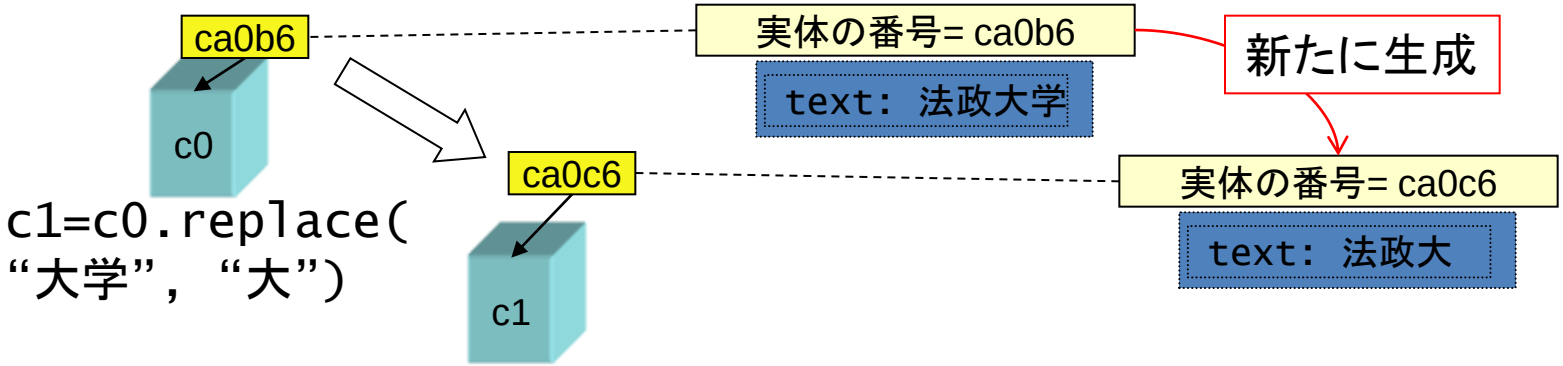
String のインスタンスメソッド

下記に、有用なメソッドの一部を示す。詳細な仕様を知りたい人は、下記を調べること
http://java.sun.com/javase/ja/6/docs/ja/api/java/lang/String.html

返り値の型	メソッド名(引数)	機能
char	charAt(int index)	指定されたインデックス位置にある char 値を返す
boolean	equals(Object obj)	文字列の中身の比較を行う
int	indexOf(int ch)	指定された文字が最初に出現する位置を返す。文字が存在しない時は-1を返す
int	indexOf(String str)	指定された文字列が最初に出現する位置を返す。文字列が存在しない時は-1を返す
int	length()	文字列の長さを返す
String	replace(char oldC, char newC)	文字列中の全ての文字 oldC をnewC に置き換えた「新しい」文字列を返す。
String	replace(String oldS, String newS)	文字列中の全ての文字列 oldS をnewS に置き換えた「新しい」文字列を返す。
String[]	split（String s)	文字列を指定された文字列 s で分割して配列にして返す
String	substring(int begin, int end)	文字列の中から、begin 番目から end-1 番目までの位置にある文字列を切り出す。
String	substring(int begin)	文字列の中から、begin番目以降の文字列を切り出す。
char[]	toCharArray()	文字列を文字の配列に変換して返す

String クラスのメソッドの注意

- replace(), substring() などを呼び出しても、元の String インスタンスは変更されない。
  - 結果の文字列を、新しい String インスタンスとして生成して返却する。
  - String インスタンスを replace したり、文字列を結合したりすると、新たなString インスタンスが別に生成されるので、大きな文字列の操作は、コンピュータのメモリを予想以上に浪費することになるので注意が必要。
  - 興味ある人は、StringBuilder クラスも調査すること。



テーマ: 等価性と Stringクラス

- インスタンスの等価性の評価
- String クラス
- まとめ

「クラス定義」の考え方

考え方	特徴	提供される機能
複合データ構造	複数の異なる型の変数をまとめて扱う単位	インスタンス変数 継承
「もの」としての抽象化	内部の実装を隠ぺいして、「もの」と「もの」の間のメッセージのやり取りとしてプログラミングする単位	型 抽象クラス インタフェース インスタンスメソッド 継承 ポリモーフィズム

第12回資料を基に作成

オブジェクト指向

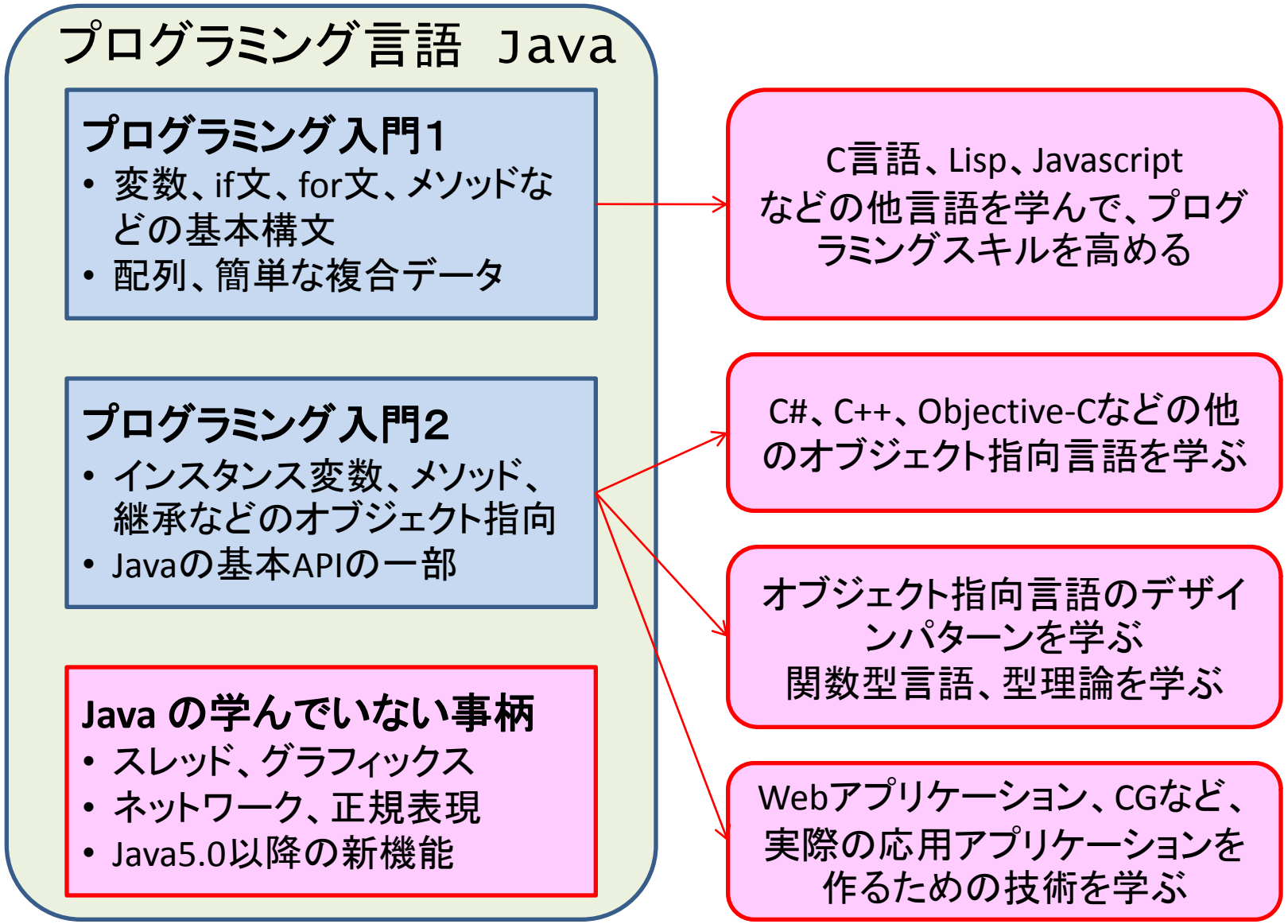
- Java言語は、インスタンス（オブジェクト）を中心にしてプログラミングを行う仕組みを備えており、このような仕組みを備えた言語は「**オブジェクト指向言語**」と呼ばれる。

Javaは、インスタンス（オブジェクト）に関して下記の重要な機構（メカニズム）を有する。

1. インスタンスを生成するための機構
   ＝「クラスからインスタンスが生成される」
2. インスタンスが主体となって、計算を進める機構。（メッセージパッシングによって計算が進む。）
   ＝「インスタンスがメソッドを持つ」

3. 多様なインスタンスを効率的に生成したり、整理するための機構
   ＝「継承によって種類の異なるインスタンスを効率的に作成する。」
   ＝「継承によってクラス階層構造が構築される」

入門1、2で学んだこと


プログラミング言語 Java
プログラミング入門1
• 変数、if文、for文、メソッドなどの基本構文
• 配列、簡単な複合データ
プログラミング入門2
• インスタンス変数、メソッド、継承などのオブジェクト指向
• Javaの基本APIの一部
Java の学んでいない事柄
• スレッド、グラフィックス
• ネットワーク、正規表現
• Java5.0以降の新機能
C言語、Lisp、Javascript
などの他言語を学んで、プログラミングスキルを高める
C#、C++、Objective-Cなどの他のオブジェクト指向言語を学ぶ
オブジェクト指向言語のデザインパターンを学ぶ
関数型言語、型理論を学ぶ
Webアプリケーション、CGなど、
実際の応用アプリケーションを作るための技術を学ぶ

まとめ : 等価性と Stringクラス

- インスタンスの等価性の評価
  - ==, equals(), hashCode()のそれぞれの等価性のレベルについて理解した
- String クラス
  - String はインスタンスである
  - 様々な String APIは、返還後のStringを新しい String インスタンスとして返却する
- まとめ
  - 今後に学ぶべき内容の確認

本日の例題と問題

- 等価性の判定
  - Ex11, Ex12, Ex13, Ex14
- 文字列の操作
  - Ex21, Ex22, Q11, Q12
- ファイルの変換
  - Ex31, Q21, Q22
- スプレッドシートの作成
  - Ex41, Ex42*, Q31(1)*, (Q31(2)*)

(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。

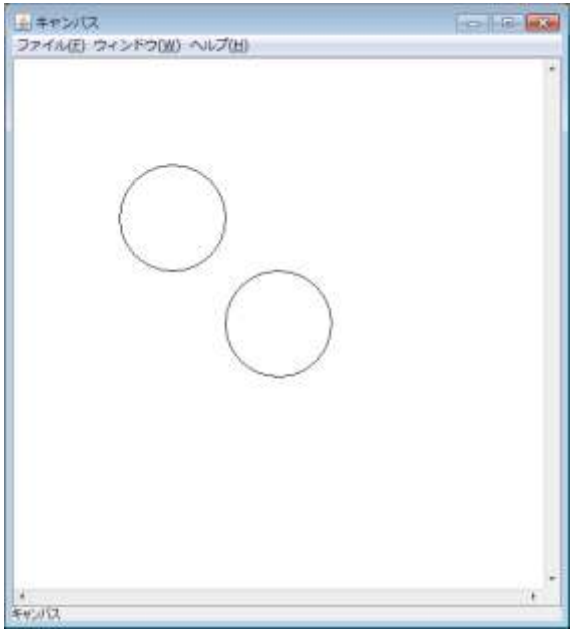
例題集

パッケージ「j2.lesson14」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11

問題: x座標、y座標、直径をインスタンス変数に持つMyCircle クラスを作成せよ。MyCircle クラスには、コンストラクタ、toString メソッド、draw メソッドを定義せよ。Ex11DrawCirclesに示すように、MyCircle 型の変数4個を用意し、Canvas に表示せよ。




```
<終了> Ex11DrawCircle [Java アプリケーション]
[x:100 y:100 diameter:100]
[x:200 y:200 diameter:100]
[x:100 y:100 diameter:100]
[x:200 y:200 diameter:100]
```

動作確認クラス： Ex11DrawCircles

例題11続き

MyCircle

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

コンストラクタ(引数)	機能
MyCircle(int x, int y, int d)	インスタンス変数の値がx, y, d であるMyCircleクラスのインスタンスを作成する。

インスタンスメソッド

返り値の型	メソッド名(引数)	機能
String	toString()	[x:… y:… diameter:…] というインスタンス変数の中身を埋め込んだ文字列を返却
void	draw()	Canvas 上に円を描く

(クラス名： MyCircle)

動作確認クラス： Ex11DrawCircles

例題11 Ex11DrawCircles

```
 1  package j2.lesson14;
 2
 3  import gpjava.Canvas;
 4
 5  public class Ex11DrawCircles {
 6      public static void main(String[] args) {
 7          MyCircle c1 = new MyCircle(100, 100, 100);
 8          MyCircle c2 = new MyCircle(200, 200, 100);
 9          MyCircle c3 = new MyCircle(100, 100, 100);
10          MyCircle c4 = c2;
11
12          Canvas.show();
13          c1.draw();
14          c2.draw();
15          c3.draw();
16          c4.draw();
17
18          System.out.println(c1);
19          System.out.println(c2);
20          System.out.println(c3);
21          System.out.println(c4);
22      }
23  }
```

例題11 MyCircle

```java
package j2.lesson14;

import gpjava.Canvas;

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

    public MyCircle(int x, int y, int d) {
        this.x = x;
        this.y = y;
        this.diameter = d;
    }

    public String toString() {
        String result = "[";
        result += "x:" + this.x;
        result += " y:" + this.y;
        result += " diameter:" + this.diameter;
        result += "]";
        return result;
    }

    public void draw() {
        Canvas.drawOval(this.x, this.y, this.diameter, this.diameter);
    }
}
```

■例題12

問題:例題11で作成した4個のMyCircle型変数について、等価性を ==, equals(), hashCode() の比較という3個の手法を用いて判断せよ。

Ex12DrawCircle のクラスメソッド

返り値の型	メソッド名(引数)	機能
void	compare(MyCircle c1, MyCircle c2, String c1Name, String c2Name,)	MyCircle c1 と c2 の等価性を ==, equals(), hashCode() の比較で判断し、結果をコンソールに表示する

```
Ex12DrawCircles (2) [Java アプリケーション]
c3とc4は同一である。
c3とc4は意味的に同一である。
c3とc4はhashCodeが同一である。
```

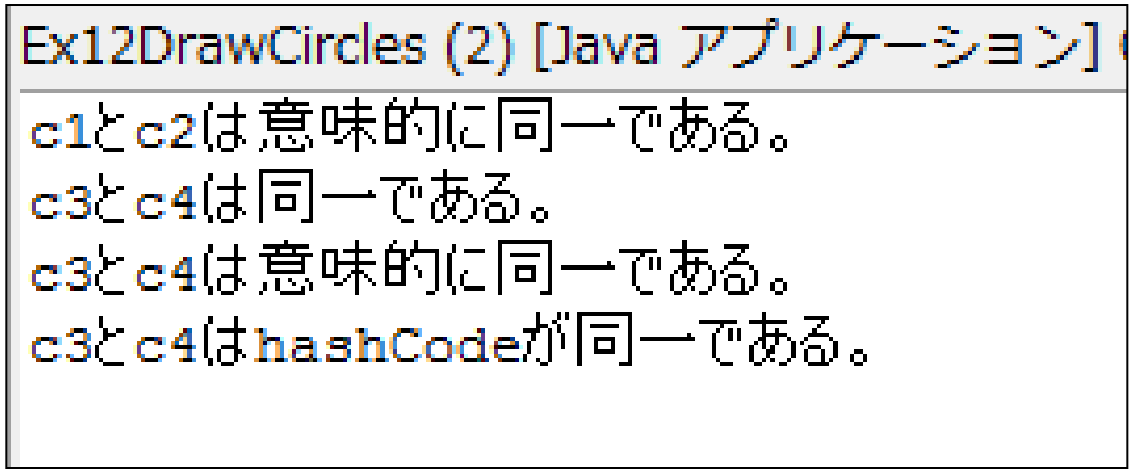
動作確認クラス: Ex12DrawCircles

例題12 Ex12DrawCircles

```java
package j2.lesson14;

import gpjava.Canvas;

public class Ex12DrawCircles {
    public static void main(String[] args) {
        MyCircle c1 = new MyCircle(200, 100, 100);
        MyCircle c2 = new MyCircle(200, 100, 100);
        MyCircle c3 = new MyCircle(100, 200, 100);
        MyCircle c4 = c3;

        Canvas.show();
        c1.draw();
        c2.draw();
        c3.draw();
        c4.draw();

        compare(c1, c2, "c1", "c2");
        compare(c2, c3, "c2", "c3");
        compare(c3, c4, "c3", "c4");
    }

    public static void compare(MyCircle c1, MyCircle c2, String c1Name, String c2Name) {
        if(c1 == c2) {
            System.out.println(c1Name + "と" + c2Name + "は同一である。");
        }

        if(c1.equals(c2)) {
            System.out.println(c1Name + "と" + c2Name + "は意味的に同一である。");
        }

        if(c1.hashCode() == c2.hashCode()) {
            System.out.println(c1Name + "と" + c2Name + "はhashCodeが同一である。");
        }
    }
}
```

■例題13

問題:MyCircle クラスに、 x座標、y座標、直径が等しいインスタンスの時に true を返すような equals メソッドを定義して、動作を確認せよ。



動作確認クラス: Ex12DrawCircles

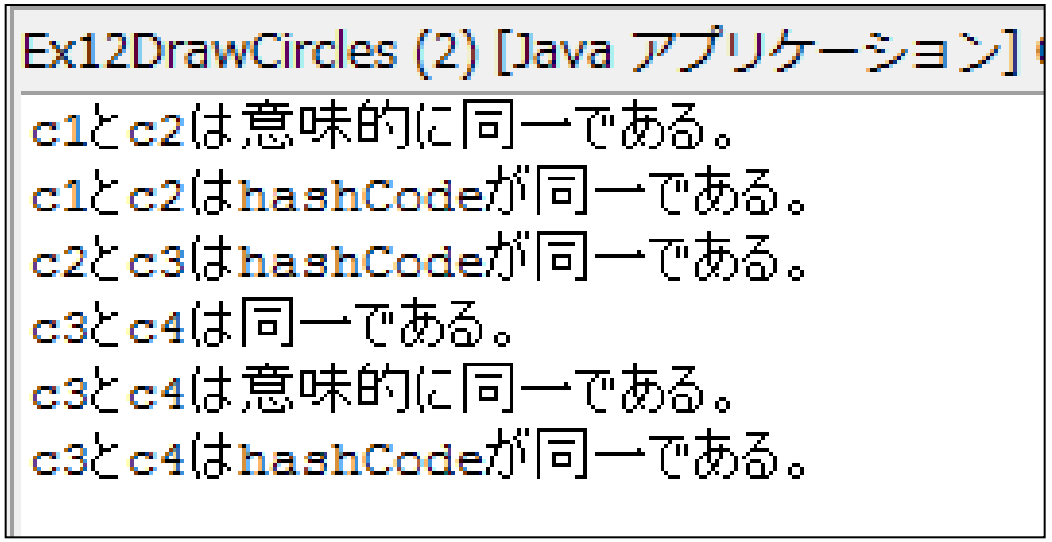
例題13 MyCircle

```
1  package j2.lesson14;
2
3  import gpjava.Canvas;
4
5  public class MyCircle {
6      int x;
7      int y;
8      int diameter;
9
10     public MyCircle(int x, int y, int d) {..
15
16     public String toString() {..
24
25     public void draw() {..
28
29     // Ex13
30     public boolean equals(Object obj) {
31         if(obj instanceof MyCircle) {
32             MyCircle c = (MyCircle)obj;
33             if(c.x == this.x && c.y == this.y && c.diameter == this.diameter) {
34                 return true;
35             } else {
36                 return false;
37             }
38         } else {
39             return false;
40         }
41     }
42 }
```

equals()の引数が Object であることに注意。
31行目で、MyCircle のインスタンス化どうかを判定。

■例題14

問題:MyCircle クラスに、x座標、y座標、直径の和を hashCode() として計算するメソッドを作成し、動作を確認せよ。



動作確認クラス： Ex12DrawCircles

例題14 MyCircle

```
1  package j2.lesson14;
2
3  import gpjava.Canvas;
4
5  public class MyCircle {
6      int x;
7      int y;
8      int diameter;
9
10     public MyCircle(int x, int y, int d) {..
15
16     public String toString() {..
24
25     public void draw() {..
28
29     // Ex13
30     public boolean equals(Object obj) {
31         if(obj instanceof MyCircle) {
32             MyCircle c = (MyCircle)obj;
33             if(c.x == this.x && c.y == this.y && c.diameter == this.diameter) {
34                 return true;
35             } else {
36                 return false;
37             }
38         } else {
39             return false;
40         }
41     }
42
43     // Ex14
44     public int hashCode() {
45         return x + y + diameter;
46     }
47 }
```

■例題21

問題:与えられた文字列を一文字ずつに分解してコンソールに表示する devideToChar(String text)、与えられた文字列の中に、ある特定文字が何回現れるかを数える countChar(String text, char c)を作成し、動作を確認せよ。また、String の replace() メソッドの動作を確認せよ。

クラスメソッド

返り値の型	メソッド名(引数)	機能
void	devideToChar(String text)	文字列 text を一文字ずつに分解してコンソールに表示する。
int	countChar(String text, char c)	文字列 text の中に char c が何回現れるかをカウントして値を返す。

動作確認クラス： Ex21String

例題21
Ex21String

文字の置き換え

文字列の置き換え

```
1  package j2.lesson14;
2
3  public class Ex21String {
4      public static void main(String[] args) {
5          String text = "私は、長野県長野市に住んでいます。";
6          devideToChar(text);
7
8          System.out.println(countChar(text, '私'));
9          System.out.println(countChar(text, '長'));
10
11         String result;
12
13         result = text.replace('野', '崎');
14         System.out.println(result);
15
16         result = text.replace("長野市", "松本市");
17         System.out.println(result);
18     }
19
20     static void devideToChar(String str) {
21         for(int i = 0; i < str.length(); i++) {
22             char c = str.charAt(i);
23             System.out.println(c);
24         }
25     }
26
27     static int countChar(String str, char c) {
28         int count = 0;
29         for(int i = 0; i < str.length(); i++) {
30             if(c == str.charAt(i)) {
31                 count++;
32             }
33         }
34         return count;
35     }
36 }
```

■例題22

問題:文字列の中から、「、」を削除する
removeComma(String text) メソッドを作成せよ。また、これを汎用化して、任意の文字を削除する
removeChar(String text, char c)を作成せよ。

クラスメソッド

返り値の型	メソッド名(引数)	機能
String	removeComma(String text)	文字列 text から、「、」を削除した文字列を返却する。
String	removeChar(String text, char c)	文字列 text から char c を削除した文字列を返却する。

動作確認クラス： Ex22String

例題22
Ex22String

charAt() を用いて
一文字ずつ調べる方法

indexOf() を用いて
検索する方法

```java
package j2.lesson14;

public class Ex22String {
    public static void main(String[] args) {
        String text = "私は、長野県長野市に住んでいます。";

        String result = removeComma(text);
        System.out.println(result);

        result = removeChar(text, '、');
        System.out.println(result);
    }

    static String removeComma(String text) {
        int start = 0;
        String result = "";
        for(int i = 0; i < text.length(); i++) {
            char c = text.charAt(i);
            if(c == '、') {
                result += text.substring(start, i);
                start = i + 1;
            }
        }
        result += text.substring(start);
        return result;
    }

    static String removeChar(String text, char c) {
        int start = 0;
        String result = "";
        int index = -1;

        while((index = text.indexOf(c, start)) != -1) {
            result += text.substring(start, index);
            start = index + 1;
        }
        result += text.substring(start);
        return result;
    }
}
```

■例題31

問題:inputFiles/12ProductData.txt を読み込んで、”,” を “ “(スペース)に置き換えたファイルを、inputFiles/14Result.txt に書きだすプログラムを作成せよ。

クラスメソッド

返り値の型	メソッド名(引数)	機能
void	readAndWriteFile(String readFileName, String writeFileName)	readFileNameの示すファイルから、文章を読み込み、”,”を” “(スペース)に置き換えて、writeFileNameに保存する。

動作確認クラス： Ex31CSV

例題31 Ex31CSV

```java
package j2.lesson14;

import java.io.*;

public class Ex31CSV {
    public static void main(String[] args) {
        readAndWriteFile("inputFiles/12ProductData.txt", "inputFiles/14Result.txt");
    }

    public static void readAndWriteFile(String readFileName, String writeFileName) {
        try {
            BufferedReader reader = new BufferedReader(new FileReader(readFileName));
            PrintWriter writer = new PrintWriter(new FileWriter(writeFileName));
            String line;

            while ((line = reader.readLine()) != null) {
                String result = line.replace(",", " ");
                writer.println(result);
            }
            reader.close();
            writer.close();
        } catch (IOException ex) {
            ex.printStackTrace();
            System.out.println("ファイルの読み書きに失敗しました。");
        }
    }
}
```

■例題41

問題:Spreadsheet のセルをクリックして、そこに文字列を代入できるようにせよ。入力文字列は配列にしてCell0 クラスのクラス変数に管理せよ。また、既に文字列が入力されているセルをクリックした時には、既存の文字列を入力ダイアログの初期値に設定せよ。

（クラス名： Cell0）
動作確認クラス： Ex41Spreadsheet

例題41続き

インスタンス変数	初期値	説明
String content	無し	セルの内容の文字列

クラス変数	初期値	説明
Cell0[][] table	new Cell0[5][5]	セルを管理する2次元配列

コンストラクタ(引数)	機能
Cell0(String content)	文字列 content をインスタンス変数に持つインスタンスを生成する。

クラスメソッド

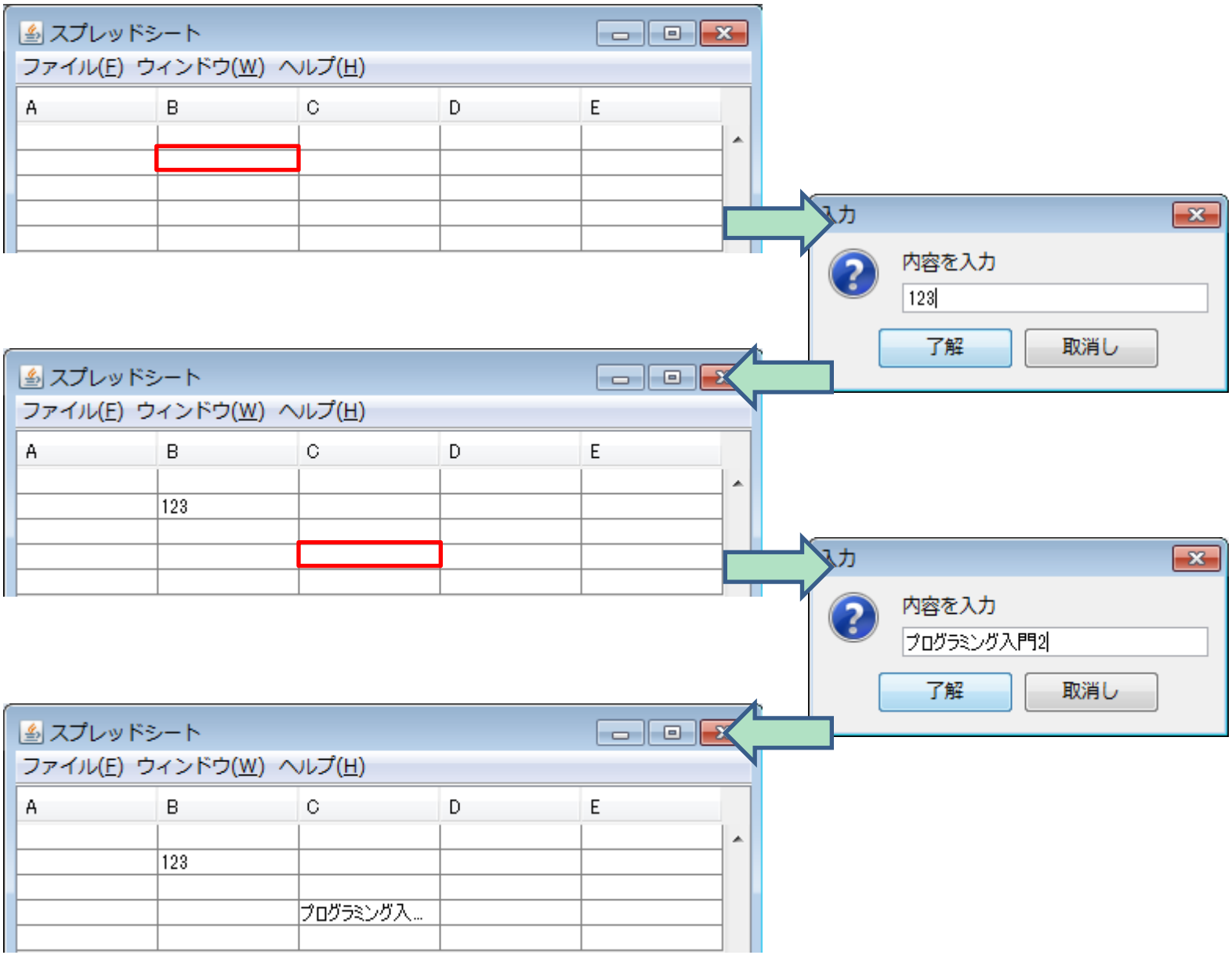
返り値の型	メソッド名(引数)	機能
void	set(int row, int column, String content)	table[row][column] に、content 文字列を持つ Cell0 インスタンスを設定する。
Cell0	get(int row, int column)	table[row][column]の値を返却する。
void	calcAll()	全てのセルの値を再計算して表示する。

インスタンスメソッド

返り値の型	メソッド名(引数)	機能
void	setContent(String content)	文字列 content を新しい値に置き換える

クラス名： Cell10

例題41　実行例


スプレッドシート
ファイル(F)　ウィンドウ(W)　ヘルプ(H)
A
B
C
D
E
入力
内容を入力
123
了解
取消し
スプレッドシート
ファイル(F)　ウィンドウ(W)　ヘルプ(H)
A
B
C
D
E
123
入力
内容を入力
プログラミング入門2
了解
取消し
スプレッドシート
ファイル(F)　ウィンドウ(W)　ヘルプ(H)
A
B
C
D
E
123
プログラミング入...

例題41 Cell0

```java
package j2.lesson14;

import gpjava.Spreadsheet;

public class Cell0 {
    static Cell0[][] table = new Cell0[5][5];
    String content;

    Cell0(String content) {
        setContent(content);
    }

    static void set(int row, int column, String content) {
        Cell0 cell = table[row][column];
        if(cell == null) {
            cell = new Cell0(content);
            table[row][column] = cell;
        } else {
            cell.setContent(content);
        }
    }

    static Cell0 get(int row, int column) {
        return table[row][column];
    }

    static void calcAll() {
        for(int i = 0; i < table.length; i++) {
            for(int j = 0; j < table[i].length; j++) {
                Cell0 cell = table[i][j];
                if(cell != null) {
                    Spreadsheet.setString(i, j, cell.content);
                }
            }
        }
    }

    void setContent(String content) {
        if(content == null) {
            content = "";
        }
        this.content = content;
    }
}
```

例題41 Ex41Spreadsheet

```java
package j2.lesson14;

import javax.swing.JOptionPane;
import gpjava.Spreadsheet;

public class Ex41Spreadsheet {
    public static void main(String[] args) {
        Spreadsheet.show(5, 5);

        while(true) {
            Spreadsheet.waitForSelect("セルを選択");
            int column = Spreadsheet.getSelectedColumn();
            int row = Spreadsheet.getSelectedRow();

            Cell0 cell = Cell0.get(row, column);
            String content = "";
            if(cell != null) {
                content = cell.content;
            }
            content = JOptionPane.showInputDialog("内容を入力", content);
            Cell0.set(row, column, content);
            Cell0.calcAll();
        }
    }
}
```

セルの選択方法

初期値

■例題42

問題:「=A2」という書き方で、A2欄(2行1列目)の値を参照して表示できるように例題41のCell0 を改造し、Cell1とせよ。

注意
Ex42Spreadsheet は、Ex41Spreadsheet の Cell0 を Cell1に変更するだけ

(クラス名: Cell1)
動作確認クラス: Ex42Spreadsheet

例題42　実行例

例題42
Cell1
その1

```java
package j2.lesson14;

import gpjava.Spreadsheet;

public class Cell1 {
    static Cell1[][] table = new Cell1[5][5];
    String content;
    boolean processed;
    String stringValue;

    Cell1(String content) {
        setContent(content);
    }

    static void set(int row, int column, String content) {
        Cell1 cell = table[row][column];
        if(cell == null) {
            cell = new Cell1(content);
            table[row][column] = cell;
        } else {
            cell.setContent(content);
        }
    }

    static Cell1 get(int row, int column) {
        return table[row][column];
    }


    static void resetAll() {
        for(int i = 0; i < table.length; i++) {
            for(int j = 0; j < table[i].length; j++) {
                Cell1 cell = table[i][j];
                if(cell != null) {
                    cell.processed = false;
                }
            }
        }
    }

}
```

評価済みセルかどうかの検査用

評価後の文字列を格納

全てのセルを評価前の状態にする

例題42 Cell1 その2

```
41      static void calcAll() {
42          resetAll();
43          boolean cont = true; // 繰り返し実行するかを判断するための変数
44          while(cont) {
45              cont = false;
46              for(int i = 0; i < table.length; i++) {
47                  for(int j = 0; j < table[i].length; j++) {
48                      Cell1 cell = table[i][j];
49                      if(cell != null) {
50                          if(cell.processed == true) {
51                              // 処理済みのセルは何もしない
52                          } else {
53                              if(cell.content.length() > 0 && cell.content.charAt(0) == '=') {
54                                  if(cell.eval() == true) {
55                                      Spreadsheet.setString(i, j, cell.stringValue);
56                                      cell.processed = true;
57                                  }
58                              } else {
59                                  cell.stringValue = cell.content;
60                                  Spreadsheet.setString(i, j, cell.stringValue);
61                                  cell.processed = true;
62                              }
63                              cont = true;
64                          }
65                      }
66                  }
67              }
68          }
69      }
70
```

全てが評価済になるまで何度も繰り返す

= で始まる文字列を見つけたら、eval を呼ぶ

普通の文字列は、そのまま評価済み文字列に設定する。

例題42 Cell1 その3

```
71      void setContent(String content) {
72          if(content == null) {
73              content = "";
74          }
75          this.content = content;
76      }
77
78      boolean eval() {
79          int column = content.charAt(1) - 'A'; // 文字'A'の時0
80          int row = content.charAt(2) - '1'; // 文字'1'の時0
81          Cell1 target = Cell1.get(row, column);
82          if(target == null) {
83              stringValue = "";
84              return true;
85          } else {
86              if(target.processed == true) {
87                  stringValue = target.stringValue;
88                  return true;
89              }
90              return false;
91          }
92      }
93  }
```

文字列から、行・列の数値を計算する

参照先が、評価済みだったら、値をコピーする