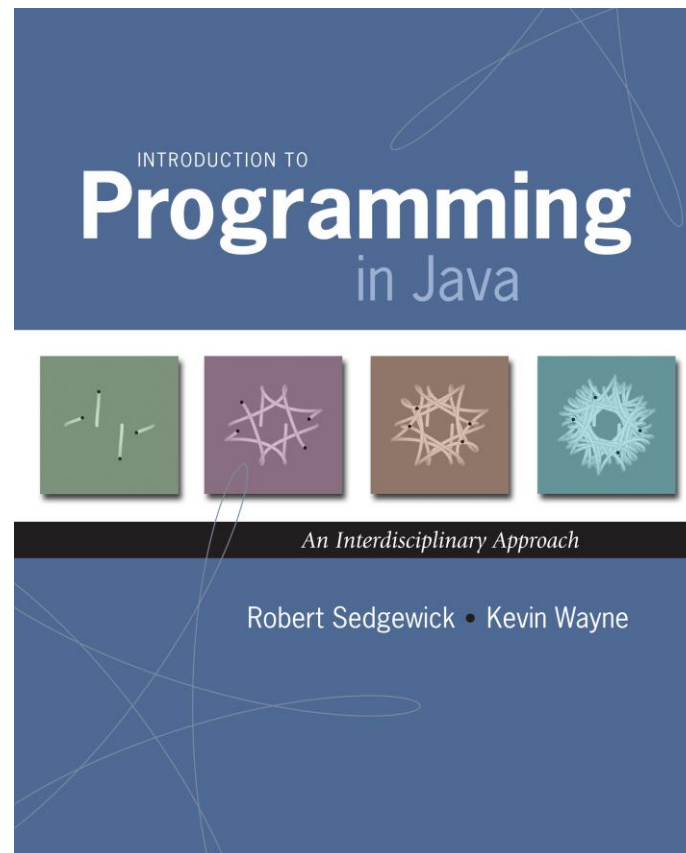


1.5 Input and Output



Input and Output

Input devices



Keyboard



Mouse



Hard drive



Network



Digital camera



Microphone

Output devices



Display



Speakers



Hard drive



Network



Printer



MP3 Player

Goal Java programs that interact with the outside world

Input and Output

Input devices



Keyboard



Mouse



Hard drive



Network



Digital camera



Microphone

Output devices



Display



Speakers



Hard drive



Network



Printer



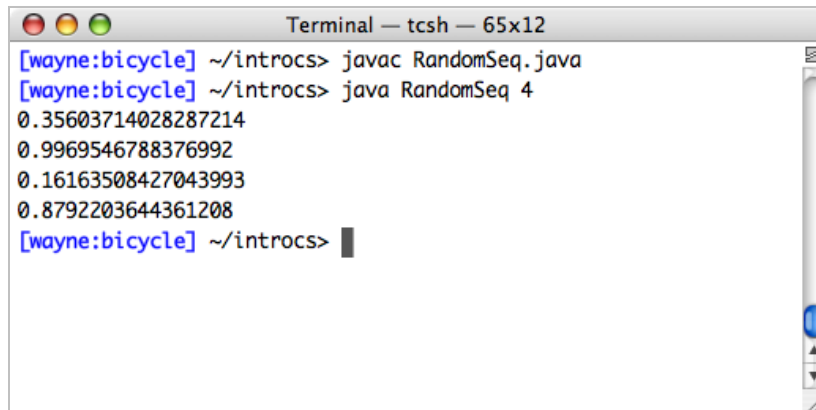
MP3 Player

Our approach

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

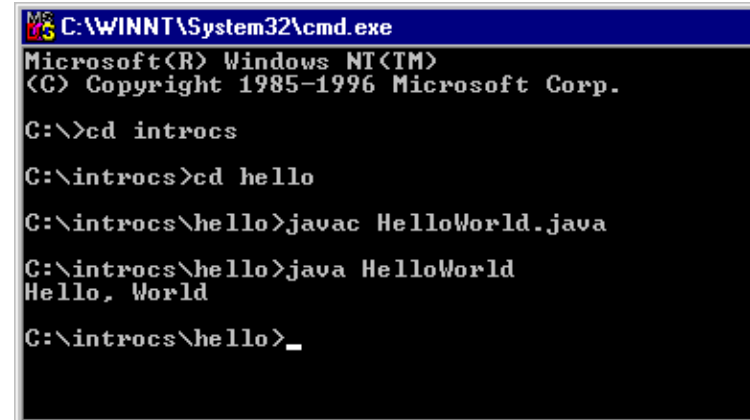
Terminal

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



```
Terminal — tcsh — 65x12
[wayne:bicycle] ~/introc> javac RandomSeq.java
[wayne:bicycle] ~/introc> java RandomSeq 4
0.35603714028287214
0.9969546788376992
0.16163508427043993
0.8792203644361208
[wayne:bicycle] ~/introc> █
```

Mac OS X



```
C:\WINNT\System32\cmd.exe
Microsoft(R) Windows NT(TM)
(C) Copyright 1985-1996 Microsoft Corp.

C:\>cd introcs
C:\introcs>cd hello
C:\introcs\hello>javac HelloWorld.java
C:\introcs\hello>java HelloWorld
Hello, World
C:\introcs\hello>_
```

Microsoft Windows

Command-Line Input and Standard Output

Command-line input. Read an integer N as command-line argument.

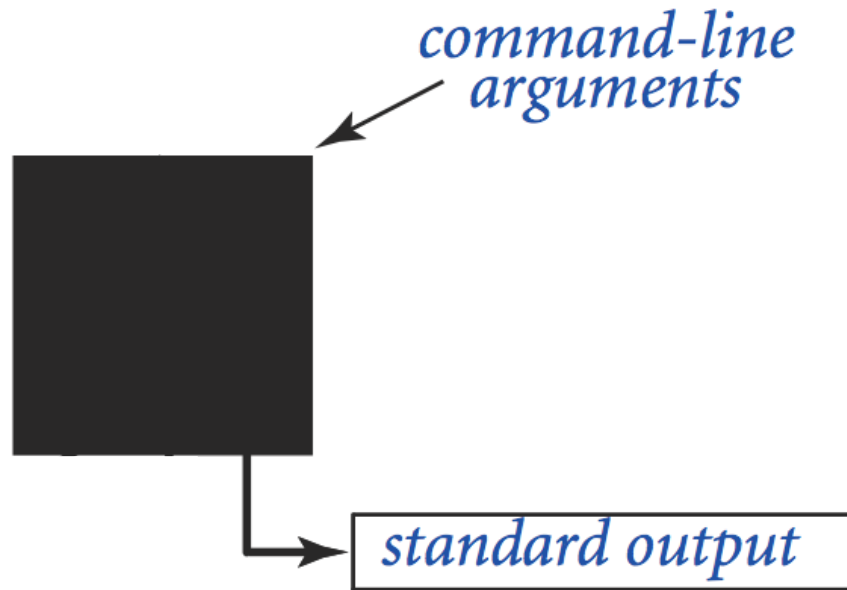
Standard output.

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

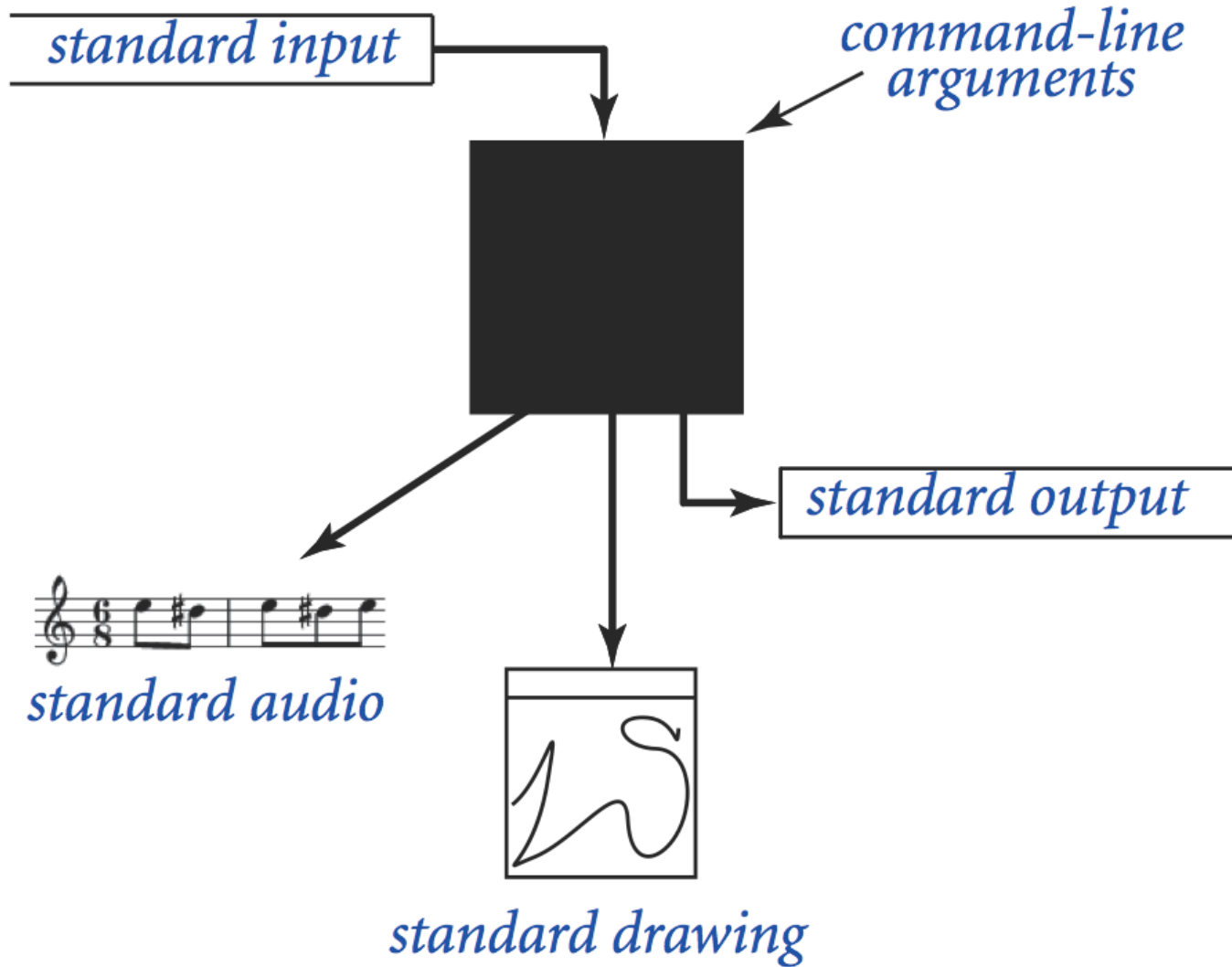
```
public class RandomSeq {  
    public static void main(String[] args) {  
        int N = Integer.parseInt(args[0]);  
        for (int i = 0; i < N; i++) {  
            System.out.println(Math.random());  
        }  
    }  
}
```

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

Old Bird's Eye View



New Bird's Eye View



Standard Input and Output

Command-Line Input vs. Standard Input

Command-line input.

- Use command-line input 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, standard input is received from Terminal window.
- Input entered **while** program is executing.

Standard Input and Output

Standard input `stdIn` is library for reading text input

Standard output `stdOut` is library for writing text output

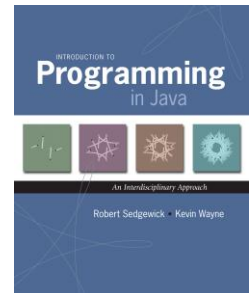
```
public class StdIn
```

<code>boolean isEmpty()</code>	<i>true if no more values, false otherwise</i>
<code>int readInt()</code>	<i>read a value of type int</i>
<code>double readDouble()</code>	<i>read a value of type double</i>
<code>long readLong()</code>	<i>read a value of type long</i>
<code>boolean readBoolean()</code>	<i>read a value of type boolean</i>
<code>char readChar()</code>	<i>read a value of type char</i>
<code>String readString()</code>	<i>read a value of type String</i>
<code>String readLine()</code>	<i>read the rest of the line</i>
<code>String readAll()</code>	<i>read the rest of the text</i>

```
public class StdOut
```

<code>void print(String s)</code>	<i>print s</i>
<code>void println(String s)</code>	<i>print s, followed by newline</i>
<code>void println()</code>	<i>print a new line</i>
<code>void printf(String f, ...)</code>	<i>formatted print</i>

libraries developed
for this course
(also broadly useful)



Standard Input and Output

To use Download `stdin.java` and `stdout.java` from booksite, and put in working directory (or use classpath)

see booksite



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

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

Averaging A Stream of Numbers

Average Read in a stream of numbers, and print their average

```
public class Average {
    public static void main(String[] args) {
        double sum = 0.0; // cumulative total
        int n = 0; // number of values

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

        StdOut.println(sum / n);
    }
}
```

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

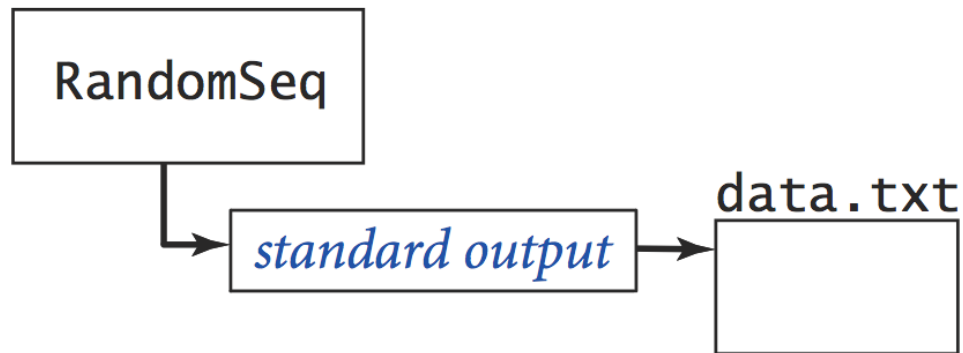
<Ctrl-d> for OS X/Linux/Unix/DrJava
<Ctrl-z> for Windows

Key point Program does not limit the amount of data

Redirection and Piping

Redirecting Standard Output

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

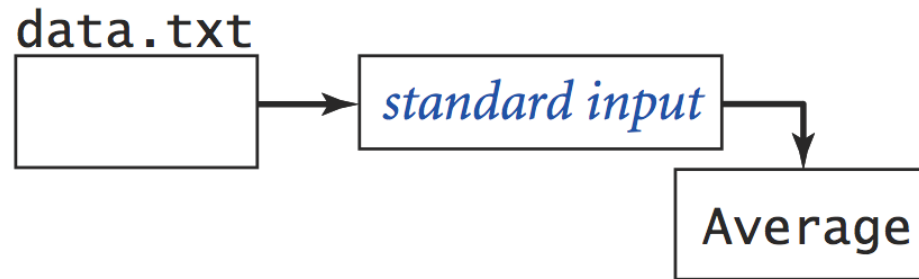


```
% java RandomSeq 1000 > data.txt
```

redirect stdout

Redirecting Standard Input

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

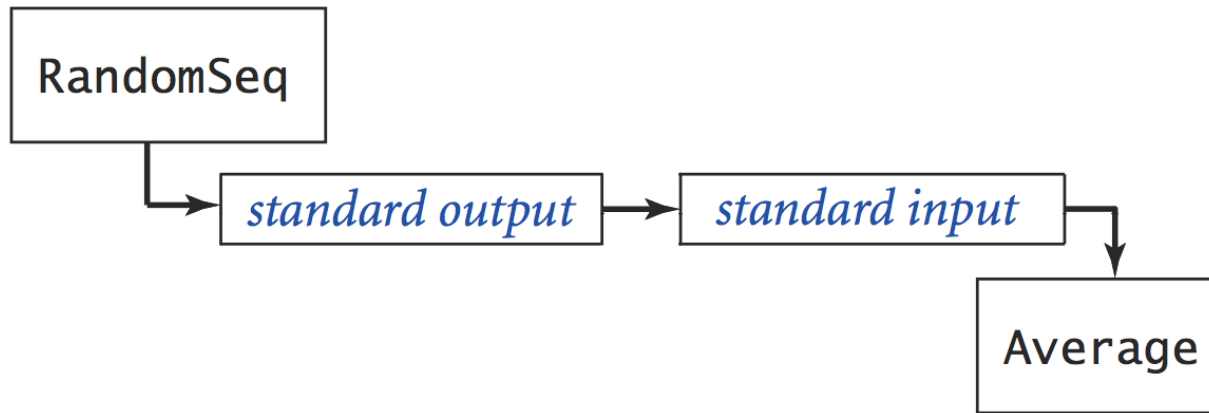


```
% more < data.txt
0.5475375782884312
0.4971087292684019
0.23123808041753813
...
% java Average < data.txt
0.4947655567740991
```

← redirect stdin

Connecting Programs

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



pipe stdout of RandomSeq
to stdin of Average

```
% java RandomSeq 1000000 | java Average  
0.4997970473016028  
  
% java RandomSeq 1000000 | java Average  
0.5002071875644842
```


Redirecting Standard Output to a Toast Printer

```
% java HelloWorld > /dev/toaster
```



Standard Drawing

Standard Drawing

Standard drawing `stdDraw` is library for producing graphical output

```
public class StdDraw
```

```
void line(double x0, double y0, double x1, double y1)
```

```
void point(double x, double y)
```

```
void text(double x, double y, String s)
```

```
void circle(double x, double y, double r)
```

```
void filledCircle(double x, double y, double r)
```

```
void square(double x, double y, double r)
```

```
void filledSquare(double x, double y, double r)
```

```
void polygon(double[] x, double[] y)
```

```
void filledPolygon(double[] x, double[] y)
```

```
void setXscale(double x0, double x1)
```

reset x range to (x_0, x_1)

```
void setYscale(double y0, double y1)
```

reset y range to (y_0, y_1)

```
void setPenRadius(double r)
```

set pen radius to r

```
void setPenColor(Color c)
```

set pen color to c

```
void setFont(Font f)
```

set text font to f

```
void setCanvasSize(int w, int h)
```

set canvas to w-by-h window

```
void clear(Color c)
```

clear the canvas; color it c

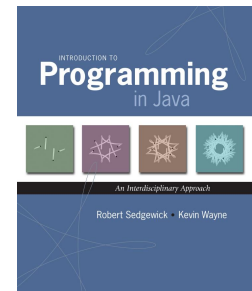
```
void show(int dt)
```

show all; pause dt milliseconds

```
void save(String filename)
```

save to a .jpg or w.png file

library developed
for this course
(also broadly useful)



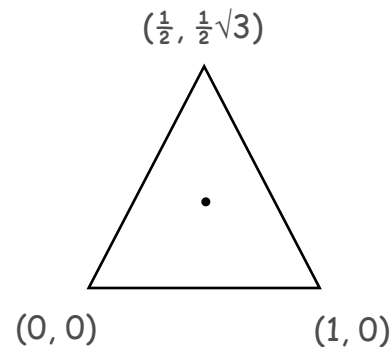
Note: Methods with the same names but no arguments reset to default values.

Standard Draw

Standard drawing 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
```



Data Visualization

Plot filter Read in a sequence of (x, y) coordinates from standard input, and plot using standard drawing

```
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

Data Visualization

```
% more < USA.txt
```

```
669905.0 247205.0 1244962.0 490000.0
```

```
1097038.8890 245552.7780
```

```
1103961.1110 247133.3330
```

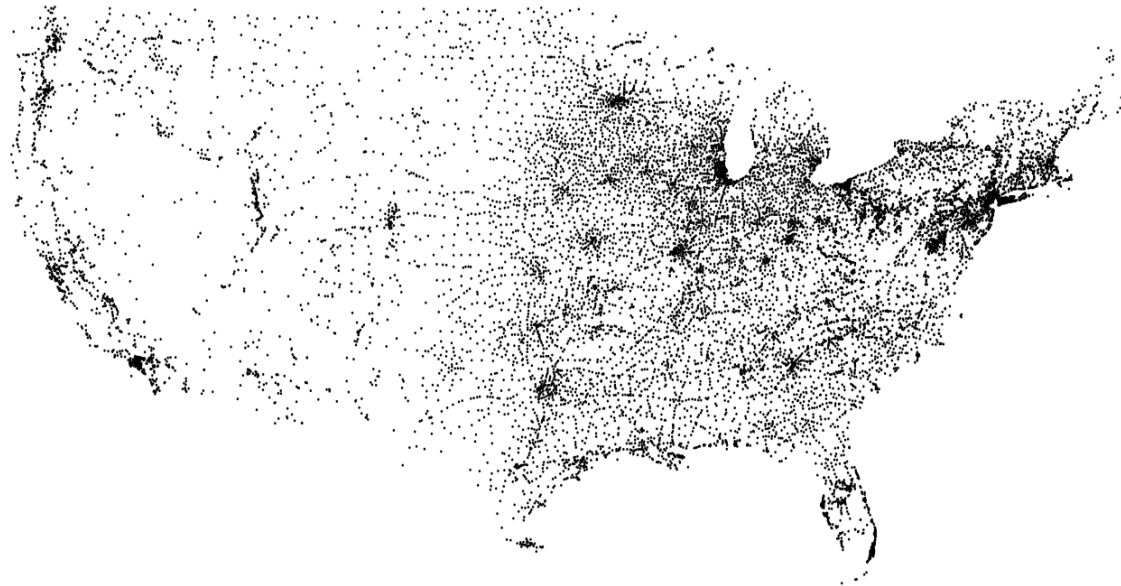
```
1104677.7780 247205.5560
```

```
...
```

```
% java PlotFilter < USA.txt
```

bounding box

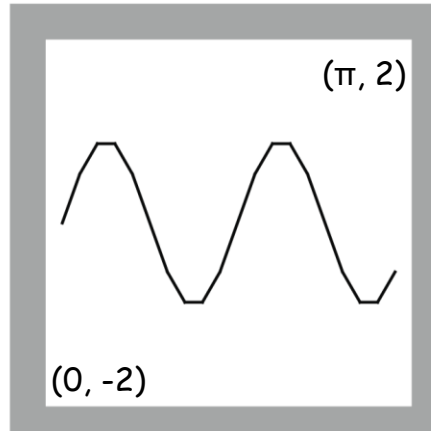
coordinates of
13,509 US cities



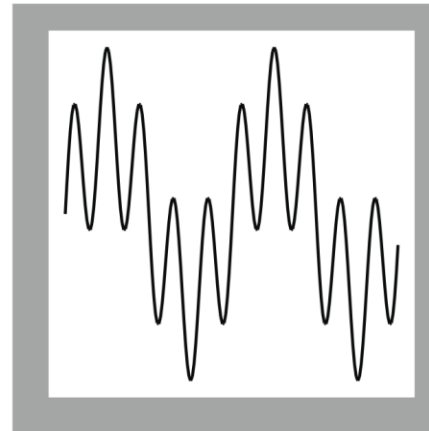
Plotting a Function

```
double[] x = new double[N+1];
double[] y = new double[N+1];
for (int i = 0; i <= N; i++) {
    x[i] = Math.PI * i / N;
    y[i] = Math.sin(4*x[i]) + Math.sin(20*x[i]);
}
StdDraw.setXscale(0, Math.PI);
StdDraw.setYscale(-2.0, +2.0);
for (int i = 0; i < N; i++)
    StdDraw.line(x[i], y[i], x[i+1], y[i+1]);
```

$N = 20$



$N = 200$



$$y = \sin 4x + \sin 20x, x \in [0, \pi]$$

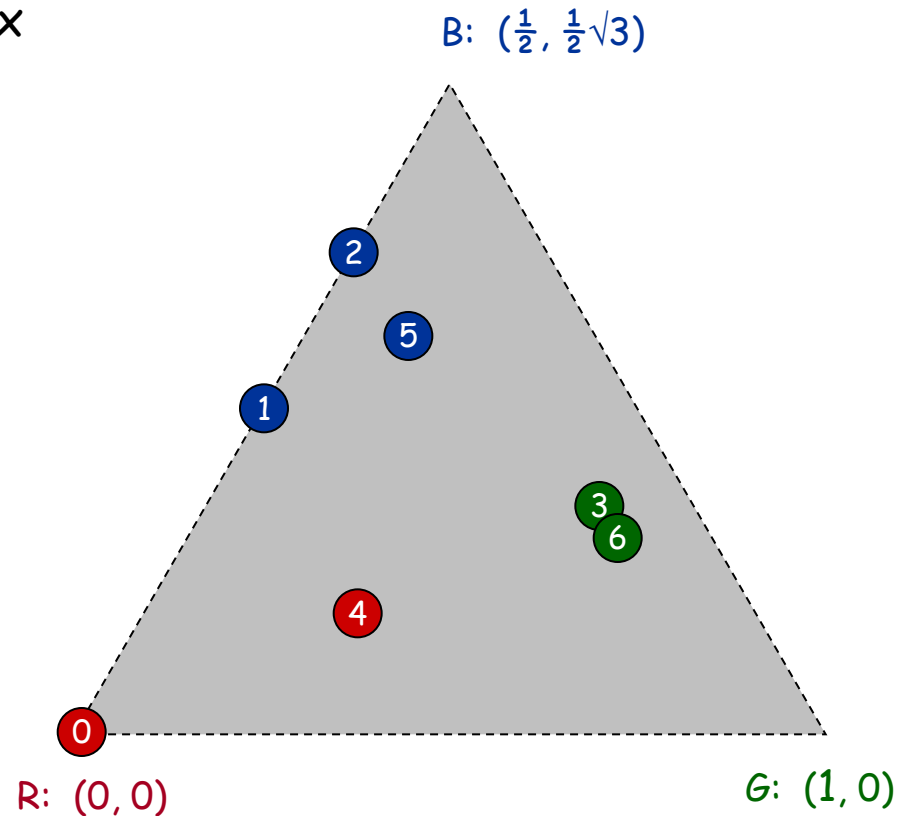
Chaos Game

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?

B B G R B G ...



Chaos Game

```
public class Chaos {  
    public static void main(String[] args) {  
        int T = 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 t = 0; t < T; t++) {  
            int r = (int) (Math.random() * 3);  
            x = (x + cx[r]) / 2.0;  
            y = (y + cy[r]) / 2.0;  
            StdDraw.point(x, y);  
        }  
    }  
}
```

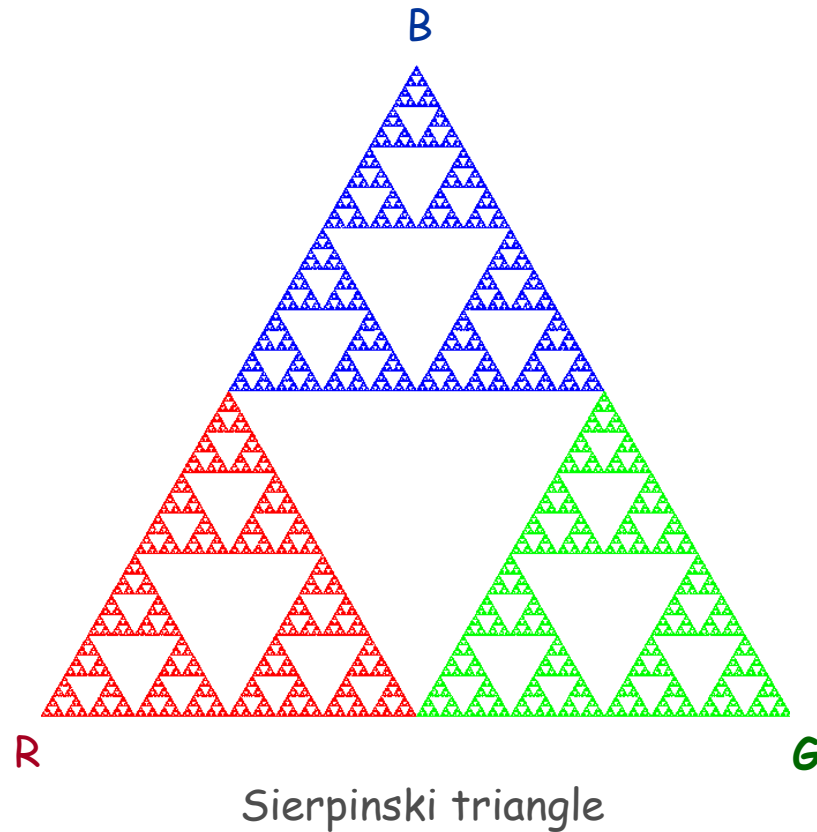
$\frac{1}{2}\sqrt{3}$
(avoid hardwired
constants like this)

between 0 and 2

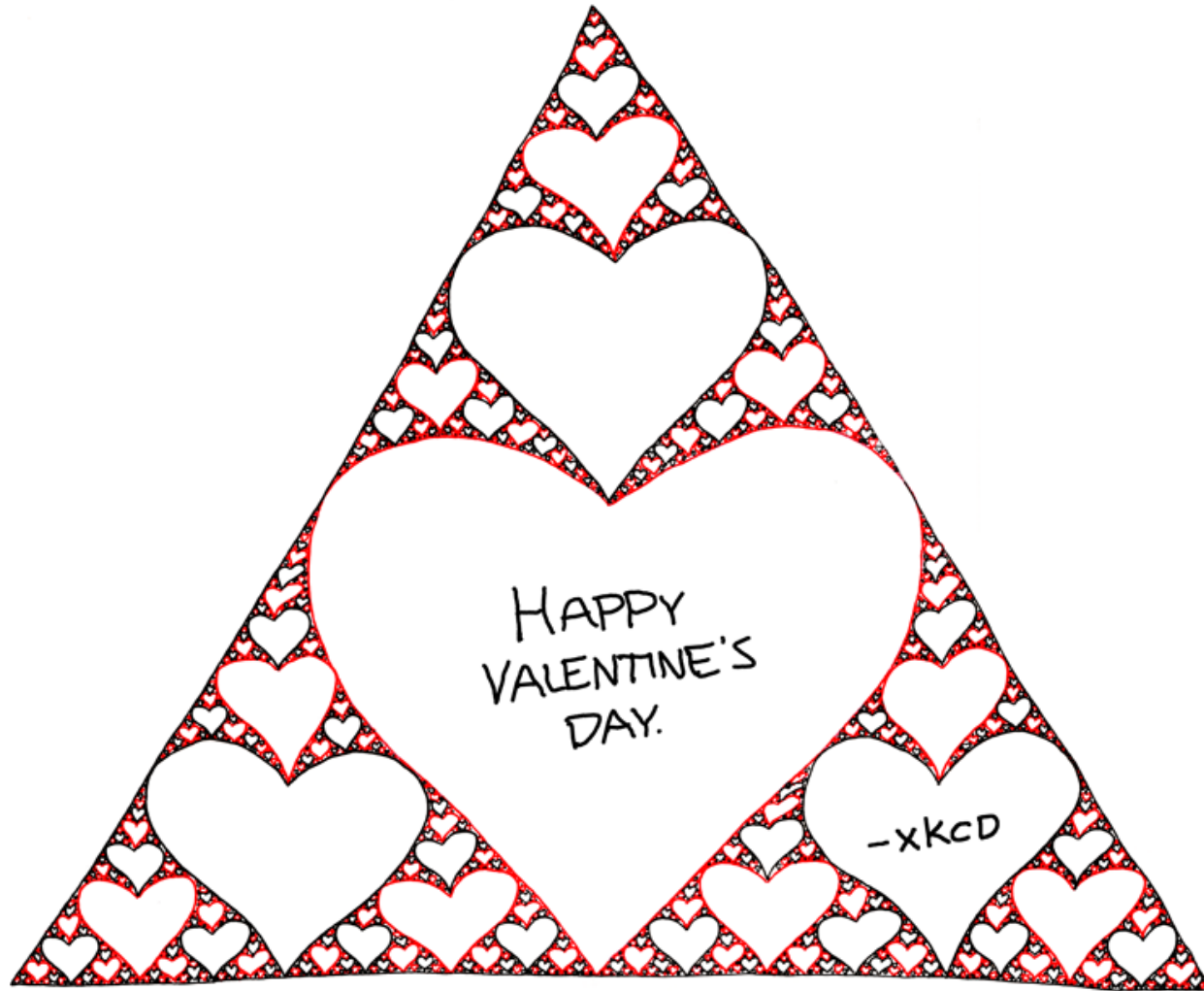
Chaos Game

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

```
% java Chaos 10000
```



Commercial Break



<http://xkcd.com/543>

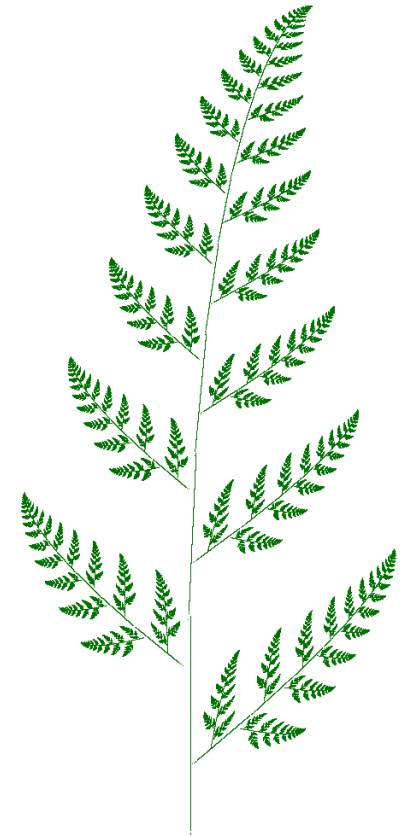
Commercial Break



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?

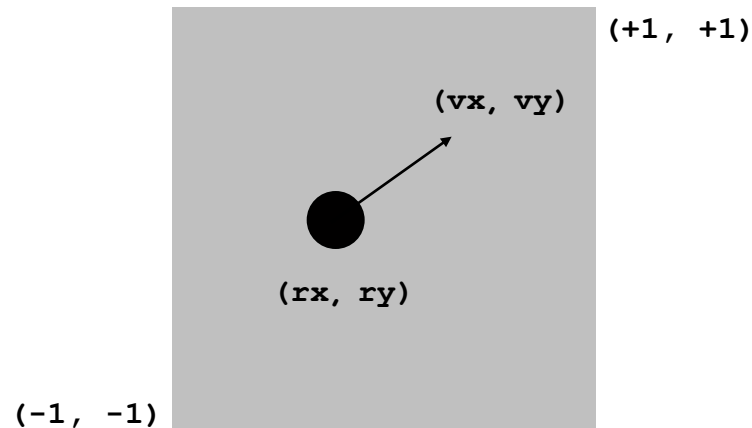
Animation

Animation loop Repeat the following:

- Clear the screen
- Move the object
- Draw the object
- Display and pause for a short while

Ex. Bouncing ball

- Ball has position (rx, ry) and constant velocity (vx, vy)
- Detect collision with wall and reverse velocity



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);
```

bounce

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

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

```
            StdDraw.setPenColor(StdDraw.GRAY); clear background  
            StdDraw.filledSquare(0.0, 0.0, 1.0);  
            StdDraw.setPenColor(StdDraw.BLACK); draw the ball  
            StdDraw.filledCircle(rx, ry, radius);  
            StdDraw.show(20);
```

```
        }
```

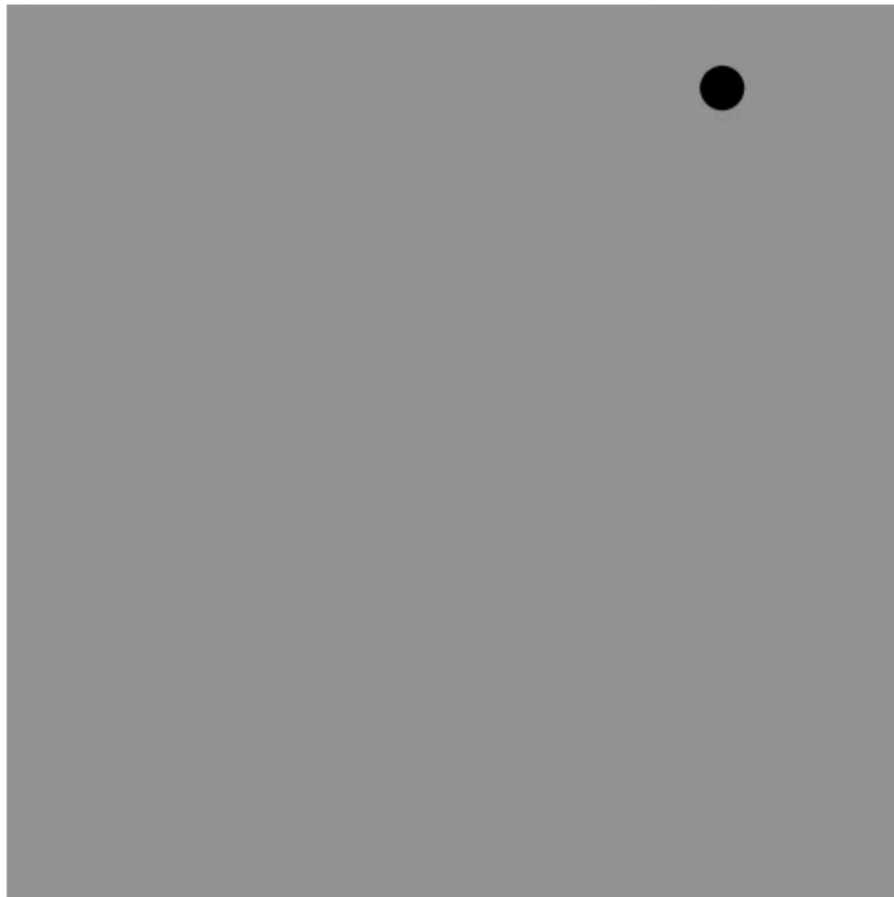
```
    }
```

```
}
```

turn on animation mode:
display and pause for 20ms

Bouncing Ball Demo

```
% java BouncingBall
```



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");
```



earth.gif

- Add following code upon collision with vertical wall:

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



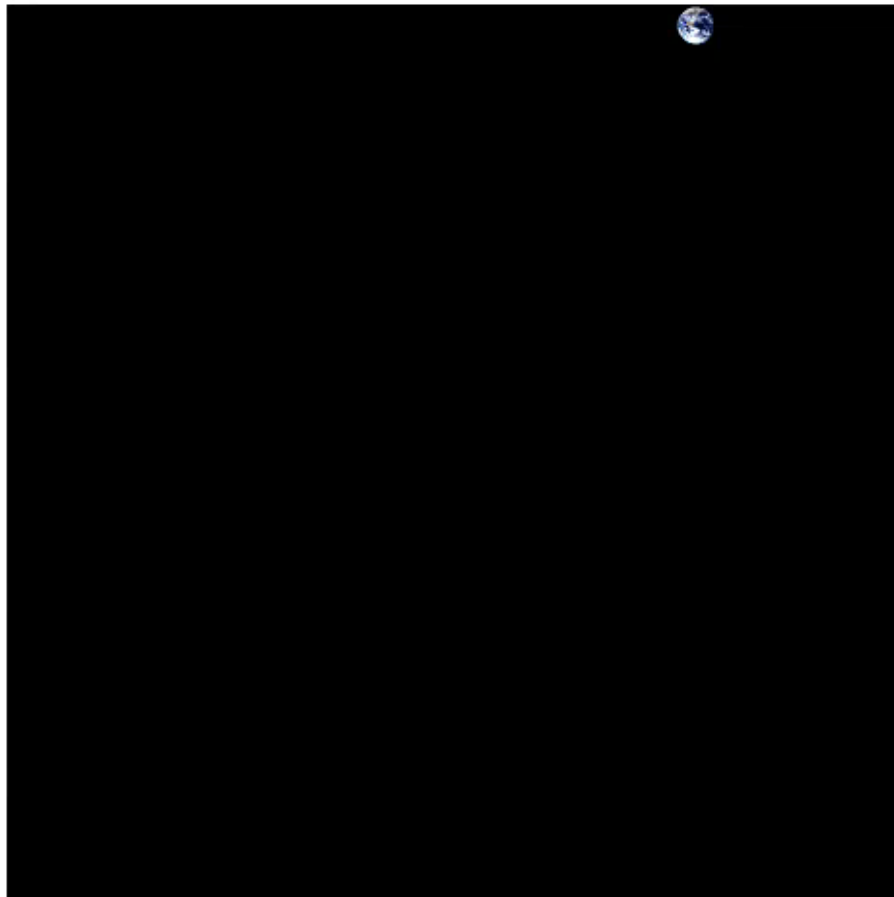
laser.wav



pop.wav

Deluxe Bouncing Ball Demo

```
% java DeluxeBouncingBall
```



Bouncing Ball Challenge

Q. What happens if you call `stdDraw.filledSquare()` once before loop (instead of inside)?

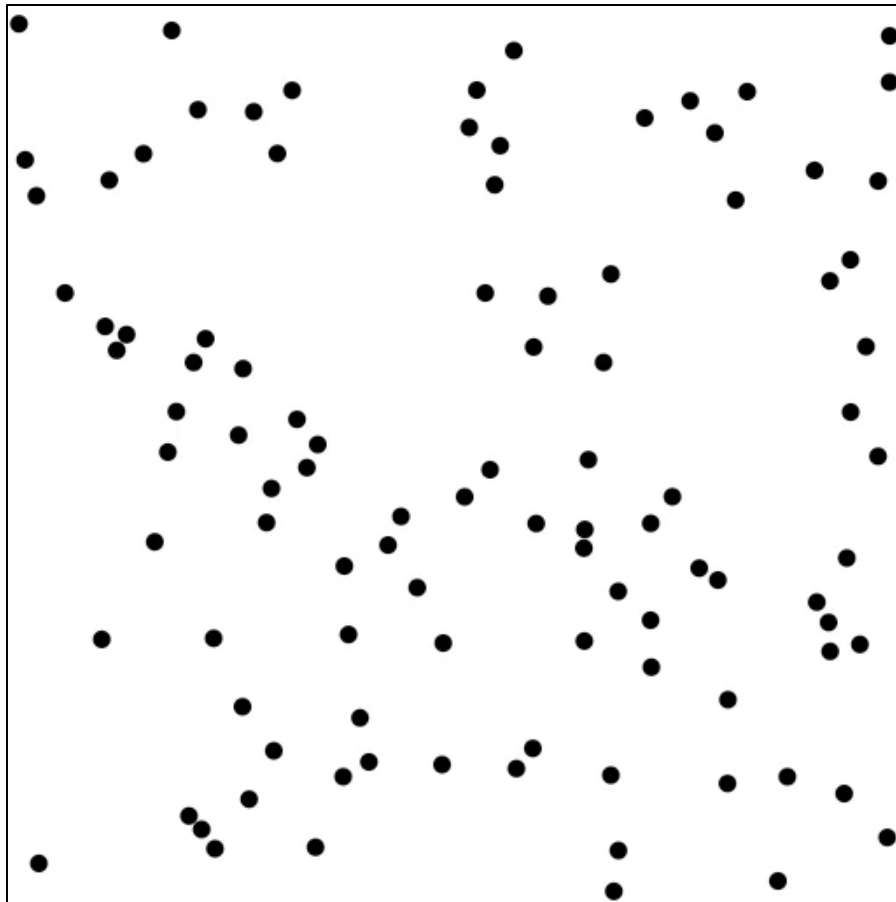
```
% java DeluxeBouncingBall
```



Colliding Balls

Challenge Add elastic collisions

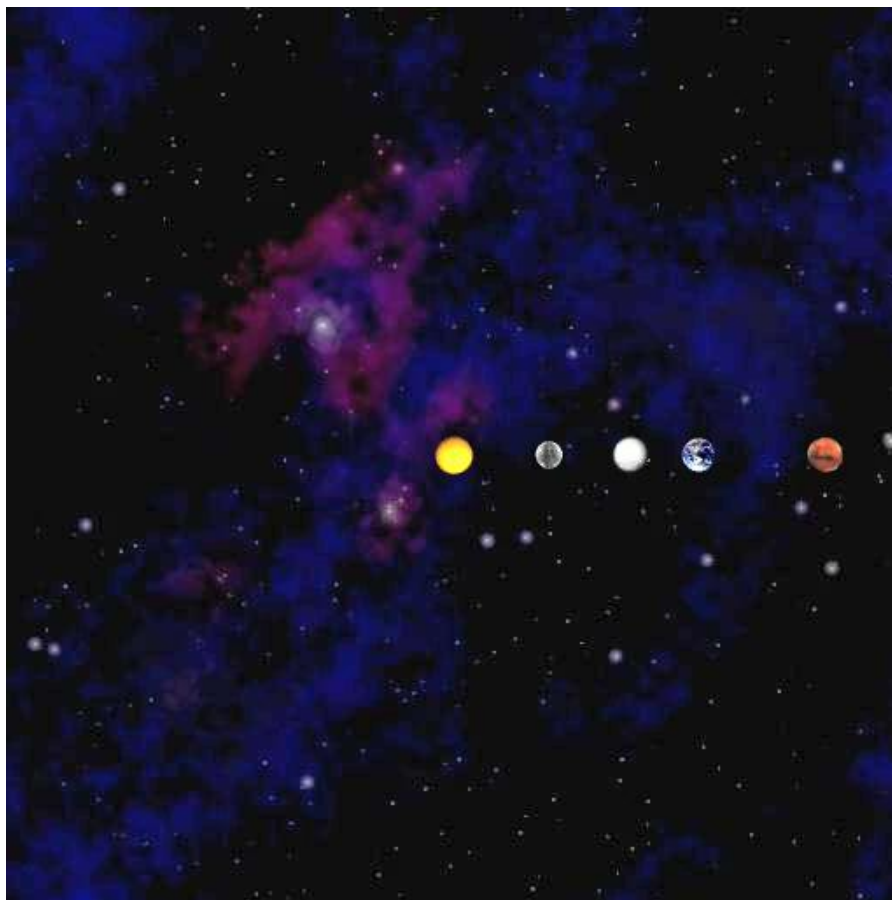
```
% java CollidingBalls 100
```



N-body Simulation

Challenge Add gravity

```
% java NBody < planets.txt
```



Digital Audio in Java

Standard audio Library for playing digital audio

```
public class StdAudio
```

```
void play(String file)
```

play the given .wav file

```
void play(double[] a)
```

play the given sound wave

```
void play(double x)
```

play sample for 1/44100 second

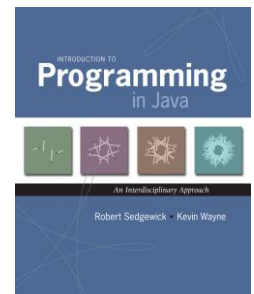
```
void save(String file, double[] a)
```

save to a .wav file

```
double[] read(String file)
```

read from a .wav file

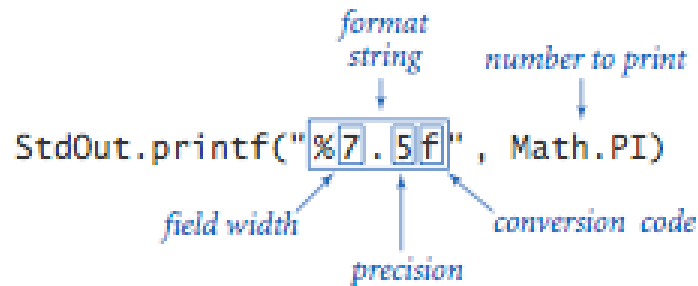
library developed
for this course
(also broadly useful)



Formatted Output

StdOut.printf()

- Print complex combinations of text and variables easily
- Use format string with placeholders for variables
- Placeholders specify variable type and output format



Anatomy of a formatted print statement

Formatted Output

StdOut.printf()

- Print complex combinations of text and variables easily
- Use format string with placeholders for variables
- Placeholders specify variable type and output format

<i>type</i>	<i>code</i>	<i>typical literal</i>	<i>sample format strings</i>	<i>converted string values for output</i>
int	d	512	"%14d" "%-14d"	" 512" "512"
double	f e	1595.1680010754388	"%14.2f" "% .7f" "%14.4e"	" 1595.17" "1595.1680011" " 1.5952e+03"
String	s	"Hello, World"	"%14s" "%-14s" "%-14.5s"	" Hello, World" "Hello, World " "Hello "

Formatted Output

Print planet positions in NBody simulation

`\n` means print a new line

`%11.4e` means print a double in scientific notation using at most 11 characters of which four are decimal places

```
StdOut.printf("%d\n", N);
StdOut.printf("%.2e\n", R)
for (int i = 0; i < N; i++) {
    StdOut.printf("%11.4e %11.4e %11.4e %11.4e %11.4e %12s\n",
                  rx[i], ry[i], vx[i], vy[i], mass[i], image[i]);
}
```

```
5
2.50e+11
1.4960e+11 0.0000e+00 0.0000e+00 2.9800e+04 5.9740e+24 earth.gif
2.2790e+11 0.0000e+00 0.0000e+00 2.4100e+04 6.4190e+23 mars.gif
5.7900e+10 0.0000e+00 0.0000e+00 4.7900e+04 3.3020e+23 mercury.gif
0.0000e+00 0.0000e+00 0.0000e+00 0.0000e+00 1.9890e+30 sun.gif
1.0820e+11 0.0000e+00 0.0000e+00 3.5000e+04 4.8690e+24 venus.gif
```