

# プログラム設計

<http://bit.ly/design4d>

## 継承 (1)


前期 第9週

2019/6/19

【Point 3】スーパークラスの**private**メンバは、サブクラスからでもアクセスできない。（privateは、サブクラスも「外部のクラス」として扱う。）**protected**メンバは、サブクラスからはアクセスできるようになる（その他のクラスは「外部のクラス」となる。）

(p.351～355)


```
1: class Car {  
2:     protected int num;  
3:     private double gas;  
4:  
5:     public Car() {  
6:         num = 0; gas = 0.0;  
7:     }  
8:     public void setNum(int n) {  
9:         num = n;  
10:    }  
11:    public int getNum() {  
12:        return num;  
13:    }
```



```
14:      public void setGas(double g) {
15:          if(g >0  && g < 1000 ) {
16:              gas = g;
17:          } else {
18:              System.out.println("setGas: Out of
range." );
19:          }
20:      }
21:      public double getGas() {
22:          return gas;
23:      }
24:      public void show() {
25:          System.out.println("show: (num) "
+ num + " (gas) " + gas);
26:      }
27:  }
28:
```

【Point 1】 「**extends**」 の後続くクラスを継承する。継承は、**クラスを拡張する**ために利用する。この場合、クラスCarを継承して、クラスRacingCarを定義している。(p.342)

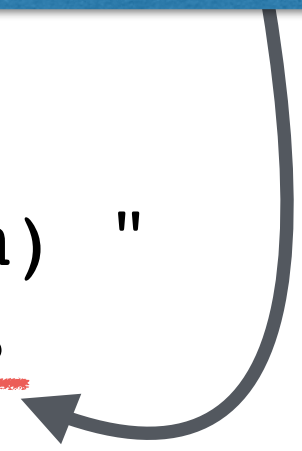
- Car → **スーパークラス** (親クラス)
- RacingCar → **サブクラス** (子クラス)
- 追加されるフィールド course
- 追加されるメソッド getCourse, setCourse, show



```
29: class RacingCar extends Car {
30:     private int course;
31:     public int getCourse() {
32:         return course;
33:     }
34:     public void setCourse(int c) {
35:         course = c;
36:     }
```

【Point 4】 スーパークラスのprotectedメンバ（ここではnum）にはサブクラスからはアクセスできるが、privateメンバ（ここではgas）にはアクセスできない。（エラーとなるためコメントにしている。） (p.351～355)

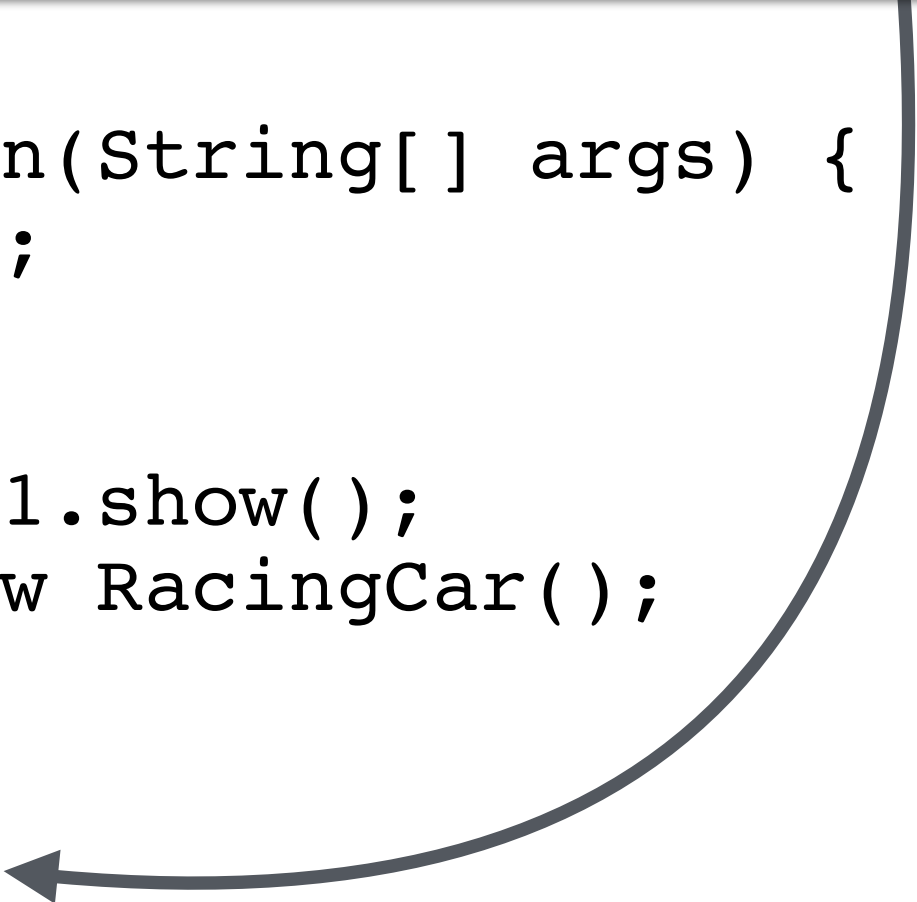
```
37:      public void show2() {
38:          System.out.println("show2: (num) "
                               + num // + " (gas) " + gas
39:                               + " (course) " + course);
40:      }
41: }
42:
```



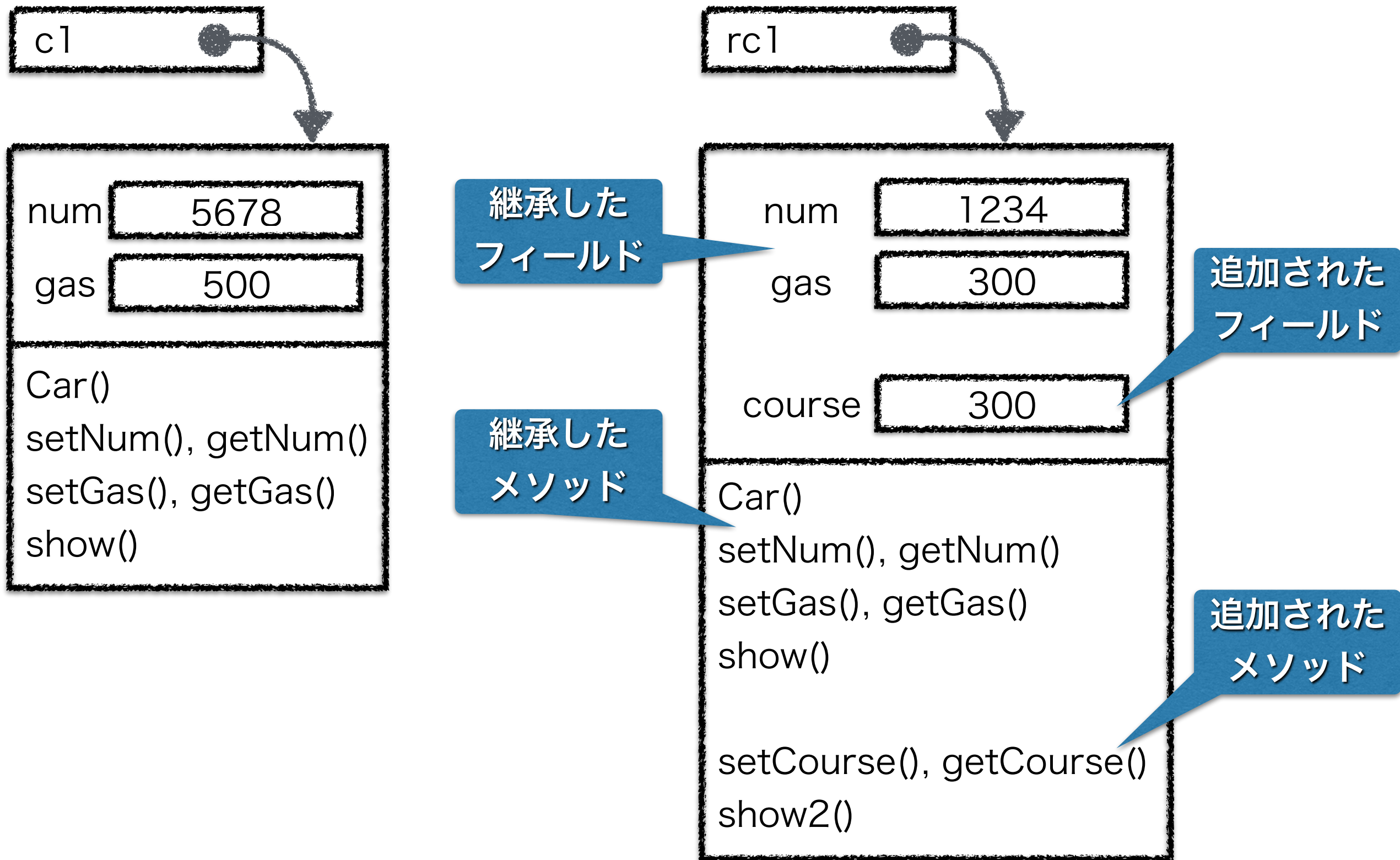


【Point 2】 継承したクラスは、**スーパークラスのメソッド**を呼び出すことができ（ここでは setNum, setGas, showを呼び出している）、さらに、**追加したメソッド**も呼び出せる。（ここではsetCourse, show2を呼び出している）（p.344～346, 図11-3）

```
43: class Pd9car1 {
44:     public static void main(String[] args) {
45:         Car c1 = new Car();
46:         c1.setGas(500);
47:         c1.setGas(-100);
48:         c1.setNum(5678); c1.show();
49:         RacingCar rc1 = new RacingCar();
50:         rc1.setCourse(7);
51:         rc1.setNum(1234);
52:         rc1.setGas(-200);
53:         rc1.setGas(300);
54:         rc1.show();
55:         rc1.show2();
56:     }
57: }
```



# インスタンスの様子



# 【課題の準備】

演習室で作業する前に、以下のコマンドを  
入れるだけで準備が完了する

```
$ mygitclone4d 「自分のGitHubユーザ名」  
$ cd prog4d-(ユーザ名)  
$ ./myconf
```

※本体をシャットダウンするまでは、  
上記「mygitclone」と「myconf」の設定は有効です



# 【課題の準備】

以下の流れで、課題のプログラムを作るためのフォルダを準備しましょう。

1. 端末を起動して、以下のコマンドを実行して**前期第9週のフォルダ**を作る  
\$ cd prog4d-(**ユーザ名**)      (←既に移動しているなら不要)  
\$ mkdir **week109**  
\$ cd **week109**

※課題で作るファイル名は各自で決めて構いません。

# 【練習9-1】

サンプルプログラム「1\_09\_Car.java」を  
コンパイルして、実行結果を確認しましょう。

# 【課題9-1】

サンプルプログラムの38行目のコメントを外してコンパイルし、**どのようなエラーが出力されるのか**確認してください。

そして、このエラーの原因となっているフィールド `gas` を **protected** となるようにプログラムを修正し、動作を確認してください。

# 【課題の提出】

以下の流れで、作ったCプログラムをGitHubにプッシュして、Webサイトで確認してみましょう。

1. 端末内で、以下のコマンドで課題を提出

```
$ git add -A
```

```
$ git commit -m “課題9-1提出”
```

```
$ git push origin master
```

2. 自分のリポジトリを開いて、提出したファイルがプッシュされているか確認する

[https://github.com/nit-ibaraki-prog4d-2019/prog4d-\(ユーザー名\)](https://github.com/nit-ibaraki-prog4d-2019/prog4d-(ユーザー名))



# 【課題9-2】

次のような整数を扱うクラスBaseを考えます。  
(「1\_09\_Base.java」から入手可能)

```
class Base {  
    protected int num;  
    public void setNum(int n) {  
        num = n;  
    }  
    public int getNum() {  
        return num;  
    }  
    public void showNum() {  
        System.out.printf("num: %d\n", num);  
    }  
}
```

(課題は次のスライドに続きます)

# 【課題9-2】

クラスBaseを継承して、次のようなメソッドが追加されたクラスEvenを作成して、動作を確認してください。

```
public void setEven(int n)
```

```
//引数nが偶数の場合、nをフィールドnumに代入する
```

(mainを持つクラスとその実行結果は、  
課題9-3とまとめてあります。)

# 【課題9-3】

クラスBaseを継承して、次のようなメソッドが追加されたクラスOddを作成して、動作を確認してください。

```
public void setOdd(int n)
```

```
//引数nが奇数の場合、nをフィールドnumに代入する
```

# 【課題9-3】

このクラスはファイル「1\_09\_Main.java」に含まれている

```
class Pd9Base {
    public static void main(String[] args) {
        Base b1 = new Base();
        //Baseはフィールドnumに自由に値を代入できる
        b1.setNum(12); b1.showNum();
        b1.setNum(5); b1.showNum();

        Even e1 = new Even();
        //Evenはnumに偶数のみ代入できる
        e1.setEven(12); e1.showNum();
        e1.setEven(5); e1.showNum();

        Odd o1 = new Odd();
        //Oddはnumに偶数のみ代入できる
        o1.setOdd(12); o1.showNum();
        o1.setOdd(5); o1.showNum();
    }
}
```



# 【課題9-3】

## [ 実行結果 ]

num: 12	( ← b1に対する処理 )
num: 5	
setEven: 値を代入します	( ← e1に対する処理 )
num: 12	
setEven: 値を代入しません	
num: 12	
setOdd: 値を代入しません	( ← o1に対する処理 )
num: 0	
setOdd: 値を代入します	
num: 5	

# 【課題の提出】

以下の流れで、作ったCプログラムをGitHubにプッシュして、Webサイトで確認してみましょう。

1. 端末内で、以下のコマンドで課題を提出

```
$ git add -A
```

```
$ git commit -m "課題9-3提出"
```

```
$ git push origin master
```

2. 自分のリポジトリを開いて、提出したファイルがプッシュされているか確認する

[https://github.com/nit-ibaraki-prog4d-2019/prog4d-\(ユーザー名\)](https://github.com/nit-ibaraki-prog4d-2019/prog4d-(ユーザー名))

# 【課題9-4】

次のような「辺の長さ」を表すクラスLengthを考えます。（「1\_09\_Length.java」から入手可能）

```
class Length {  
    protected int height; //高さ  
    protected int width;  //幅  
    public void setHeight(int h) {  
        height = h;  
    }  
    public int getHeight() {  
        return height;  
    }  
    public void setWidth(int w) {  
        width = w;  
    }  
    public int getWidth() {  
        return width;  
    }  
}
```

（課題は次のスライドに続きます）

# 【課題9-4】

このクラスを継承して、「**四角形**」を表すクラスRectangle、「**三角形**」を表すクラスTriangleの**2つのサブクラス**を作成してください。

この2つのサブクラスは、次のような「面積を求めて表示する」メソッドareaを**それぞれ**持ちます。

```
public void area()
```

```
//フィールドheightとwidthを使って面積を計算して表示する
```



# 【課題9-4】

このクラスはファイル「1\_09\_Main.java」に含まれている

```
class Pd9Length {  
    public static void main(String[] args) {  
        //四角形のインスタンスで面積を表示する  
        Rectangle r1 = new Rectangle();  
        r1.setHeight(10);  
        r1.setWidth(20);  
        r1.area();  
        //三角形のインスタンスで面積を表示する  
        Triangle t1 = new Triangle();  
        t1.setHeight(30);  
        t1.setWidth(15);  
        t1.area();  
    }  
}
```

[ 実行結果 ]

四角形の面積: 200

三角形の面積: 225

# 【課題の提出】

以下の流れで、作ったCプログラムをGitHubにプッシュして、Webサイトで確認してみましょう。

1. 端末内で、以下のコマンドで課題を提出

```
$ git add -A
```

```
$ git commit -m "課題9-4提出"
```

```
$ git push origin master
```

2. 自分のリポジトリを開いて、提出したファイルがプッシュされているか確認する

[https://github.com/nit-ibaraki-prog4d-2019/prog4d-\(ユーザー名\)](https://github.com/nit-ibaraki-prog4d-2019/prog4d-(ユーザー名))