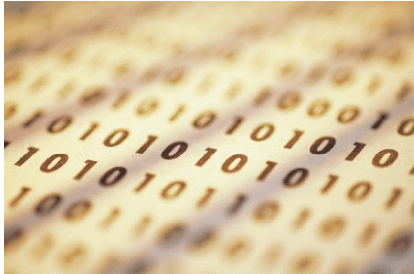


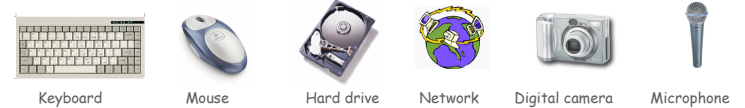
2.2 Input and Output



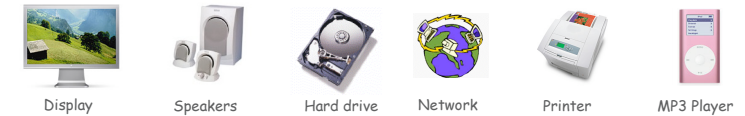
Today's goal. Java programs that interact with outside world.

Input and Output

Input devices.



Output devices.

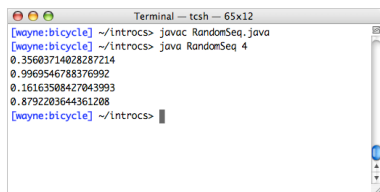


Our approach.

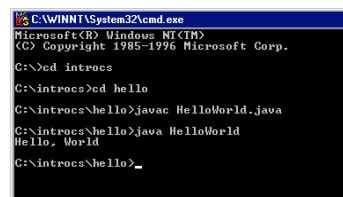
- Define Java libraries of functions for input and output.
- Use operating system (OS) to connect Java programs to: file system, each other, display.

Terminal

Terminal. Application where you can type commands to control the operating system.



Mac OS X



Microsoft Windows

Bird's Eye View

Command line input. Read an integer N from command line.

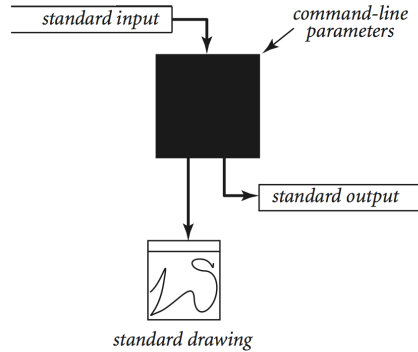
Standard output.

- Flexible OS abstraction for output.
- In Java, output from `System.out.println()` goes to `stdout`.
- By default, `stdout` is sent to Terminal.

```
public class RandomSeq {  
    public static void main(String[] args) {  
        int N = Integer.parseInt(args[0]);  
        for (int i = 0; i < N; i++)  
            System.out.println(Math.random());  
    }  
}
```

```
% java RandomSeq 4  
0.9320744627218469  
0.4279508713950715  
0.08994615071160994  
0.6579792663546435
```

New Bird's Eye View



6

Standard Input Abstraction

Command line inputs.

- Use command line inputs to read in a **few** user values.
- Not practical for many user inputs.
- Input entered **before** program begins execution.

Standard input.

- Flexible OS abstraction for input.
- By default, `stdin` is received from Terminal window.
- Input entered **while** program is executing.

7

Standard Input

Standard input. We provide library `StdIn` to read text input.
To use. Download `StdIn.java` from booksite and put in working directory.

```
public class Add {
    public static void main(String[] args) {
        System.out.println("Type the first integer");
        int x = StdIn.readInt();
        System.out.println("Type the second integer");
        int y = StdIn.readInt();
        int z = x + y;
        System.out.println("Their sum is " + z);
    }
}
```

```
% java Add
Type the first integer
1
Type the second integer
2
Their sum is 3
```

8

Twenty Questions

Twenty questions. User thinks of an integer between one and 1 million.
 Computer tries to guess it.

```
public class TwentyQuestions {
    public static void main(String[] args) {
        int lo = 1, hi = 1000000;
        while (lo < hi) {
            int mid = (lo + hi) / 2;
            System.out.println("Is your number <= " + mid + "?");
            boolean response = StdIn.readBoolean();
            if (response) hi = mid;
            else lo = mid + 1;
        }
        System.out.println("Your number is " + lo);
    }
}
```

Binary search. Each question removes half of possible remaining values.

Consequence. Always succeeds after 20 questions.

$2^{20} \approx 1 \text{ million}$

Invariant: user's number
always between lo and hi

9

Averaging A Stream of Numbers

Average. Read in real numbers, and print their average.

```
public class Average {
    public static void main(String[] args) {
        double sum = 0.0;
        int N = 0;

        while (!StdIn.isEmpty()) {
            double x = StdIn.readDouble();
            sum = sum + x;
            N++;
        }

        System.out.println(sum / N);
    }
}
```

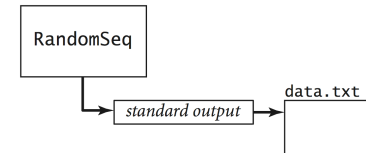
```
% java Average
10.0 5.0 6.0
3.0 7.0 32.0
<Ctrl-d>
10.5
```

<Ctrl-d> is Mac/Linux/Unix EOF
<Ctrl-z> is Windows analog

10

Redirecting Standard Output

Redirecting standard output. Use OS directive to send standard output to a file for permanent storage (instead of terminal window).



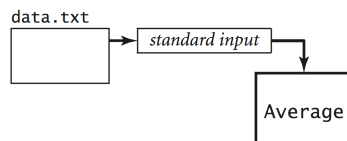
```
% java RandomSeq 1000 > data.txt
```

redirect stdout

11

Redirecting Standard Input

Redirecting standard input. Use OS directive to read standard input from a file (instead of terminal window).



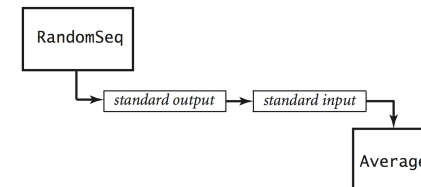
```
% more < data.txt
0.5475375782884312
0.4971087292684019
0.23123808041753813
...
% java Average < data.txt
0.4947655567740991
```

redirect stdin

12

Connecting Programs

Piping. Use OS directive to make the standard output of one program become the standard input of another.



```
% java RandomSeq 1000000 | java Average
0.4997970473016028
```

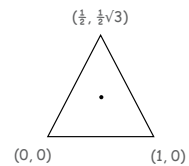
13

Standard Draw

Standard draw. We provide library `StdDraw` to plot graphics.
To use. Download `StdDraw.java` and put in working directory.

```
public class Triangle {
    public static void main(String args[]) {
        double t = Math.sqrt(3.0) / 2.0;
        StdDraw.line(0.0, 0.0, 1.0, 0.0);
        StdDraw.line(1.0, 0.0, 0.5, t);
        StdDraw.line(0.5, t, 0.0, 0.0);
        StdDraw.point(0.5, t/3.0);
    }
}
```

```
% java Triangle
```



14

Data Visualization

Plot filter. Read in a sequence of xy-coordinates from standard input, and plot using standard draw.

```
public class PlotFilter {
    public static void main(String args[]) {
        double xmin = StdIn.readDouble();
        double ymin = StdIn.readDouble();
        double xmax = StdIn.readDouble();
        double ymax = StdIn.readDouble();
        StdDraw.setXscale(xmin, xmax);
        StdDraw.setYscale(ymin, ymax);

        while (!StdIn.isEmpty()) {
            double x = StdIn.readDouble();
            double y = StdIn.readDouble();
            StdDraw.point(x, y);
        }
    }
}
```

rescale coordinate system

read in points, and plot them

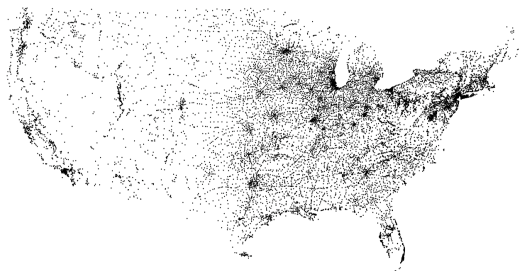
15

Data Visualization

```
% more < USA.txt
669905.0 247205.0 1244962.0 490000.0
1097038.8890 245552.7780
1103961.1110 247133.3330
1104677.7780 247205.5560
...
% java PlotFilter < USA.txt
```

bounding box

coordinates of 13,509 US cities



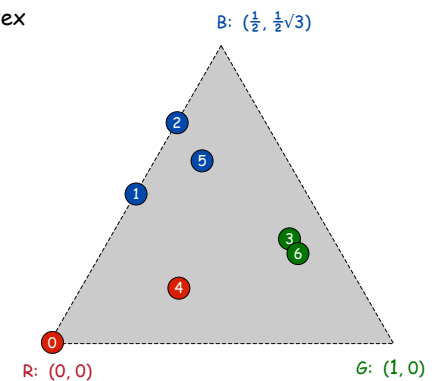
16

Chaos Game

Chaos game. Play on equilateral triangle, with vertices R, G, B.

- Start at R.
- Repeat the following N times:
 - pick a random vertex
 - move halfway between current point and vertex
 - draw a point in color of vertex

Q. What picture emerges?



17

Chaos Game

```
public class Chaos {
    public static void main(String args[]) {
        int N = Integer.parseInt(args[0]);
        double[] cx = { 0.000, 1.000, 0.500 };
        double[] cy = { 0.000, 0.000, 0.866 };
        double x = 0.0, y = 0.0;

        for (int i = 0; i < N; i++) {
            int r = StdRandom.uniform(3);
            x = (x + cx[r]) / 2.0;
            y = (y + cy[r]) / 2.0;
            StdDraw.point(x, y);
        }
    }
}
```

$\frac{1}{2}\sqrt{3}$
(avoid hardwired
constants like this)

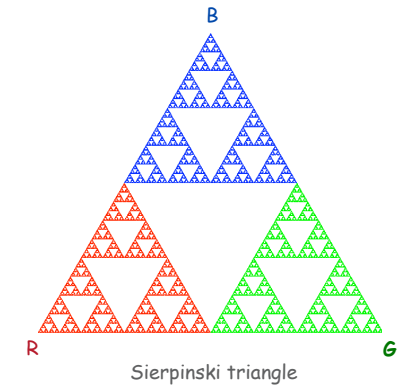
equally likely to be
0, 1, or 2

Chaos Game

Easy modification. Color point according to random vertex chosen using `StdDraw.setPenColor(StdDraw.RED)` to change the pen color.

see text for list of available colors
(see Section 3.1 for creating your own colors)

```
% java Chaos 10000
```



18

19

Barnsley Fern

Barnsley fern. Play chaos game with different rules.

probability	new x	new y
2%	.50	.27y
15%	-.14x + .26y + .57	.25x + .22y - .04
13%	.17x - .21y + .41	.22x + .18y + .09
70%	.78x + .03y + .11	-.03x + .74y + .27



- Q. What does computation tell us about nature?
Q. What does nature tell us about computation?

20th century sciences. Formulas.

21st century sciences. Algorithms?

Animation

Animation loop. Repeat the following:

- Move object.
- Draw object.
- Display and pause for a short while.

Ex. Bouncing ball.

- Ball has position (r_x , r_y) and constant velocity (v_x , v_y).
- Detect collision with wall and reverse velocity.

21

22

Bouncing Ball

```
public class BouncingBall {
    public static void main(String[] args) {
        double rx = .480, ry = .860;           position
        double vx = .015, vy = .023;           constant velocity
        double radius = .05;                   radius

        StdDraw.setXscale(-1.0, +1.0);         rescale coordinates
        StdDraw.setYscale(-1.0, +1.0);

        while(true) {
            if (Math.abs(rx + vx) > 1.0) vx = -vx;
            if (Math.abs(ry + vy) > 1.0) vy = -vy;    bounce

            rx = rx + vx;
            ry = ry + vy;                            update position

            StdDraw.clear(StdDraw.GRAY);             clear background
            StdDraw.setPenColor(StdDraw.BLACK);
            StdDraw.filledCircle(rx, ry, radius);     draw the ball
            StdDraw.show(50);                         display and pause for 50ms
        }
    }
}
```

23

Special Effects

Images. Put .gif, .png, or .jpg file in the working directory and use `StdDraw.picture()` to draw it.

Sound effects. Put .wav, .mid, or .au file in the working directory and use `StdAudio.play()` to play it.

Ex. Modify `BouncingBall` to display image and play sound upon collision.

- Replace `StdDraw.filledCircle()` with:

```
StdDraw.picture(rx, ry, "earth.gif");
```

- Add following code when collision detected:

```
StdAudio.play("boing.wav");
```

24

User Interfaces

Command line interface.

- User types commands at terminal.
- Easily customizable.
- Extends to complex command sequences.

Point and click.

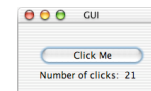
- User launches applications by clicking.
 - File → Open → HelloWorld.java
- Restricted to pre-packaged menu options.



Swing Graphical User Interface

"Swing" is Java's GUI.

- Buttons.
- Menus.
- Scrollbars.
- Toolbars.
- File choosers.



```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class GUI implements ActionListener {
    private int clicks = 0;
    private JFrame frame = new JFrame();
    private JLabel label = new JLabel("Number of clicks: 0");
    public GUI() {
        JButton button = new JButton("Click Me");
        button.addActionListener(this);
        JPanel panel = new JPanel();
        panel.setBorder(BorderFactory.createEmptyBorder(30, 30, 10, 30));
        panel.setLayout(new GridLayout(0, 1));
        panel.add(button);
        panel.add(label);
        frame.add(panel, BorderLayout.CENTER);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setTitle("GUI");
        frame.pack();
        frame.show();
    }

    public void actionPerformed(ActionEvent e) {
        clicks++;
        label.setText("Number of clicks: " + clicks);
    }

    public static void main(String[] args) {
        GUI gui = new GUI();
    }
}
```

a sample Swing application

Ignore details.

25

26