# Week 4

# Arrays

# Announcements

- **1st Assignment** is due at the end of **this week**: deadline **4pm Friday**
- the deadline for the **first three lab assessments** is the end of **next week**
- Tech support tutorials with Derryn McMaster:

  **Friday** at **1-3pm** in PC labs

# This Week

**Lecture: Introducing and**

**Working with arrays**

***Java Genesis*:**

– **Ch5: Arrays** (Section 5.4 is optional)

**Quick Quiz for Chapter 4**

# Problem: preparing a time-sheet

We want a program to prepare time-sheets.

An employee supplies the number of days worked and the hours worked each day. The program then prints out details of their salary.

In each day an employee is paid $15/hour for the first 8 hours and $20/hour thereafter.

# Example

An employee works for **5** days and the hours worked on these days are **9**, **5**, **11**, **8** and **4**.

This employee's time-sheet will be:

| Day | Hours | Salary |
|---|---|---|
| 1 | 9 | $140 |
| 2 | 5 | $75 |
| 3 | 11 | $180 |
| 4 | 8 | $120 |
| 5 | 4 | $60 |

**Total hours: 37**

**Total salary: $575**

# Introducing the array

Mathematically, could store hours worked in a sequence:

$hours = \{9, 5, 11, 8, 4\}$ where

$hours_1=9$, $hours_2=5$, …, $hours_5=4$

In Java a sequence is captured by an **array**.

# Arrays

An array is a data structure in which

- the elements are listed in some order

  i.e. the 0th, 1st, 2nd, …, (*n*-1)th, where *n* is the length of the array;

- all the elements are of the same type

  (the **base-type** of the array).

# Creating an array

The sequence

$hours = \{9, 5, 11, 8, 4\}$ where

$hours_1=9$, $hours_2=5$, …, $hours_5=4$

in Java becomes the **array**:

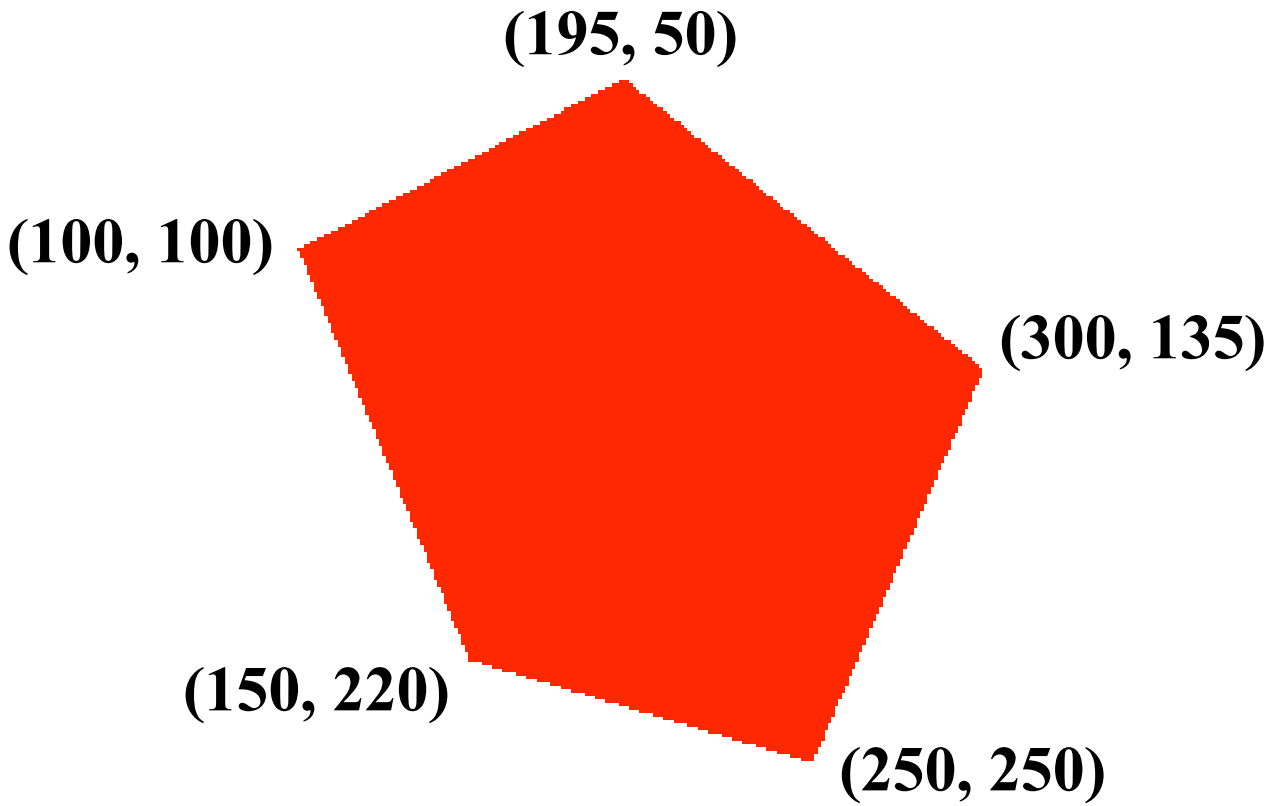
**int [ ] hours = new int [5]** where

hours[0]=9, hours[1]=5, …, hours[4]=4

```
import genesis.*;
public class TimeSheetExtra {

 public static void main (String [ ] args) {
  int days = DialogBox.requestInt("num days worked");
  int [ ] hours = new int [days];
  int [ ] salary = new int [days];
  int totalHours = 0;
  int totalSalary = 0;
  Transcript.println("Day\tHours\tSalary");
  for (int i=0; i<days; i++) {
   hours[i] = DialogBox.requestInt("hours on day "+(i+1));
   if (hours[i] <= 8) salary[i] = 15*hours[i];
   else salary[i] = 15*8 + (hours[i]-8)*20;
   Transcript.println((i+1)+"\t"+hours[i]+"\t$"+salary[i]);
   totalHours = totalHours + hours[i];
   totalSalary = totalSalary + salary[i];
  }
  Transcript.println("Total hours: "+totalHours);
  Transcript.println("Total salary: $"+totalSalary);
 }
}
```

# A Polygon



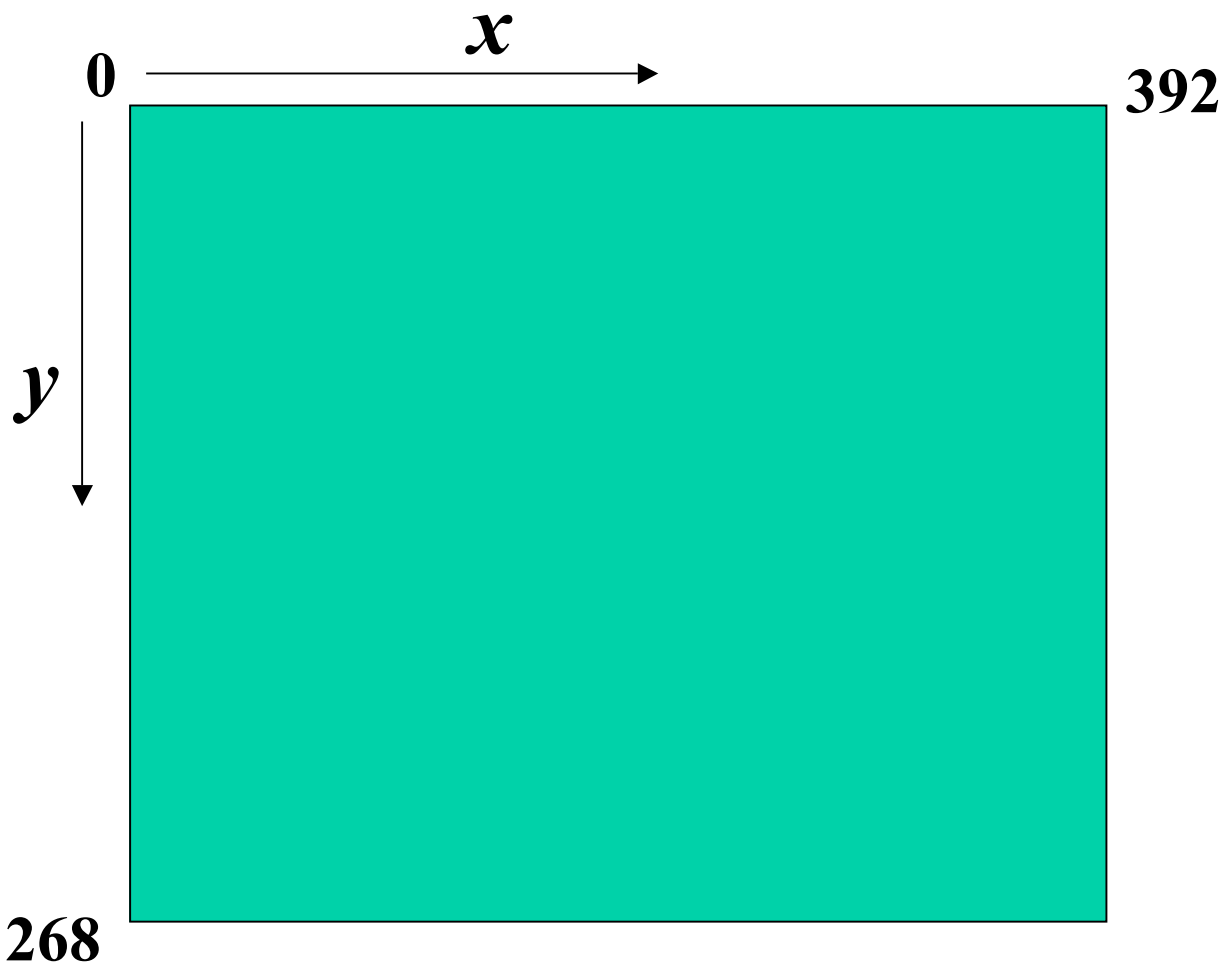
# The Shape class

When sent the message **create()** an empty window opens.

When sent the message **draw(xs, ys)** where **xs** and **ys** are integer arrays of the same length, a solid red polygon is drawn with vertices **(xs[0], ys[0]), (xs[1], ys[1]), (xs[2], ys[2]), ….**

```
public class DrawShape {

   public static void main (String [ ] args) {
      Shape.create();
      int [ ] xCoords = {195, 300, 250, 150, 100};
      int [ ] yCoords = {50, 135, 250, 220, 100};
      Shape.draw(xCoords, yCoords);
   }
}
```

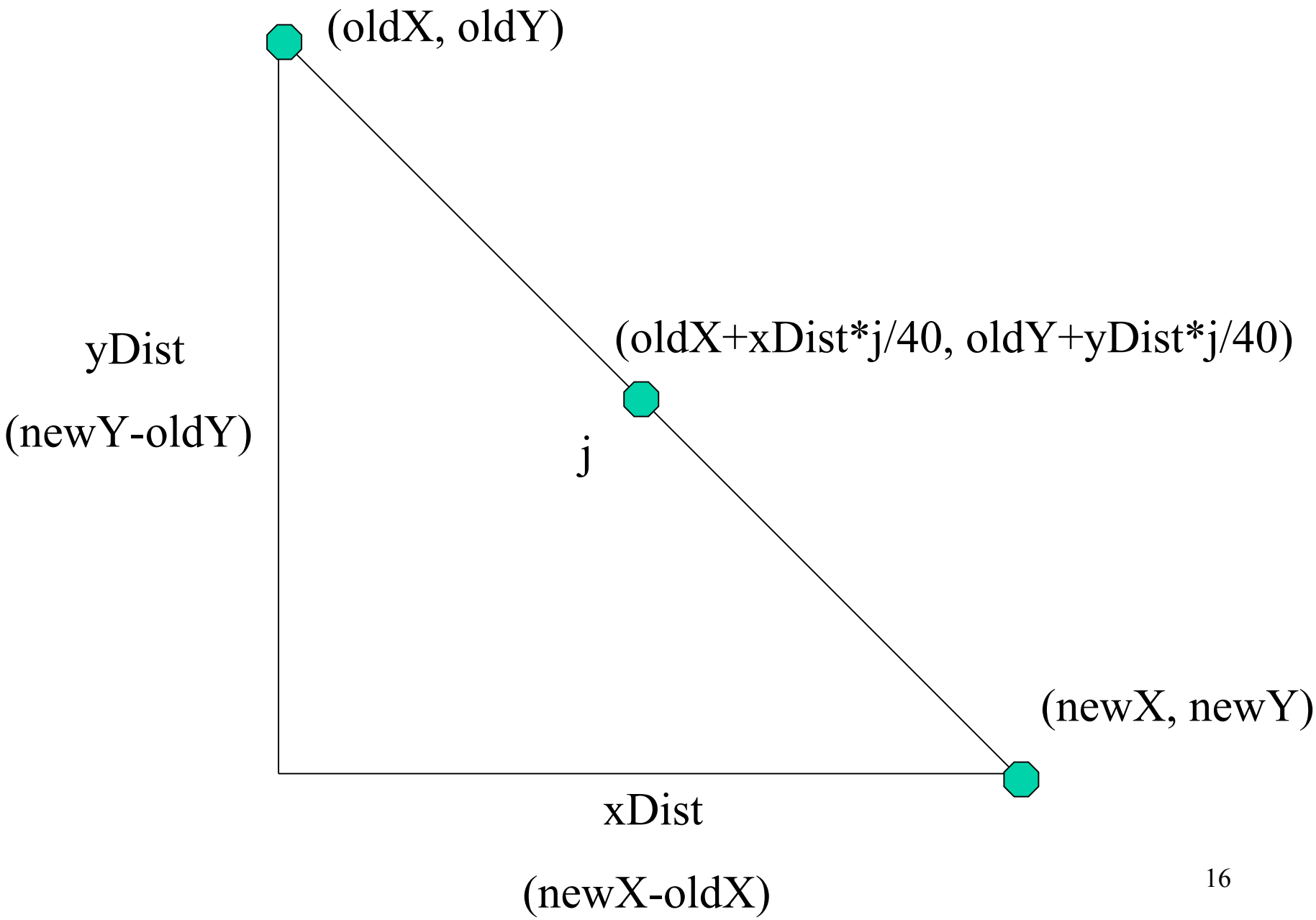

x
0
392
y
268

```
public class DrawShape {

   public static void main (String [ ] args) {
      Shape.create();
      int num = 40;
      int [ ] xCoords = new int [num];
      int [ ] yCoords = new int [num];
      for (int i=0; i<num; i++) {
           xCoords[i] = (int)(392*Math.random());
           yCoords[i] = (int)(268*Math.random());
      }
      Shape.draw(xCoords, yCoords);
   }
}
```

```
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);
              }
       }
   }
}
```


(oldX, oldY)
yDist
(newY-oldY)
(oldX+xDist*j/40, oldY+yDist*j/40)
j
(newX, newY)
xDist
(newX-oldX)

```
import genesis.*;
public class LostShape {
    public static void main (String [ ] args) {
        int [ ] xCoords = {195, 300, 250, 150, 100};
        int [ ] yCoords = {50, 135, 250, 220, 100};
        Shape.create( );
        Shape.draw(xCoords, yCoords);
        int newX, newY, oldX, oldY, xDist, yDist;
        while (true) {
                newX = (int)(392*Math.random());
                newY = (int)(268*Math.random());
                oldX = xCoords[2];
                oldY = yCoords[2];
                xDist = newX - oldX;
                yDist = newY - oldY;
                for (int j=1; j<=40; j++) {
                        Delay.milliseconds(25);
                        xCoords[2] = (int)(oldX+xDist*j/40.0);
                        yCoords[2] = (int)(oldY+yDist*j/40.0);
                        Shape.draw(xCoords, yCoords);
                }
        }
    }
}
```

```
import genesis.*;

public class LostShape {

  public static void main (String [ ] args) {
       int num = 40;
       int [ ] xCoords = new int [num];
       int [ ] yCoords = new int [num];
       for (int i=0; i<num; i++) {
             xCoords[i] = 200;
             yCoords[i] = 150;
       }
       Shape.create( );
       Shape.draw(xCoords, yCoords);
       int [ ] newX = new int [num];
       int [ ] newY = new int [num];
       int [ ] oldX = new int [num];
       int [ ] oldY = new int [num];
       int [ ] xDist = new int [num];
       int [ ] yDist = new int [num];
```

```
        while (true) {
            for (int i=0; i<num; i++) {
                newX[i] = (int)(392*Math.random());
                newY[i] = (int)(268*Math.random());
                oldX[i] = xCoords[i];
                oldY[i] = yCoords[i];
                xDist[i] = newX[i] - oldX[i];
                yDist[i] = newY[i] - oldY[i];
            }
            for (int j=1; j<=40; j++) {
                Delay.milliseconds(50);
                for (int i=0; i<num; i++) {
                   xCoords[i] = (int)(oldX[i]+xDist[i]*j/40.0);
                   yCoords[i] = (int)(oldY[i]+yDist[i]*j/40.0);
                }
                Shape.draw(xCoords, yCoords);
            }
        }
    }
}
```

```
import genesis.*;
import java.awt.*;
public class RandomColours {

  public static void main (String [ ] args) {
    CircleFigure.create( );
    CircleFigure.setRadius(100);
    int rand;
    while (true) {
      Delay.milliseconds(500);
      rand = (int)(5*Math.random( ));
      switch (rand) {
       case 0:CircleFigure.setColour(Color.red); break;
       case 1:CircleFigure.setColour(Color.blue); break;
       case 2:CircleFigure.setColour(Color.green); break;
       case 3:CircleFigure.setColour(Color.magenta);break;
       case 4:CircleFigure.setColour(Color.yellow); break;
      }
    }
  }
}
```

```
import genesis.*;
import java.awt.*;
public class RandomColours {

  public static void main (String [ ] args) {
      Color [ ] colours =
            {Color.red, Color.blue, Color.green,
                        Color.magenta, Color.yellow};
      CircleFigure.create( );
      CircleFigure.setRadius(100);
      int rand;
      while (true) {
            Delay.milliseconds(500);
            rand = (int)(5*Math.random( ));
            CircleFigure.setColour(colours[rand]);
      }
   }
}
```