T-301-REIR, fall 2018

Reiknirit/Algorithms

F1 Introduction

- Course organization
- Why study algorithms
- Java & textbook libraries

Magnús M. Halldórsson

HÁSKÓLINN Í REYKJAVÍK
REYKJAVIK UNIVERSITY

V101 & V102

What is T-301?

- Programming and problem solving, with applications.
  - "How to solve things with a computer"

- Intermediate-level survey course on algorithms and data structures

topic	data structures and algorithms
data types	stack, union-find, heaps, tries
sorting	quicksort, mergesort, heapsort, radix sorts
searching	hash table, BST, red-black tree
graphs	BFS, DFS, Prim, Kruskal, ~~Dijkstra~~
strings	TST, Huffman, LZW, ~~KMP, Boyer-Moore~~
reductions	

Comparison with Gagnaskipan

Common material

- Stacks, Priority queues, Binary search trees, Hash tables
- Sorting (selection sort, merge sort), Big-oh notation
- Programming fundamentals

Differences

- Focus here on efficiency: time complexity and memory use
- Use mathematics for analyzing algorithms
- Builds on Gagnaskipan and goes deeper/more intensive
- More abstraction (separate from concrete implementations)
- Java

Prerequisites

- Strjál stærðfræði + Gagnaskipan
- Contact instructor for exceptions

Skipulag (Organization)

Canvas : Allar helstu upplýsingar : *Syllabus, Teaching Methods, Assessment*

Upptökur

- Fyrirlestrum er streymt á Google Hangouts (sem geymast á YouTube)
- Skoða má eldri upptökur (`rucomputerscience` notandi)

Umræðuþræðir

- Spurningar/svör: *Piazza*

Fjarnemar: Sendið mér póst í þessari viku sem lýsir ykkar stöðu

Non-native speakers: Send me mail ASAP, describing your language level

Quizzes during class

- Socrative.com
- Canvas

Örpásur

Assessment (Verkefni og einkunnagjöf)

Forritunarverkefni 29 %

- 1 einstaklings, 2 (af 3) hópa, 8% hvert + 5% skýrsla fyrir S1
- Skilafrestir: sunnudögum kl. 24
- Einn heill dreginn frá fyrir hvern (hámark 2) sólarhring framyfir

Minni verkefni 6%

- 7 sett fyrir, 6 bestu gilda 1% hvert. Staðið/Fallið yfirferð.
- Þær vikur sem ekki eru stór skil

Æfingar 11%

- Ent.ru.is : Þjónn sem framleiðir æfingar úr efninu
- Getið endurtekið eins oft eins og þörf er á

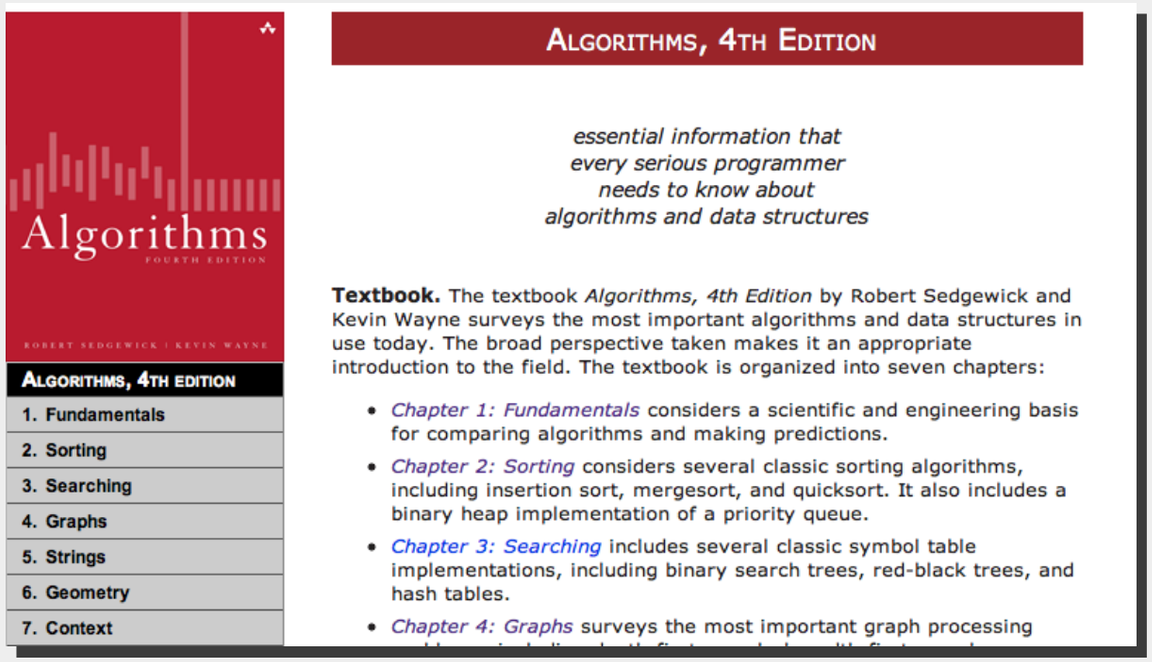
Lokapróf 56%

- Engin gögn leyfð, nema eitt (eigið) A4 blað, skrifað öðru megin

Textbook and Website

Required reading.

- Algorithms 4th edition, Sedgewick & Wayne



Website.

- Exercises.
- Lecture slides.
- Programming assignments.
- Brief summary of content.
- Download code from lecture.

- Libraries: algs4.jar

http://www.cs.princeton.edu/algs4

Coming up

D0 (3%): Due this Sunday, 19 Aug

Install Java and experiment with the course libraries

D1 (3%): Due Sunday 26 Aug

Stacks

Time complexity

S1 (8%): Due Sunday 1 Sep

Report for S1 (5%): Due Sunday 8 Sep

Dæmatímar verða haldnir í þessari viku

Introduction

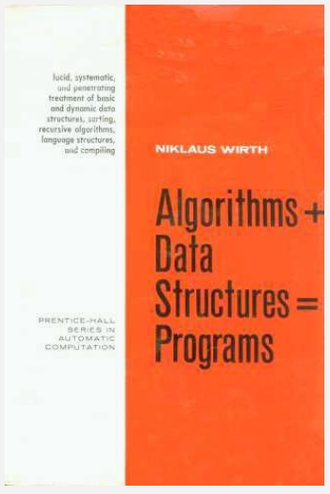
- Skipulag námskeiðs
- Hví læra um reiknirit?
- Um Java

Why study algorithms?

To become a proficient programmer.

> *" I will, in fact, claim that the difference between a bad programmer and a good one is whether he considers his code or his data structures more important. Bad programmers worry about the code. Good programmers worry about data structures and their relationships. "*
>
> *— Linus Torvalds (creator of Linux)*

> *" Algorithms + Data Structures = Programs. "* *— Niklaus Wirth*



Why study algorithms?

For fun and profit.

Why study algorithms?

Their impact is broad and far-reaching.

Internet. Web search, packet routing, distributed file sharing, ...

Biology. Human genome project, protein folding, ...

Computers. Circuit layout, file system, compilers, ...

Computer graphics. Movies, video games, virtual reality, ...

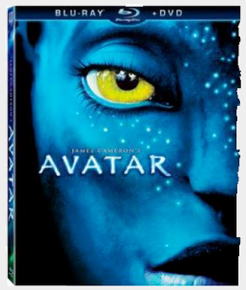
Security. Cell phones, e-commerce, voting machines, ...

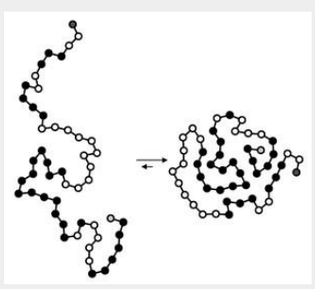
Multimedia. CD player, DVD, MP3, JPG, DivX, HDTV, ...

Transportation. Airline crew scheduling, map routing, ...

Physics. N-body simulation, particle collision simulation, ...

...

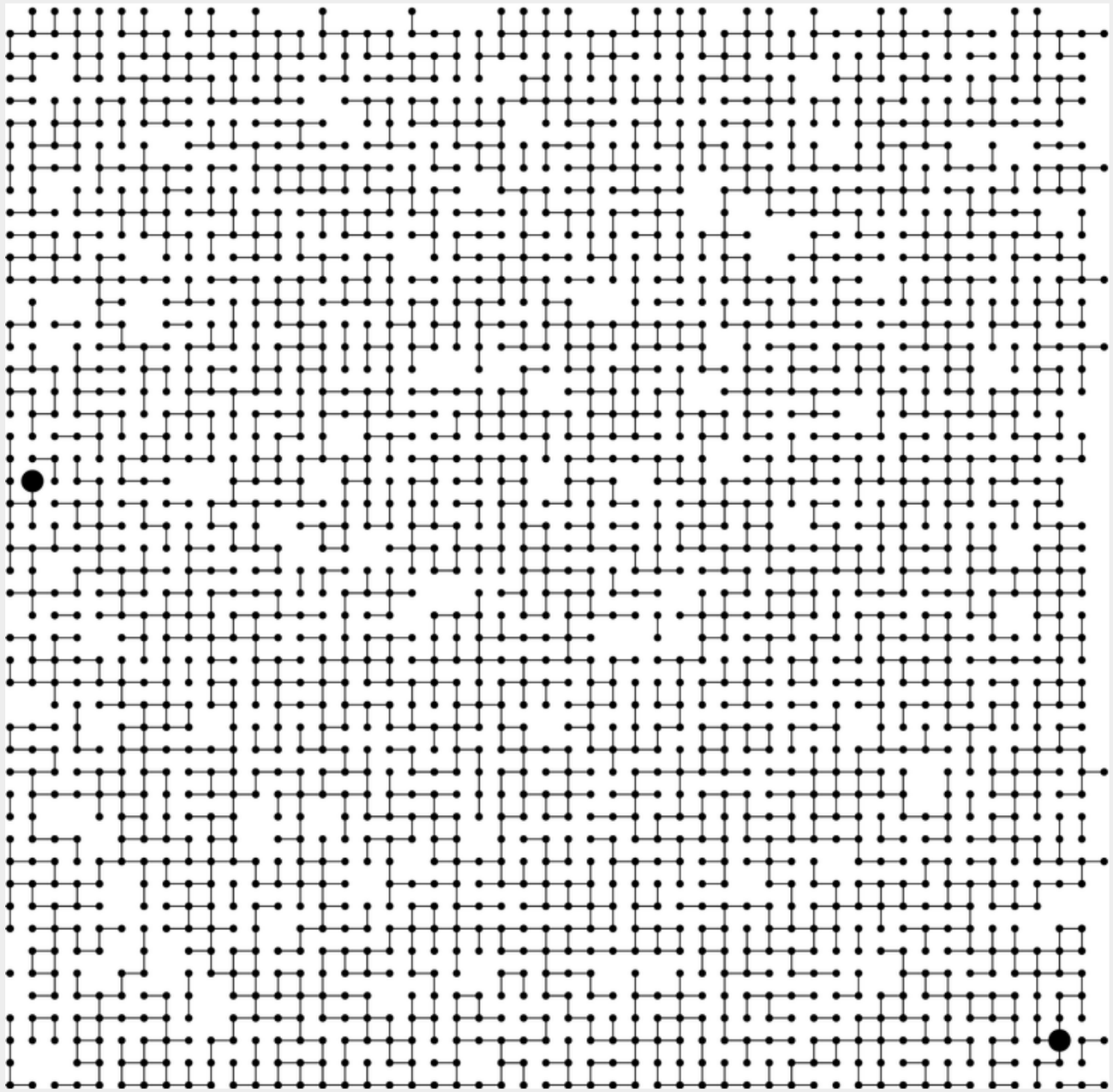


Why study algorithms?

To solve problems that could not otherwise be addressed.

Ex. Network connectivity. [stay tuned]



Why study algorithms?

They may unlock the secrets of life and of the universe.

Computational models are replacing mathematical models in scientific inquiry.

$$E = mc^2$$

$$F = ma \qquad F = \frac{Gm_1m_2}{r^2}$$

$$\left[-\frac{\hbar^2}{2m}\nabla^2 + V(r)\right]\Psi(r) = E\,\Psi(r)$$

20th century science

(formula based)

```
for (double t = 0.0; true; t = t + dt)
   for (int i = 0; i < N; i++)
   {
      bodies[i].resetForce();
      for (int j = 0; j < N; j++)
         if (i != j)
            bodies[i].addForce(bodies[j]);
}
```

21st century science

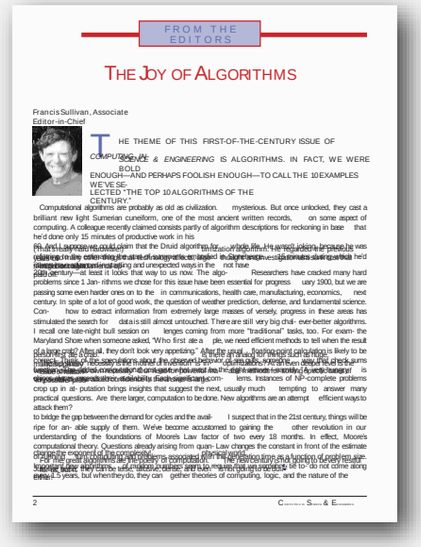
(algorithm based)

" Algorithms: a common language for nature, human, and computer. " — *A. Wigderson*

Why study algorithms?

For intellectual stimulation.

> *" For me, great algorithms are the poetry of computation. Just like verse, they can be terse, allusive, dense, and even mysterious. But once unlocked, they cast a brilliant new light on some aspect of computing. "* — *Francis Sullivan*



FROM THE EDITORS

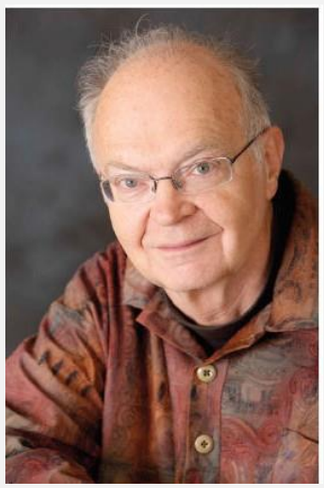
THE JOY OF ALGORITHMS

Francis Sullivan, Associate Editor-in-Chief

THE THEME OF THIS FIRST-OF-THE-CENTURY ISSUE OF *COMPUTING IN SCIENCE & ENGINEERING* IS ALGORITHMS. IN FACT, WE WERE BOLD ENOUGH—AND PERHAPS FOOLISH ENOUGH—TO CALL THE 10 EXAMPLES WE'VE SELECTED "THE TOP 10 ALGORITHMS OF THE CENTURY."

Computational algorithms are probably as old as civilization. mysterious. But once unlocked, they cast a brilliant new light Sumerian cuneiform, one of the most ancient written records, on some aspect of computing. A colleague recently claimed consists partly of algorithm descriptions for reckoning in base that he'd done only 15 minutes of productive work in his

2 C..... S.... & E.....



> *" An algorithm must be seen to be believed. "* — *Donald Knuth*

Introduction

- **Studying: Do‘s and don‘ts**

Study habits

Consult the material early

- Skim before class
- Quickly review soon after class

Read the assignments as soon as available

- What you don‘t know/understand will guide your reading/study
- „Good cooking takes time“
- Avoid the FIFO (or EDF) trap

Be active

- Multitasking reduces attention
- Be proactive : Ask yourself: What don‘t I understand? How do I apply this?
- Dæmatímar are for you to get more customized, in-depth explanations

Use people resources

- Instructor available before and after class, and at other times
- Discuss together

Þitt framlag

Reglur um verkefnaskil

- Minnt á reglur skólans um verkefnavinnu.
- Allur texti/kóði sem þú skilar skal vera þinn eiginn og ekki sýndur öðrum

Ekki:

- Nýta þér lausnir af netinu, eða annara nemenda
- Samnýta eða samvinna lausnir í stærri hópum
- Dreifa kóða (t.d. á github)
- “Hjálpa” öðrum með því að gefa þeim eða sýna þeim þinn kóða

Í þessu námskeiði er þessum reglum fylgt eftir

- Gerður er sjálfvirkur samanburður

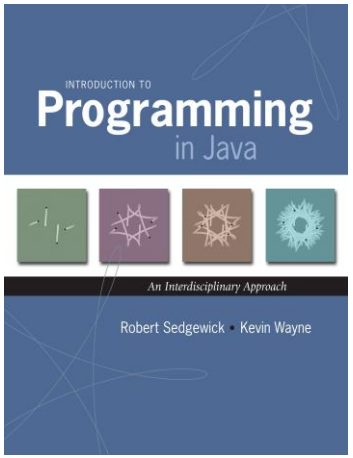
Viðurlög við ritstuld eða aðra misnotkun:

- Núll í verkefni
- Formleg áminning
- Fyrir annað brot: núll í námskeiði

REIR Reiknirit

HÁSKÓLINN Í REYKJAVÍK
REYKJAVIK UNIVERSITY

F1b Java og klasasöfn



Our Choice: Java

Java features.

- Widely used.
- Widely available.
- Embraces full set of modern abstractions.
- Variety of automatic checks for mistakes in programs.

Java economy.

- Mars rover.
- Cell phones.
- Blu-ray Disc.
- Web servers.
- Medical devices.
- Supercomputing.
- ...

$100 billion,
5 million developers

James Gosling
http://java.net/jag

Why Java?

Java features.

- Widely used.
- Widely available.
- Embraces full set of modern abstractions.
- Variety of automatic checks for mistakes in programs.

Facts of life.

- No perfect language.
- We need to choose some language.

> *" There are only two kinds of programming languages: those people always [gripe] about and those nobody uses."*
>
> *– Bjarne Stroustrup*

Our approach.

- Minimal subset of Java.
- Textbook library makes it very similar to C++.
- Develop general programming skills that are applicable to many languages.

It's not about the language!

A Rich Subset of the Java Language

Built-In Types	
`int`	`double`
`long`	***`String`***
`char`	`boolean`

I/O
`System.in.readInt()`
`System.out.print()`
`System.out.printf()`

Math Library	
`Math.sin()`	`Math.cos()`
`Math.log()`	`Math.exp()`
`Math.sqrt()`	`Math.pow()`
`Math.min()`	`Math.max()`
`Math.abs()`	`Math.PI`

Flow Control	
`if`	`else`
`for`	`while`

Parsing
`Integer.parseInt()`
`Double.parseDouble()`

Operators		
`+`	`-`	`*`
`/`	`%`	`++`
`--`	`>`	`<`
`<=`	`>=`	`==`
`!=`		

Boolean	
`true`	`false`
`||`	`&&`
`!`	

Punctuation	
`{`	`}`
`(`	`)`
`,`	`;`

Assignment
`=`

String	
`+`	`""`
`length()`	`compareTo()`
`charAt()`	`matches()`

Arrays
`a[i]`
`new`
`a.length`

Objects	
`class`	`static`
`public`	`private`
`final`	`main()`
`new`	

Data Types in Java

A Java program/class is either:

- A data type, or
- A library of static methods

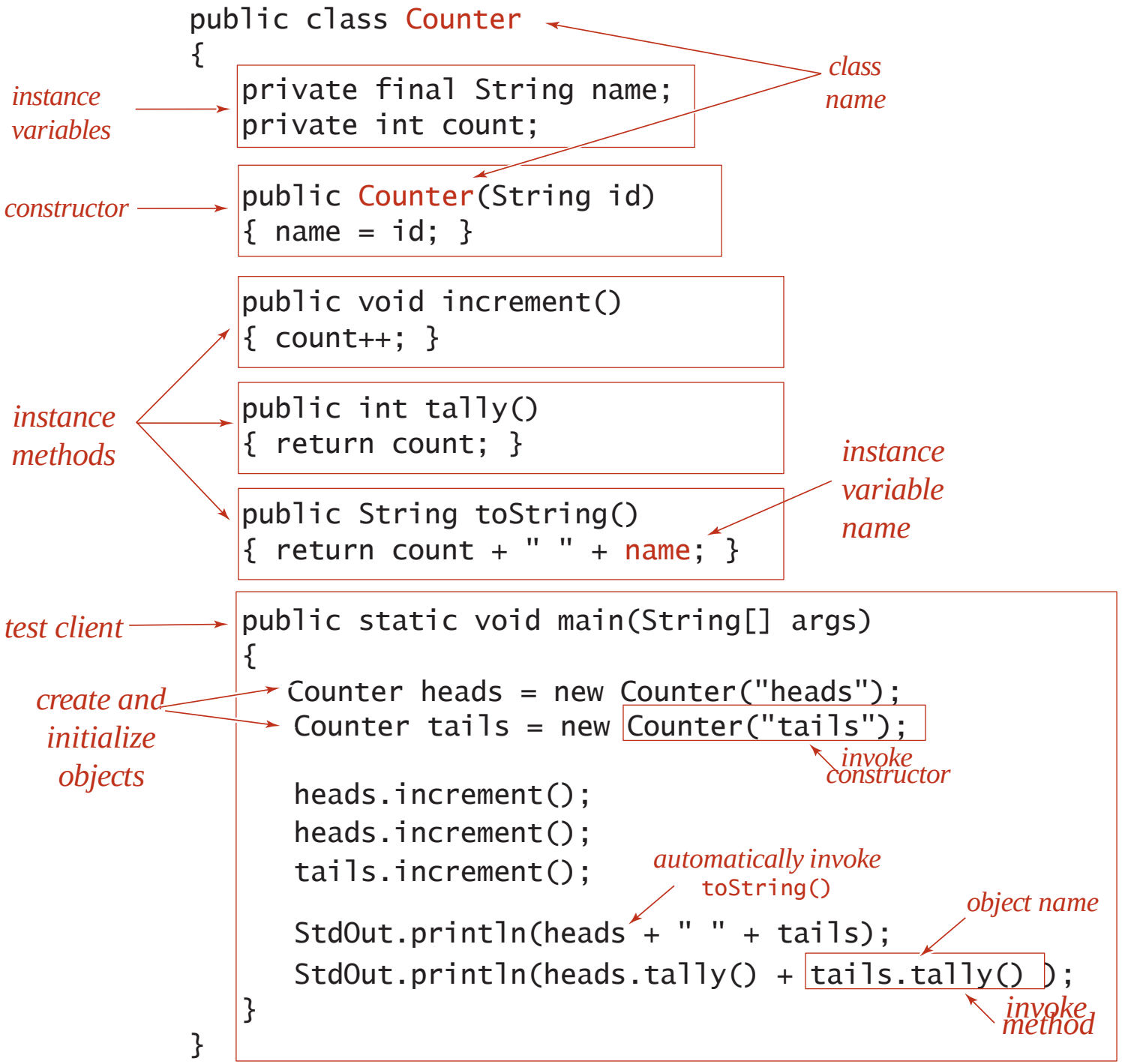
A data type is:

- Set of values, and
- Operations defined on those values.

Java class. Defines a data type by specifying:

- Instance variables. (set of values)
- Methods. (operations defined on those values)
- Constructors. (create and initialize new objects)

Anatomy of a class that defines a data type



Libraries

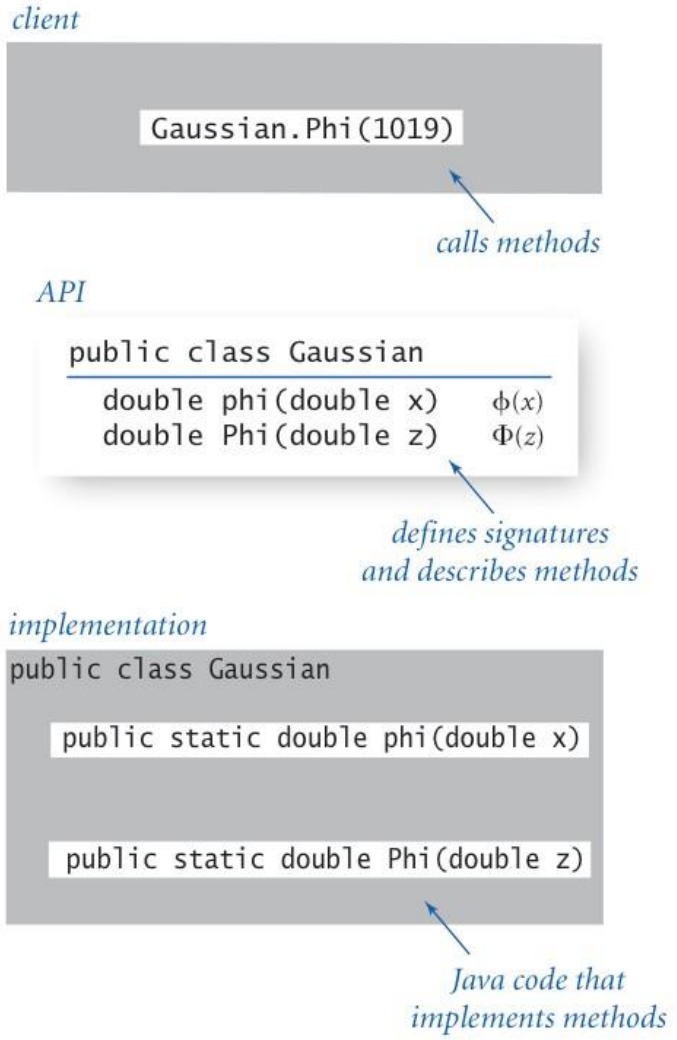
Library. A module whose methods are primarily intended for use by many other programs.

Client. Program that calls a library.

API. Contract between client and implementation.

Implementation. Program that implements the methods in an API.

Let's examine the textbook library



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

public class StdOut	
void print(String s)	*print* `s`
void println(String s)	*print* `s`*, followed by newline*
void println()	*print a new line*
void printf(String f, ...)	*formatted print*

Standard Input and Output

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
```

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*

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

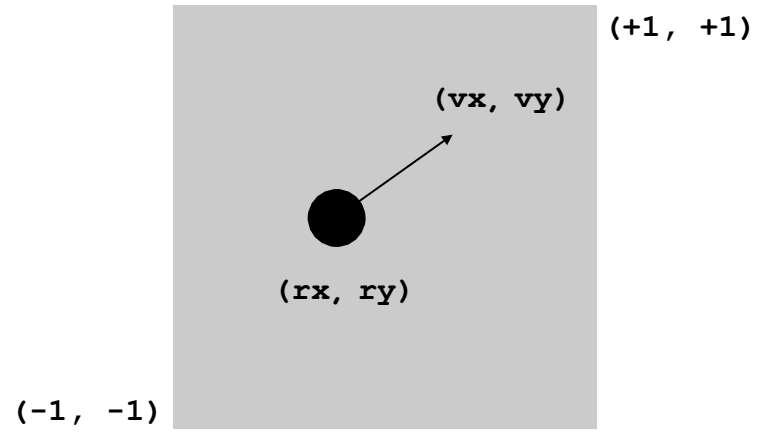
Animation

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;
      double vx = .015, vy = .023;
      double radius = .05;

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

      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;
         ry = ry + vy;

         StdDraw.setPenColor(StdDraw.GRAY);
         StdDraw.filledSquare(0.0, 0.0. 1.0);
         StdDraw.setPenColor(StdDraw.BLACK);
         StdDraw.filledCircle(rx, ry, radius);
         StdDraw.show(20);
      }
   }
}
```

position

constant velocity

radius

rescale coordinates

bounce

update position

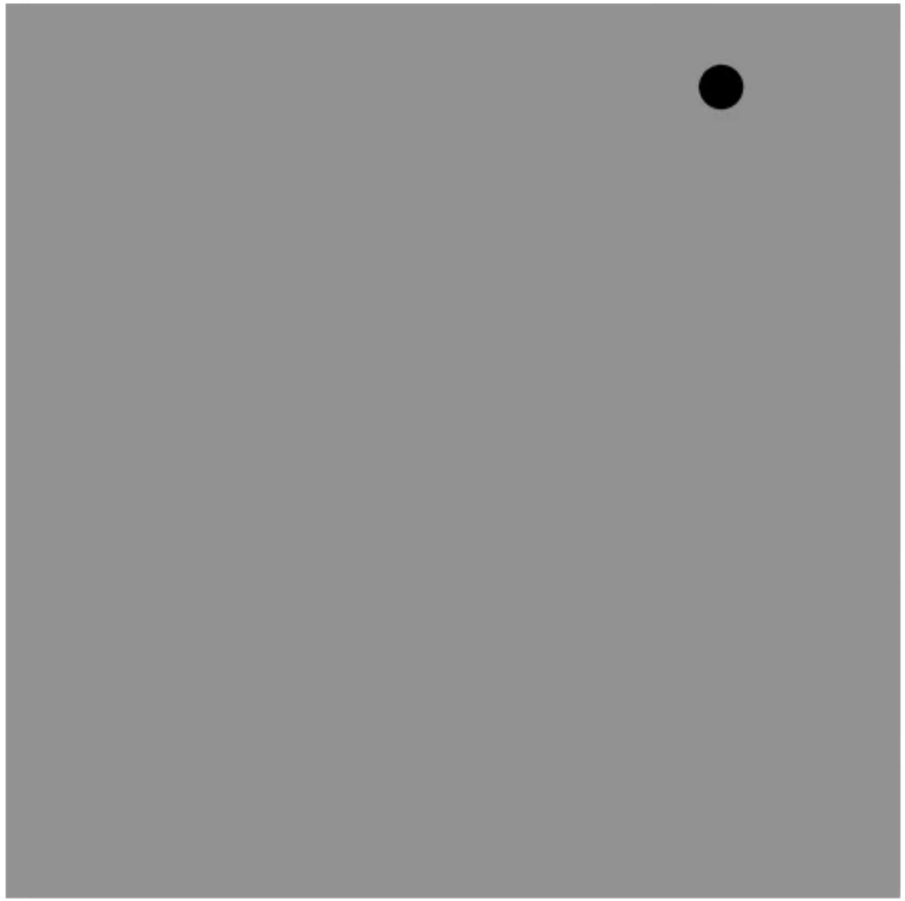
clear background

draw the ball

turn on animation mode:
display and pause for 50ms

Bouncing Ball Demo

```
% java BouncingBall
```



Standard Random

Standard random. Our library to generate pseudo-random numbers.

public class StdRandom	
int uniform(int N)	*integer between* 0 *and* N-1
double uniform(double lo, double hi)	*real between* lo *and* hi
boolean bernoulli(double p)	true *with probability* p
double gaussian()	*normal, mean* 0, *standard deviation* 1
double gaussian(double m, double s)	*normal, mean* m, *standard deviation* s
int discrete(double[] a)	i *with probability* a[i]
void shuffle(double[] a)	*randomly shuffle the array* a[]

```
int getRandomNumber()
{
   return 4;  // chosen by fair dice roll.
              // guaranteed to be random.
}
```

Standard Statistics

Ex. Library to compute statistics on an array of real numbers.

public class StdStats	
double max(double[] a)	*largest value*
double min(double[] a)	*smallest value*
double mean(double[] a)	*average*
double var(double[] a)	*sample variance*
double stddev(double[] a)	*sample standard deviation*
double median(double[] a)	*median*
void plotPoints(double[] a)	*plot points at* (i, a[i])
void plotLines(double[] a)	*plot lines connecting points at* (i, a[i])
void plotBars(double[] a)	*plot bars to points at* (i, a[i])

$$\mu = \frac{a_0 + a_1 + \cdots + a_{n-1}}{n}, \quad \sigma^2 = \frac{(a_0 - \mu)^2 + (a_1 - \mu)^2 + \cdots + (a_{n-1} - \mu)^2}{n - 1}$$

mean *sample variance*