

# Week 8

## Introducing Inheritance

EXPLORING  
**Java**



# This Week

**Lecture: Introducing inheritance &**

**Case study: a day at the zoo**

*Java Genesis:*

**–Ch8: Inheritance**

**Lab Assessment 6**

**Quick Quiz for Chapter 7**

# Object-oriented Reuse

## 1. Instantiation:

can create objects of a class

## 2. Inheritance:

can create a new class that is 'like'  
an existing class

# Example

## 1. Instantiation:

to cook a chocolate cake, reuse the chocolate cake recipe

## 2. Inheritance:

to create a recipe for cherry cake, take the chocolate cake recipe and replace 'chocolate' by 'cherries'

# Another Example

## 1. Instantiation:

new cars are produced by the car factory

## 2. Inheritance:

to build a new car factory, reuse the plans for the existing factory and modify them as required

# Extending CircleFigure

Extend the class **CircleFigure** to include a method

```
public static void smoothRight (int x)
```

which when invoked moves the circle right **x** pixels at the rate of 1 pixel every 10 milliseconds.

## CircleFigure

```
moveRight (int)
moveLeft (int)
moveUp (int)
moveDown (int)
moveTo (int, int)
setRadius (int)
setColour (Color)
drawHollow ( )
drawFilled ( )
getXCentre ( ) → int
getYCentre ( ) → int
```

## CircleFigure

```
moveRight (int)
moveLeft (int)
moveUp (int)
moveDown (int)
moveTo (int, int)
setRadius (int)
setColour (Color)
drawHollow ( )
drawFilled ( )
getXCentre ( ) → int
getYCentre ( ) → int
```



## BetterCircleFigure

```
smoothRight (int)
```

```
import genesis.*;

public class BetterCircleFigure
        extends CircleFigure {

    public static void smoothRight (int x) {
        int dist = Math.abs(x);
        for (int i=1; i<=dist; i++) {
            Delay.milliseconds(10);
            if (x>0) moveRight(1);
            else moveRight(-1);
        }
    }
}
```

# Tossing Dice

An object of the class **DiceGame** denotes:

- a game consisting of a number of dice;
- each die has a current value (in the range 1 to 6);
- the dice can be tossed with the result a new current value for each die is randomly chosen;
- the sum of the current values of the dice can be calculated;
- the current value of each die and the sum of these values can be output.

```

public class DiceGame {

    private int number;        // instance variables
    private int [ ] currentValues;

    public DiceGame (int num) { // constructor
        number = num;
        currentValues = new int [num];
    }

    public int randInt ( ) {
        return (int)(6*Math.random()+1);
    }

    public void tossAll ( ) {
        for (int i=0; i<number; i++) {
            currentValues[i] = randInt();
        }
    }
}

```

```
public int getSum ( ) {  
    int sum = 0;  
    for (int i=0; i<number; i++) {  
        sum = sum + currentValues[i];  
    }  
    return sum;  
}
```

```
public String toString ( ) {  
    String data = "";  
    for (int i=0; i<number; i++) {  
        data = data + currentValues[i] + " ";  
    }  
    data = data + " " + getSum();  
    return data;  
}
```

```
}
```

## DiceGame

```
int number  
int [ ] currentValues
```

```
DiceGame ()  
DiceGame (int)  
randInt () → int  
tossAll ()  
getSum () → int  
toString () → String
```

# Tossing Coins

## Observation:

Tossing coins is much like tossing dice except the outcome of each toss is 0 or 1 (for heads and tails respectively).

Therefore to create coin tossing games construct a class **CoinsGame** that inherits (extends) the class **DiceGame** and modifies the code that determines the outcome of a toss.

```
public class CoinsGame extends DiceGame {  
  
    public CoinsGame (int num) {  
        super(num);  
    }  
  
    public int randInt ( ) {  
        return super.randInt() % 2;  
    }  
}
```

## DiceGame

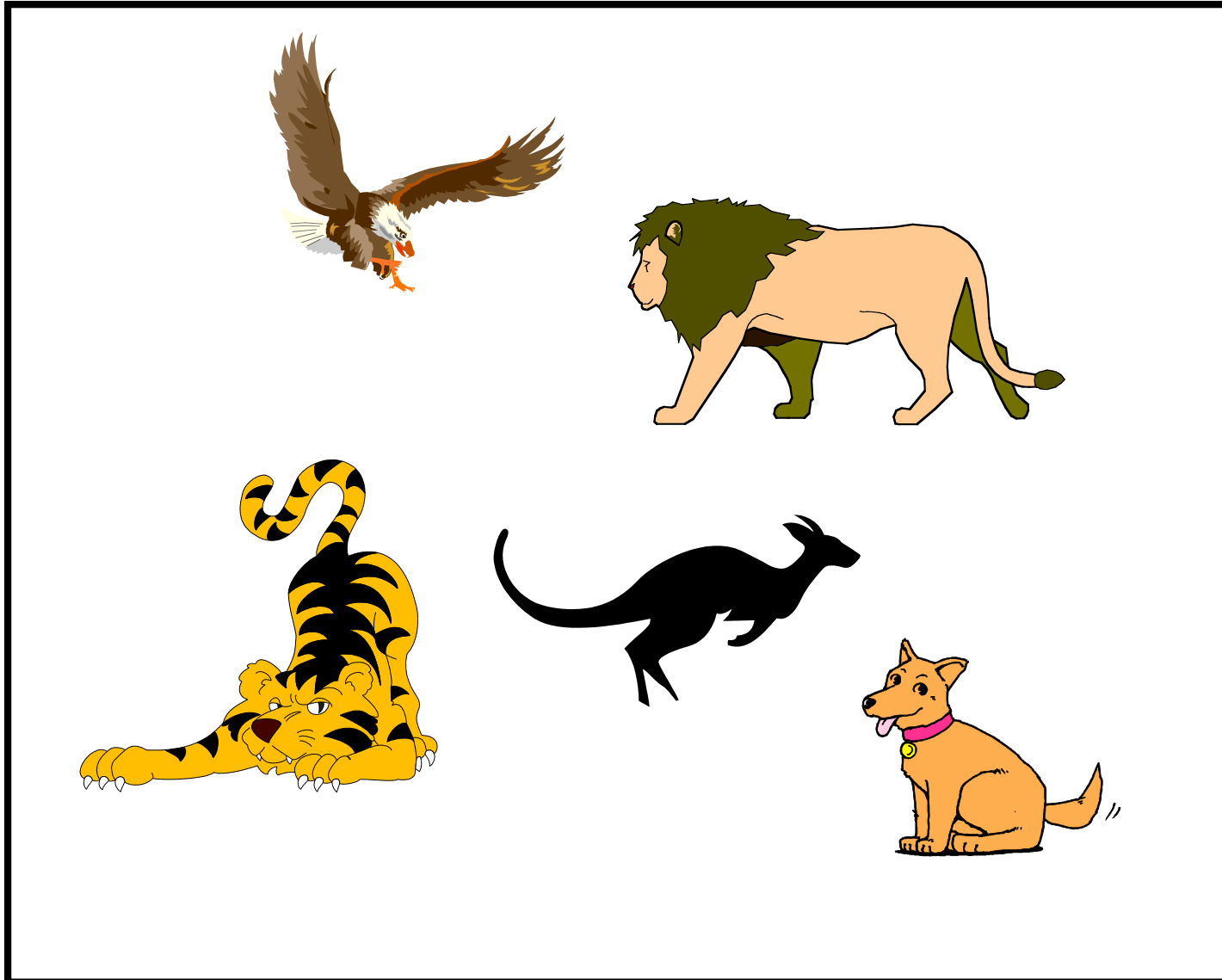
```
int number  
int [ ] currentValues
```

```
DiceGame ( )  
DiceGame (int)  
randInt ( ) → int  
tossAll ( )  
getSum ( ) → int  
toString ( ) → String
```

## CoinsGame

```
CoinsGame (int)  
randInt ( ) → int
```

# Case Study: a day at the Zoo



```
import java.awt.*;

public abstract class Animal {

    // instance variables
    protected int xPosn, yPosn;
    protected Color colour;

    // constructor methods
    public Animal (int x, int y) {
        xPosn = x;
        yPosn = y;
    }
}
```

```
// instance methods
public int getX ( ) {
    return xPosn;
}

public int getY ( ) {
    return yPosn;
}

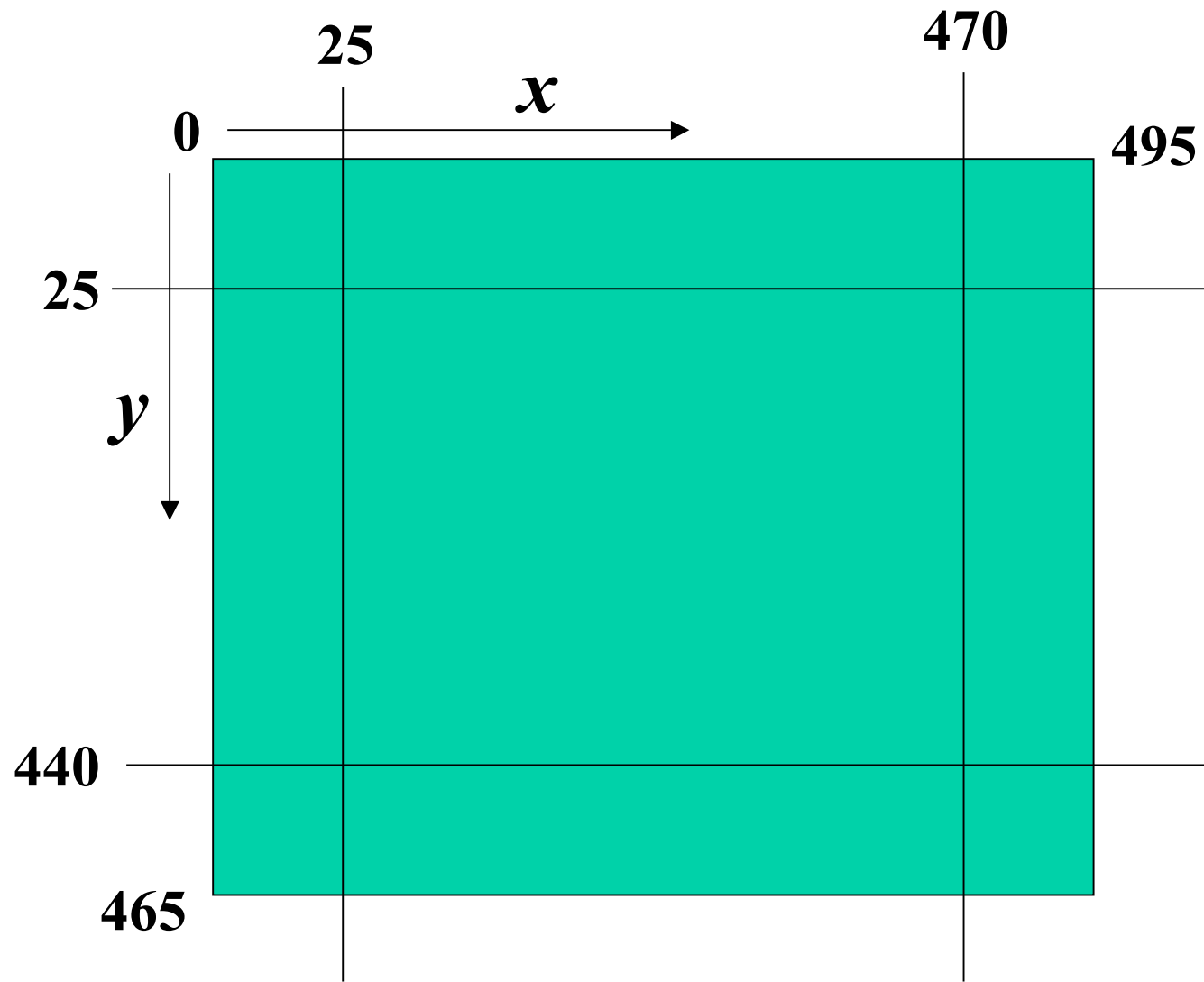
public Color getColour ( ) {
    return colour;
}

public abstract void move ( );
}
```

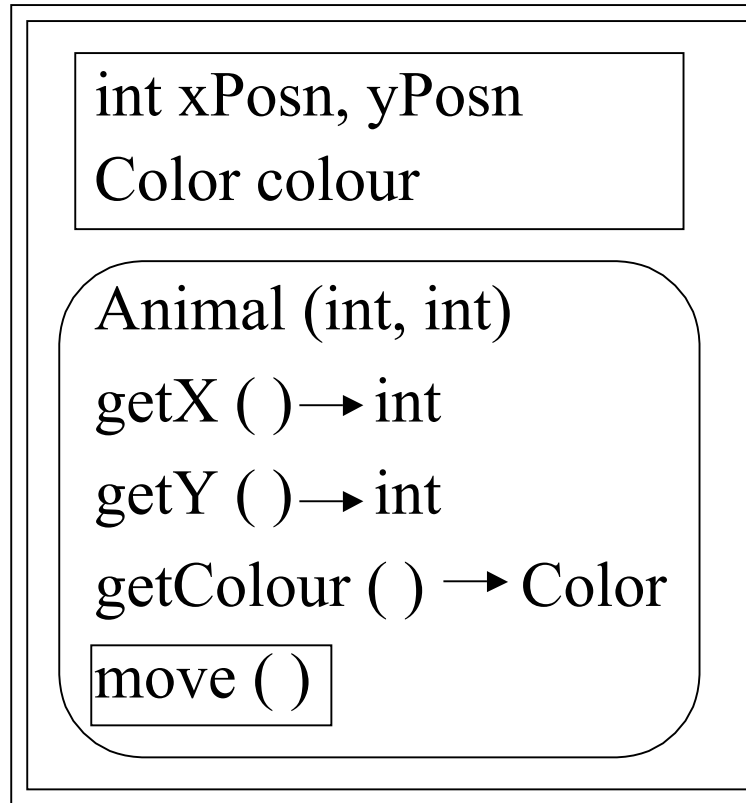
## Animal

```
int xPosn, yPosn  
Color colour
```

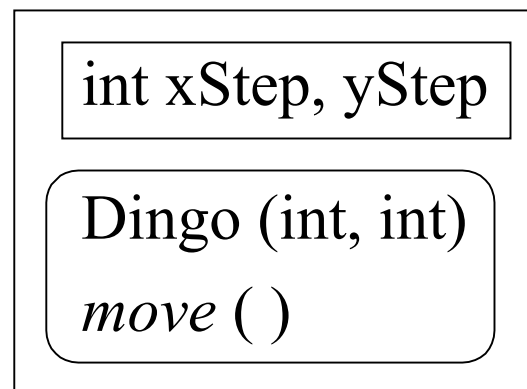
```
Animal (int, int)  
getX () → int  
getY () → int  
getColour () → Color  
move ()
```



## Animal



## Dingo



```

import java.awt.*;
public class Dingo extends Animal {

    protected int xStep, yStep;    // instance vars

    public Dingo (int x, int y) {    // constructor
        super(x, y);
        colour = Color.red;
        xStep = 5;
        yStep = 5;
    }
    public void move ( ) {
        if (xPosn > 470 || xPosn < 25)xStep = -xStep;
        xPosn = xPosn + xStep;
        if (yPosn > 440 || yPosn < 25)yStep = -yStep;
        yPosn = yPosn + yStep;
    }
}

```

```

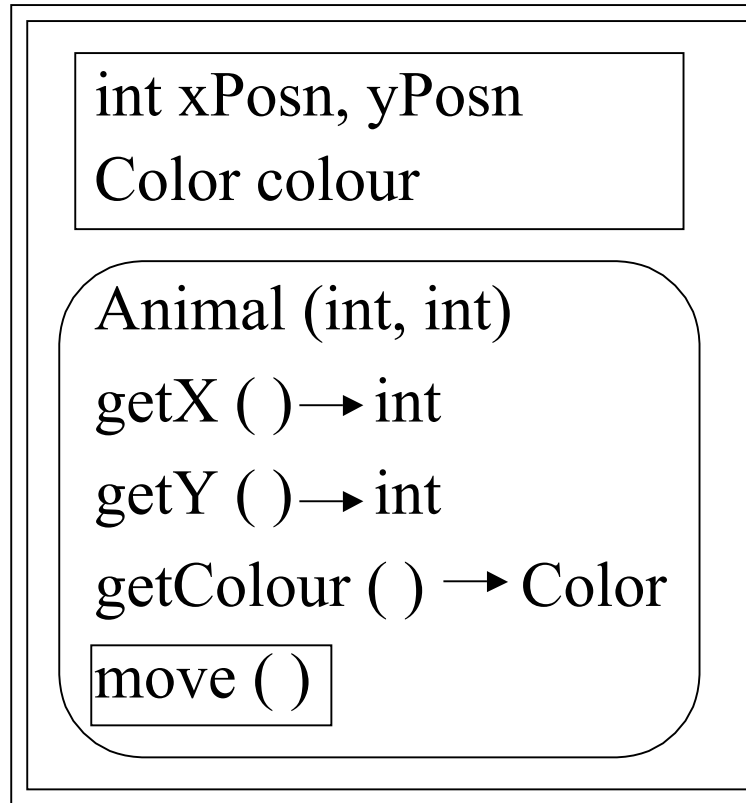
import genesis.*;

public class Zoo {

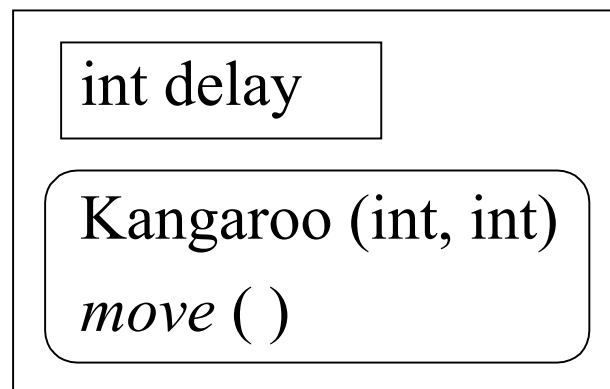
    public static void main (String [ ] args) {
        int animalNums = 1;
        Animal [ ] zoo = new Animal [animalNums];
        zoo[0] = new Dingo(400, 200);
        while (true) {
            for (int i=0; i<animalNums; i++) {
                zoo[i].move();
            }
            ZooWindow.display(zoo);
            Delay.milliseconds(20);
        }
    }
}

```

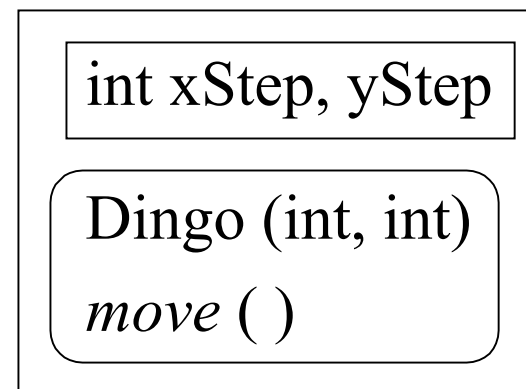
## Animal



## Kangaroo



## Dingo



```

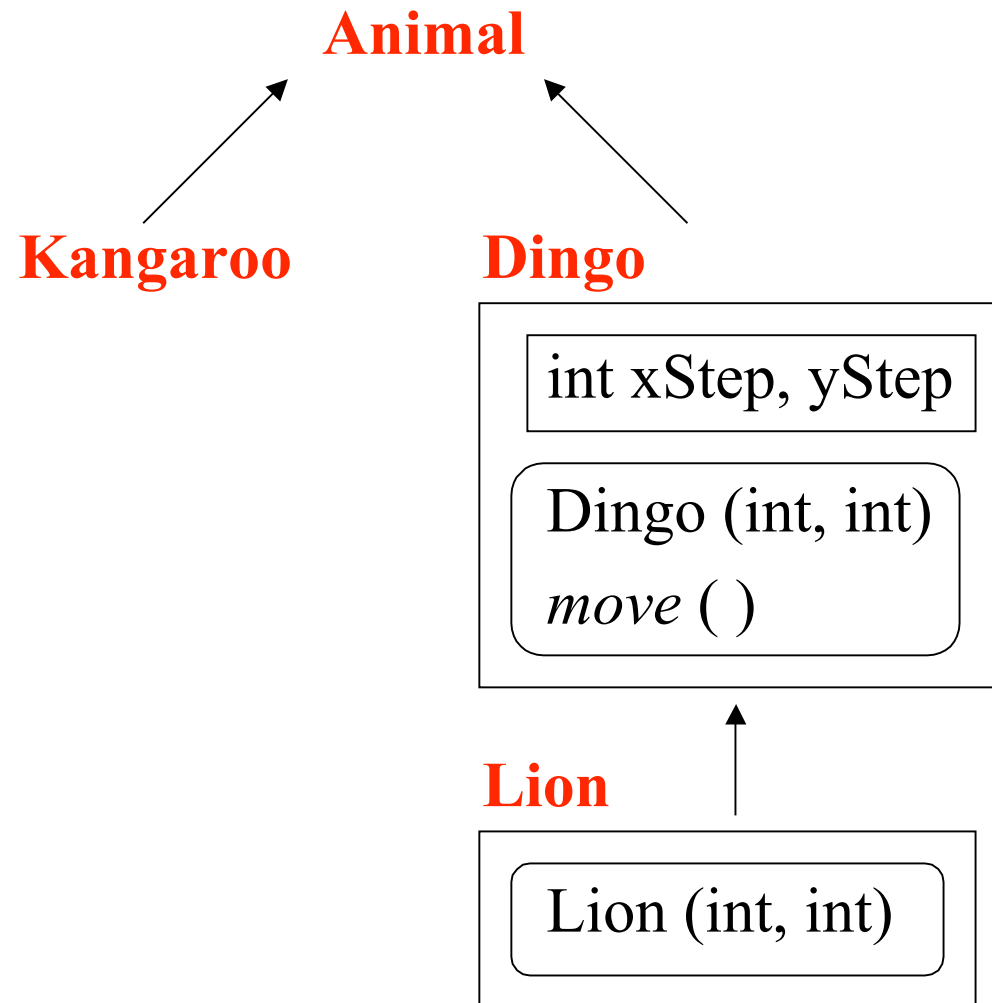
import java.awt.*;
public class Kangaroo extends Animal {

    protected int delay = 1;    // instance variable

    public Kangaroo (int x, int y) { // constructor
        super(x, y);
        colour = Color.green;
    }

    public void move ( ) {
        delay++;
        if (delay > 40) {
            xPosn = (int)(446*Math.random()+25);
            yPosn = (int)(416*Math.random()+25);
            delay = 1;
        }
    }
}

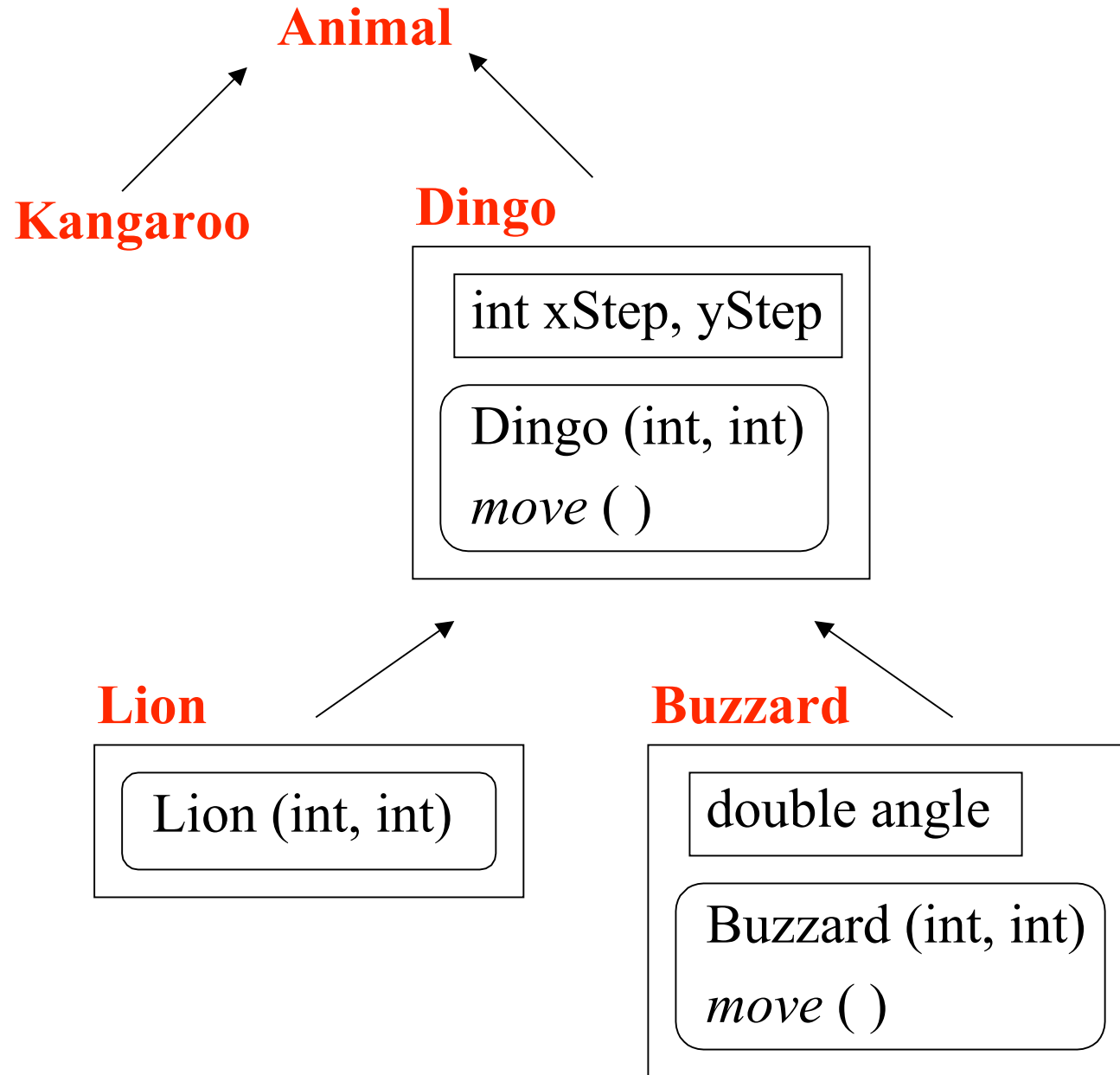
```



```
import java.awt.*;

public class Lion extends Dingo {

    // constructor
    public Lion (int x, int y) {
        super(x, y);
        colour = Color.magenta;
        xStep = 4;
        yStep = 0;
    }
}
```



```

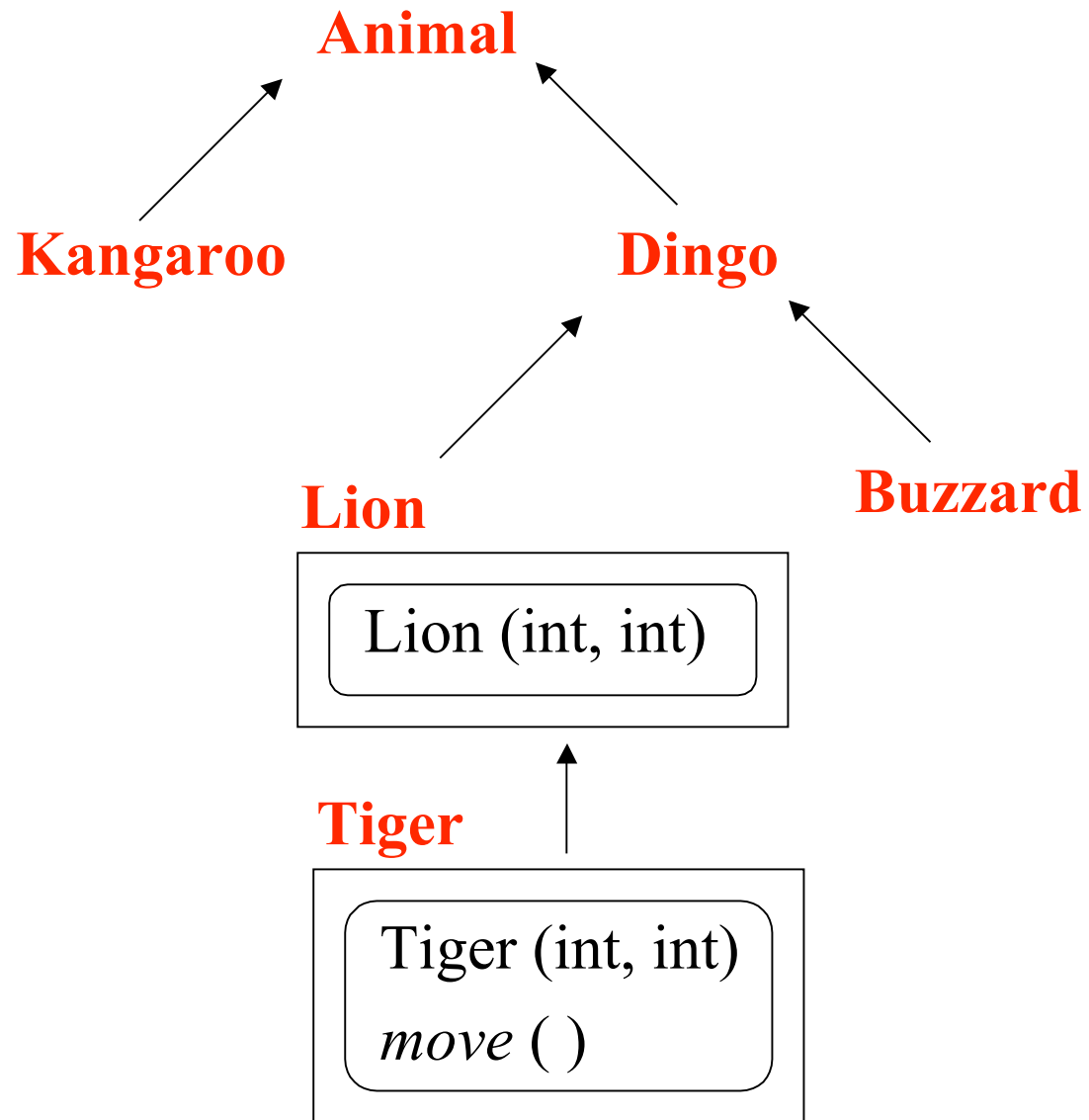
import java.awt.*;
public class Buzzard extends Dingo {

    // instance variables
    protected double angle = 0;

    // constructor
    public Buzzard (int x, int y) {
        super(x, y);
        colour = Color.gray;
    }

    public void move ( ) {
        xPosn = xPosn + (int)(xStep*Math.sin(angle));
        yPosn = yPosn + (int)(yStep*Math.cos(angle));
        angle = angle + 0.025;
    }
}

```



```
import java.awt.*;

public class Tiger extends Lion {

    // constructor
    public Tiger (int x, int y) {
        super(x, y);
        colour = Color.orange;
    }

    public void move ( ) {
        super.move();
        super.move();
    }
}
```

```

import genesis.*;
public class Zoo {

    public static void main (String [ ] args) {
        int animalNums = 5;
        Animal [ ] zoo = new Animal [animalNums];
        zoo[0] = new Dingo(400, 200);
        zoo[1] = new Kangaroo(300, 300);
        zoo[2] = new Lion(100, 100);
        zoo[3] = new Buzzard(100, 220);
        zoo[4] = new Tiger(200, 400);
        while (true) {
            for (int i=0; i<animalNums; i++)
                zoo[i].move();
            ZooWindow.display(zoo);
            Delay.milliseconds(20);
        }
    }
}

```