OBJECT-TIGER: SYNTAX

Declaration

dec → *classdec*

classdec → **class** *class-id* **extends** *class-id* { {*classfield* } }

classfield → *vardec*

classfield → *method*

method → **method** *id*(*tyfields*) = *exp*

method → **method** *id*(*tyfields*) : *type-id* = *exp*

Expression

exp → **new** *class-id*

→ *lvalue* . *id*

→ *lvalue* . *id*(*exp*{, *exp*})

OBJECT — TIGER: EXAMPLE

```
let start := 10

    class Vehicle extends Object {
       var position := start
       method move(int x) = (position := position + x)
    }
    class Car extends Vehicle {
       var passengers := 0
       method await(v: Vehicle) =
            if (v.position < position)
                  then v.move(position - v.position)
            else self.move(10)
    }
    class Truck extends Vehicle {
       method move(int x) =
          if x <= 55 then position := position + x
    }

   var t := new Truck
   var c := new Car
   var v : Vehicle := c
in
    c.passengers := 2;
    c.move(60);
    v.move(70);
    c.await(t)
end
```

PROGRAM 14.1. An object-oriented program.

Single Inheritance

<Data Fields>

```
class A extends Object {
                    var a := 0}
class B extends A {var b := 0
                    var c := 0}
class C extends A {var d := 0}
class D extends B {var e := 0}
```

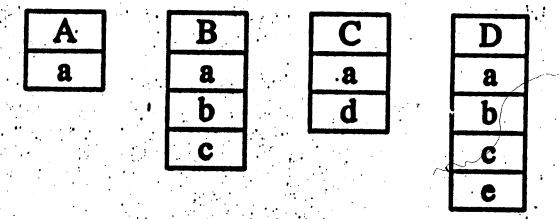
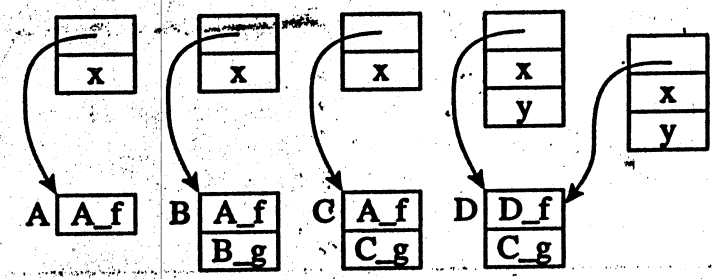


FIGURE 14.2. Single inheritance of data fields.

Single Inheritance

< Methods >

```
class A extends Object {
                    var x := 0
                    method f() }
class B extends A {method g() }
class C extends B {method g() }
class D extends C {var y := 0
                    method f() }
```



PROGRAM 14.3. Class descriptors for dynamic method lookup.

" Vtable "

< per class >

MULTIPLE INHERITANCE

```
class A extends Object {var a := 0}
class B extends Object {var b := 0
                        var c := 0}
class C extends A {var d := 0}
class D extends A,B,C {var e := 0}
```

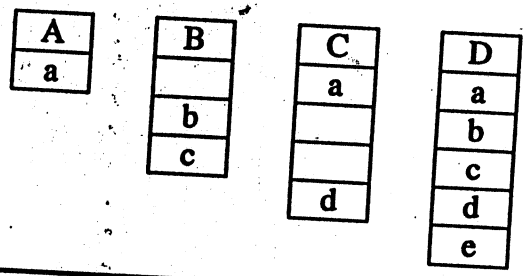
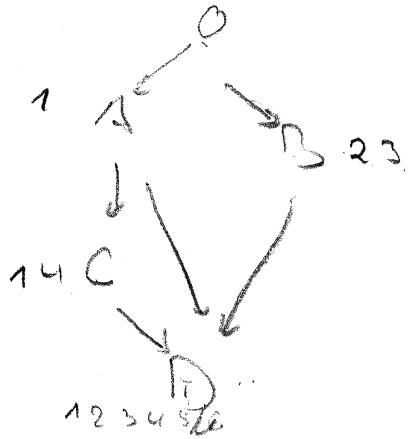


FIGURE 14.4. Multiple inheritance of data fields.

Graph coloring



A MORE CLEVER SOLUTION

Per Object

Per Class

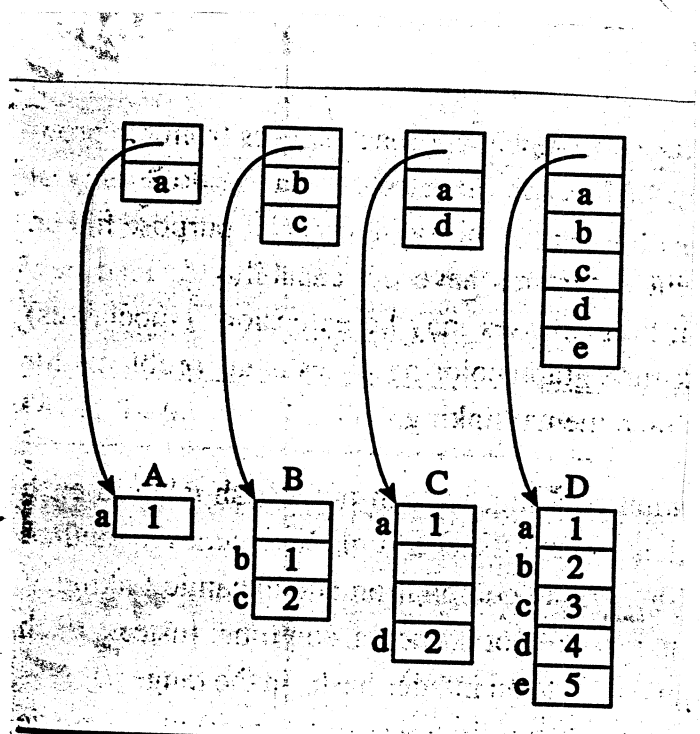
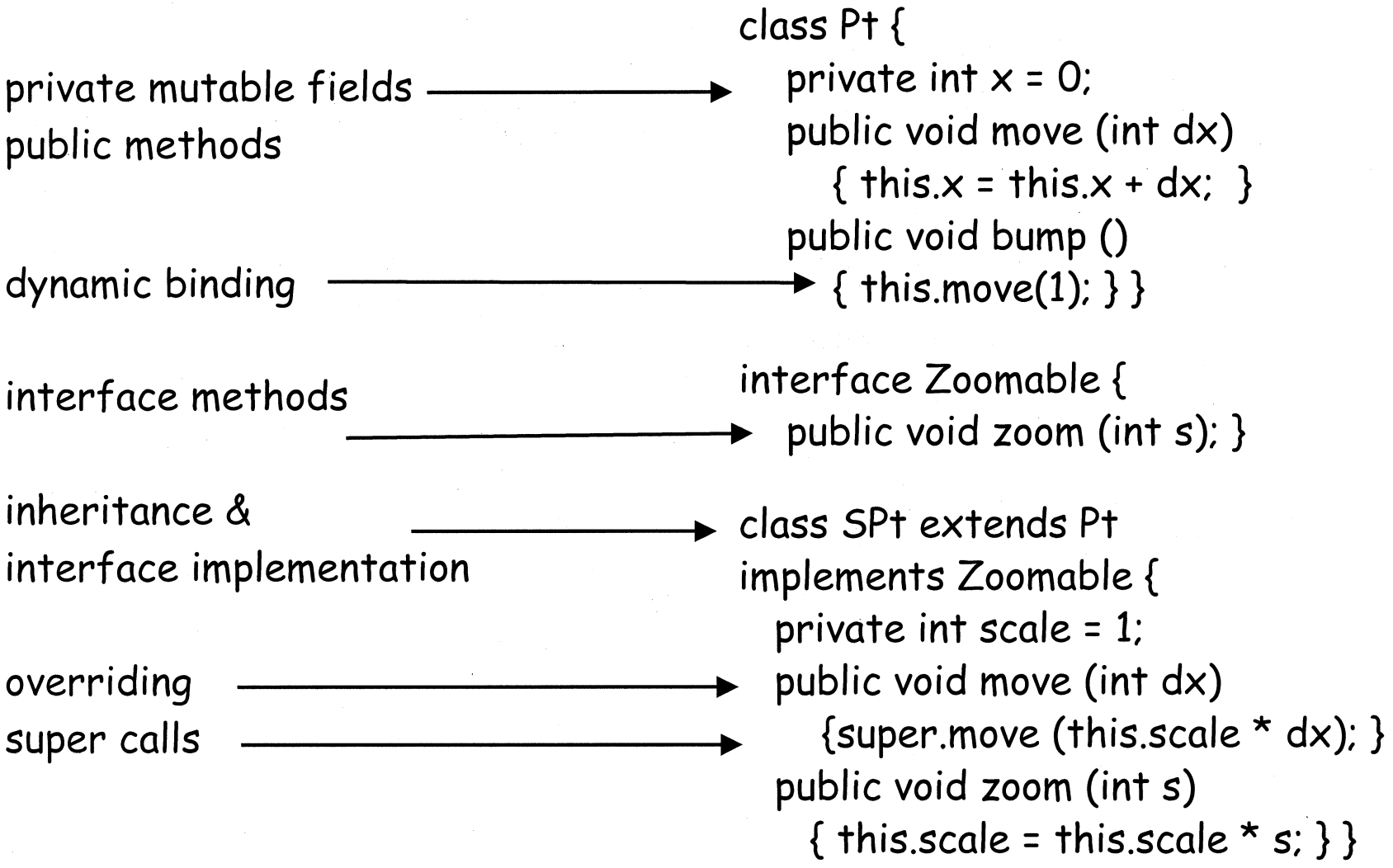


FIGURE 14.5. Field offsets in descriptors for multiple inheritance.

Sample Java program



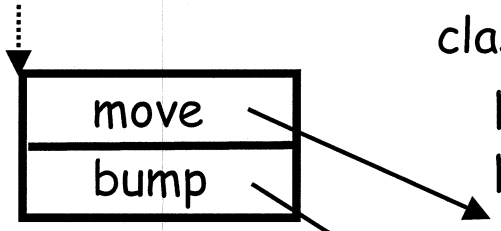
Sample object layout

```
class Pt {
   private int x = 0;
   public void move (int dx)
      { this.x = this.x + dx;  }
   public void bump ()
      { this.move(1); } }

interface Zoomable {
   public void zoom (int s);}

class SPt extends Pt
implements Zoomable {
  private int scale = 1;
  public void move (int dx)
     {super.move (this.scale * dx); }
  public void zoom (int s)
    { this.scale = this.scale * s; } }
```

Sample object layout: *Vtable*

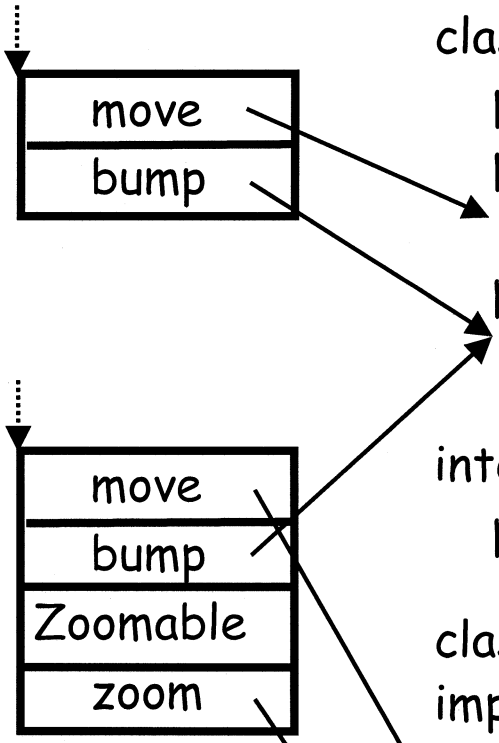



```
class Pt {
   private int x = 0;
   public void move (int dx)
      { this.x = this.x + dx;  }
   public void bump ()
      { this.move(1); } }

interface Zoomable {
   public void zoom (int s);}

class SPt extends Pt
implements Zoomable {
  private int scale = 1;
  public void move (int dx)
     {super.move (this.scale * dx); }
  public void zoom (int s)
    { this.scale = this.scale * s; } }
```

Sample object layout: *Vtable*

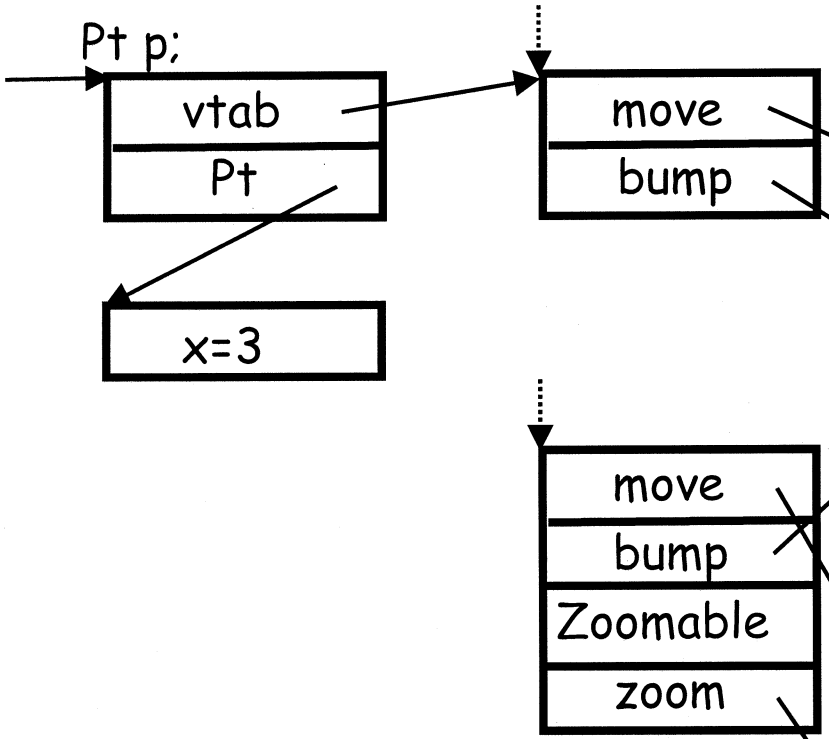



```
class Pt {
   private int x = 0;
   public void move (int dx)
     { this.x = this.x + dx;  }
   public void bump ()
     { this.move(1); } }

interface Zoomable {
   public void zoom (int s);}

class SPt extends Pt
implements Zoomable {
  private int scale = 1;
  public void move (int dx)
     {super.move (this.scale * dx); }
  public void zoom (int s)
    { this.scale = this.scale * s; } }
```

Sample object layout: *Pt* p

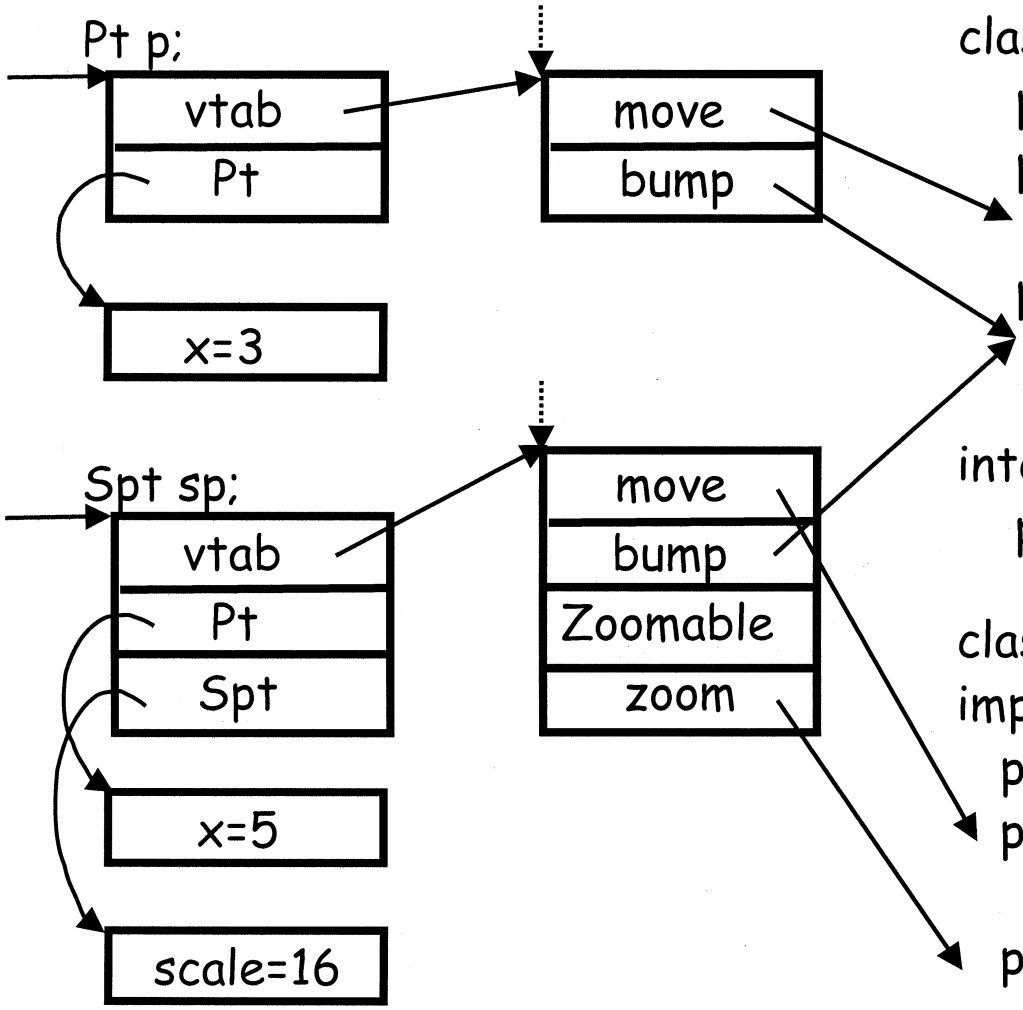



```
class Pt {
   private int x = 0;
   public void move (int dx)
     { this.x = this.x + dx;  }
   public void bump ()
     { this.move(1); } }

interface Zoomable {
   public void zoom (int s);}

class SPt extends Pt
implements Zoomable {
  private int scale = 1;
  public void move (int dx)
     {super.move (this.scale * dx); }
  public void zoom (int s)
    { this.scale = this.scale * s; } }
```

Sample object layout: *Spt* sp

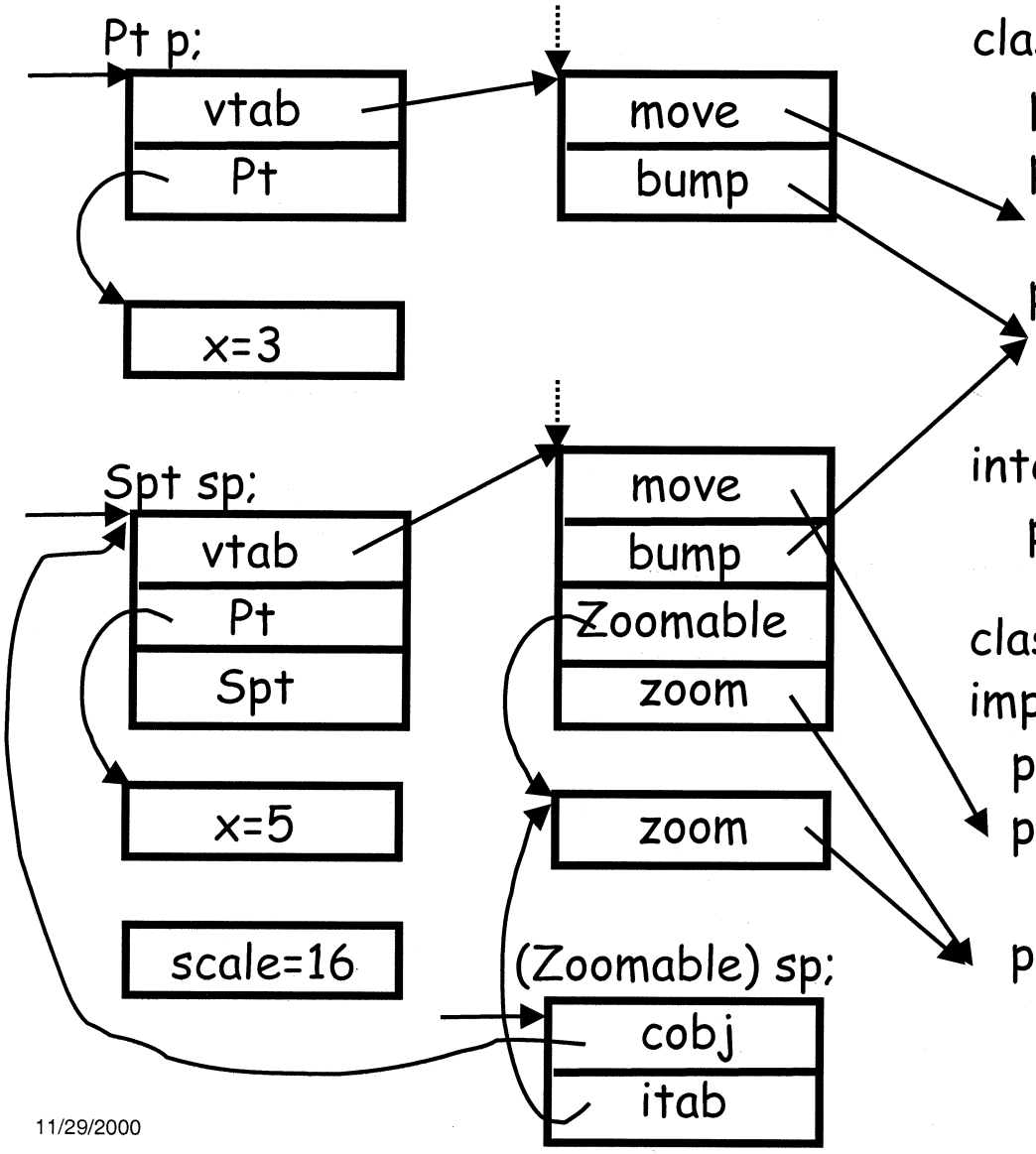



```
class Pt {
   private int x = 0;
   public void move (int dx)
     { this.x = this.x + dx;  }
   public void bump ()
     { this.move(1); } }

interface Zoomable {
   public void zoom (int s);}

class SPt extends Pt
implements Zoomable {
  private int scale = 1;
  public void move (int dx)
     {super.move (this.scale * dx); }
  public void zoom (int s)
    { this.scale = this.scale * s; } }
```

Sample object layout: *(Zoomable)* sp

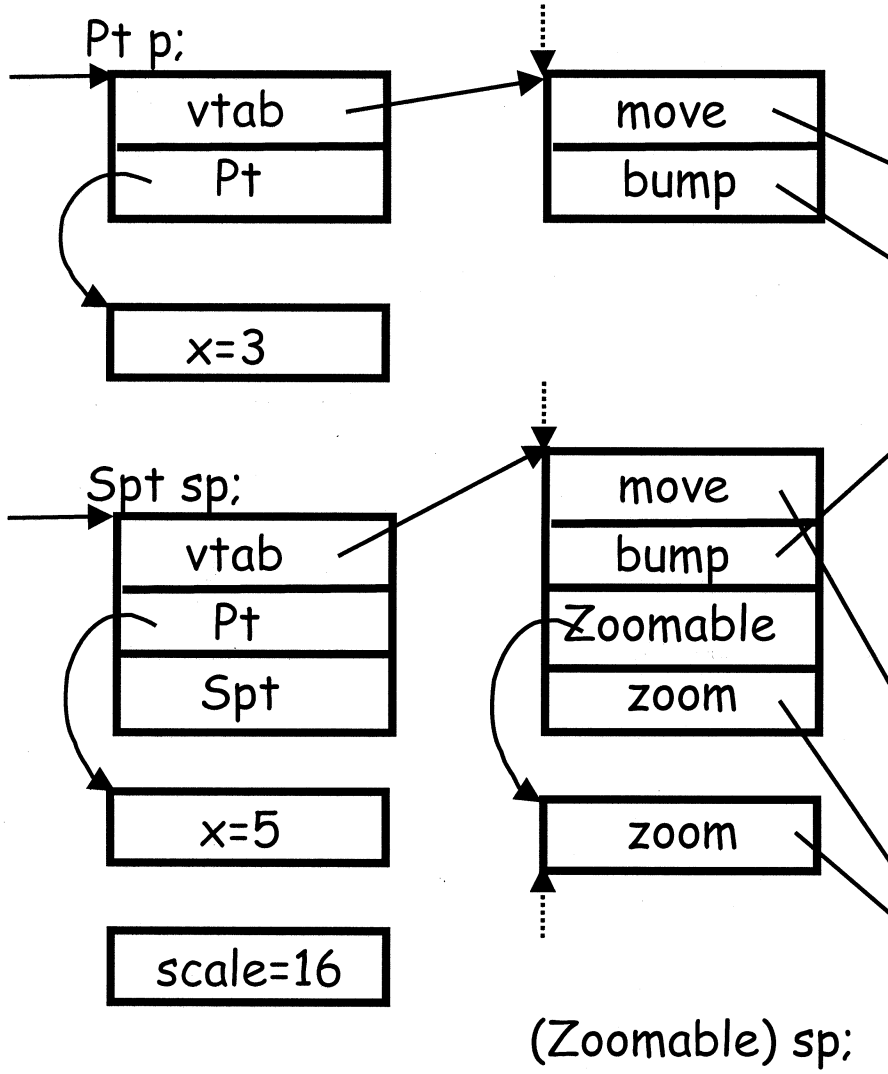



```
class Pt {
   private int x = 0;
   public void move (int dx)
     { this.x = this.x + dx;  }
   public void bump ()
     { this.move(1); } }

interface Zoomable {
   public void zoom (int s);}

class SPt extends Pt
implements Zoomable {
  private int scale = 1;
  public void move (int dx)
     {super.move (this.scale * dx); }
  public void zoom (int s)
    { this.scale = this.scale * s; } }
```

Sample object layout: *Itable*




```
class Pt {
   private int x = 0;
   public void move (int dx)
     { this.x = this.x + dx;  }
   public void bump ()
     { this.move(1); } }

interface Zoomable {
   public void zoom (int s);}

class SPt extends Pt
implements Zoomable {
  private int scale = 1;
  public void move (int dx)
     {super.move (this.scale * dx); }
  public void zoom (int s)
    { this.scale = this.scale * s; } }
```