

Week 2

Arithmetic Operations, Iteration: for-loops

EXPLORING
Java



This Week

Lecture Arithmetic operations

Iteration: for-loops

Java Genesis:

–Ch3: Basic programming constructs

Labs begin

Concepts

- Variables
 - Named placeholder object
 - E.g. Counter, x, y, i
- Basic data types
 - Integers: `int`
 - Floating point: `double`
 - Characters: `string`

Operators

- Multiplication: $*$
- Division: $/$
- Subtraction: $-$
- Addition: $+$ (e.g. `Counter = Counter + 1`)
- Increment: $++$ (e.g. `Counter++`)

```
import genesis.*;
```

```
public class Fah2Cent {
```

```
    public static void main (String [ ] args) {
```

```
        double fahTemp;
```

```
        fahTemp = DialogBox.requestDouble("Fah temp:");
```

```
        double centTemp = (int)(100*(fahTemp-32)*5/9)/100.0;
```

```
        Transcript.println(centTemp);
```

```
    }
```

```
}
```

Iteration: for loop

```
for ( initial, test, increment) {  
    <some actions>  
}
```

```
int fact = 1;  
for (int i = 2; i <= 10; i = i+1) {  
    fact = fact*i;  
}  
Transcript.println(fact);
```

```
fact=1, i=2, fact = fact*i = 2 (=2!), i=i+1=3  
fact=2, i=3, fact = fact*i = 6 (=3!), i=i+1=4  
fact=6, i=4, fact = fact*i = 24 (=4!), i=i+1=5  
---  
fact=9!, i=10, fact = fact*i = 10!, i=i+1=11  
fact=10!, i=11, Transcript.println(fact)
```

```
import genesis.*;
```

```
public class Factorial {
```

```
    public static void main (String [ ] args) {
```

```
        int num = DialogBox.requestInt("supply integer:");
```

```
        double fact = 1;
```

```
        for (int i = 2; i <= num; i = i+1) {
```

```
            fact = fact*i;
```

```
        }
```

```
        Transcript.println(fact);
```

```
    }
```

```
}
```

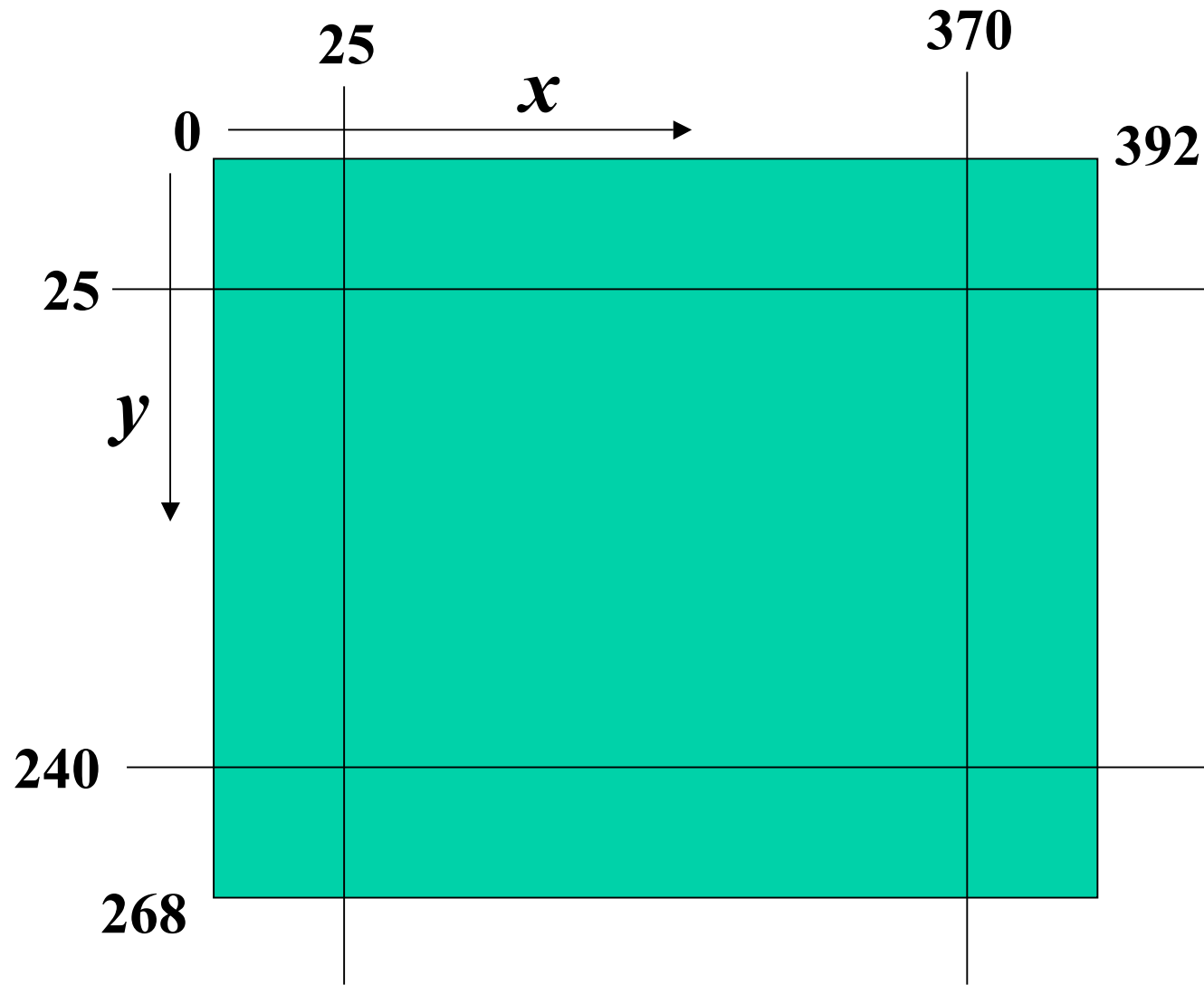
A Scenario

1. Open window with circle at its centre.
2. Wait one second.
3. Select at random a new position in the window.
4. Move the circle to the new position.
5. Repeat fifty times steps 2 to 4.

CircleFigure class

some messages:

- **create()**
- **moveTo(x, y)**
- **getXCentre()**
- **getYCentre()**



Math class

some messages:

- **random()**
- **sqrt(*x*)**
- **max(*x*, *y*)**
- **round(*x*)**

(For x-coordinate of circle's centre)

Random integer in range 25 to 370

$$370-25+1 = 346$$

$$0 < \text{Math.random()} < 1$$

$$0 < 346 * \text{Math.random()} < 346$$

(int)(346*Math.random()) int in range 0 to 345

(int)(346*Math.random()+25) in range 25 to 370

(For y-coordinate of circle's centre)

Random integer in range 25 to 240

$$240-25+1 = 216$$

$$0 < \text{Math.random()} < 1$$

$$0 < 216 * \text{Math.random()} < 216$$

(int)(216*Math.random()) int in range 0 to 215

(int)(216*Math.random()+25) in range 25 to 240

```
import genesis.*;


public class JumpingCircle {

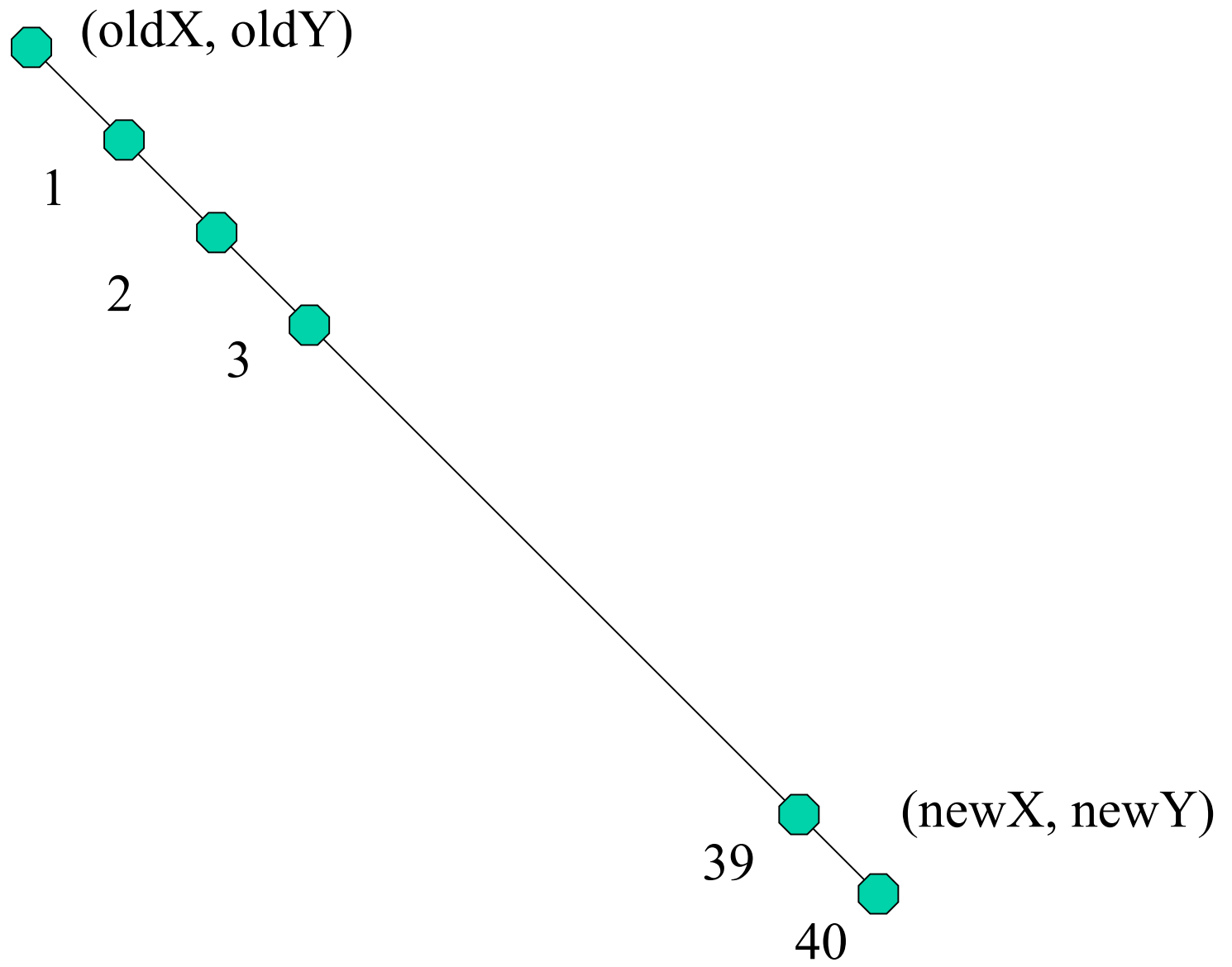
    public static void main (String [ ] args) {
        CircleFigure.create();
        for (int i=1; i<=50; i++) {
            Delay.milliseconds(1000);
            int newX = (int)(346*Math.random()+25;
            int newY = (int)(216*Math.random()+25;
            CircleFigure.moveTo(newX, newY);
        }
    }
}
```

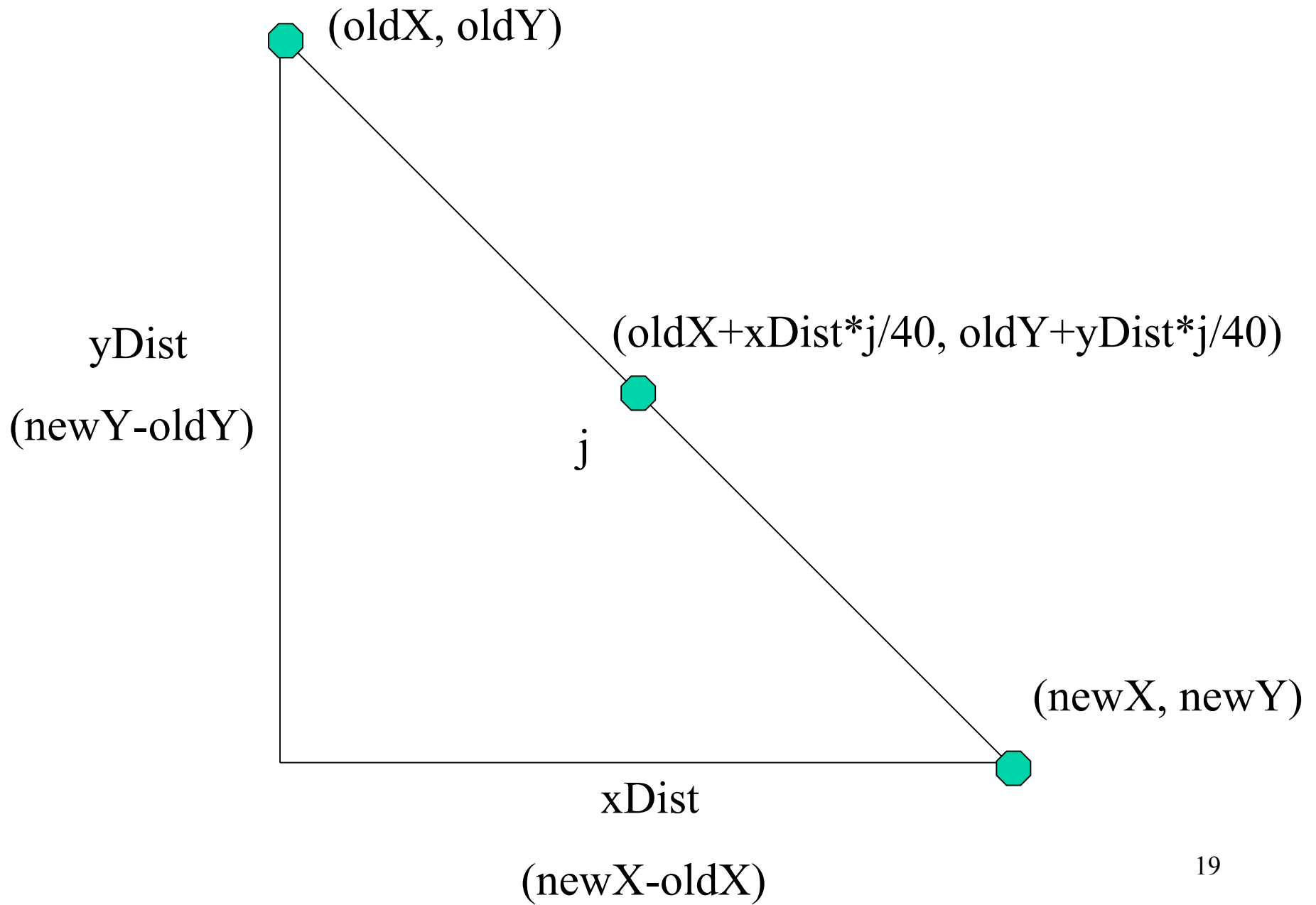
New Scenario

1. Open window with circle at its centre.
2. Select at random a new position in the window.
3. Move the circle to the new position by taking 40 equal steps waiting 25 milliseconds between steps.
4. Repeat fifty times steps 2 and 3.

 (oldX, oldY)

(newX, newY)






```

import genesis.*;
public class LostCircle {

    public static void main (String [ ] args) {
        CircleFigure.create( );
        int newX,newY,oldX,oldY,xDist,yDist,currentX,currentY;
        for (int i=1; i<=50; i++) {
            newX = (int)(346*Math.random( ))+25;
            newY = (int)(216*Math.random( ))+25;
            oldX = CircleFigure.getXCentre( );
            oldY = CircleFigure.getYCentre( );
            xDist = newX - oldX;
            yDist = newY - oldY;
            for (int j=1; j<=40; j++) {
                Delay.milliseconds(25);
                currentX = (int)(oldX+xDist*j/40.0);
                currentY = (int)(oldY+yDist*j/40.0);
                CircleFigure.moveTo(currentX, currentY);
            }
        }
    }
}

```