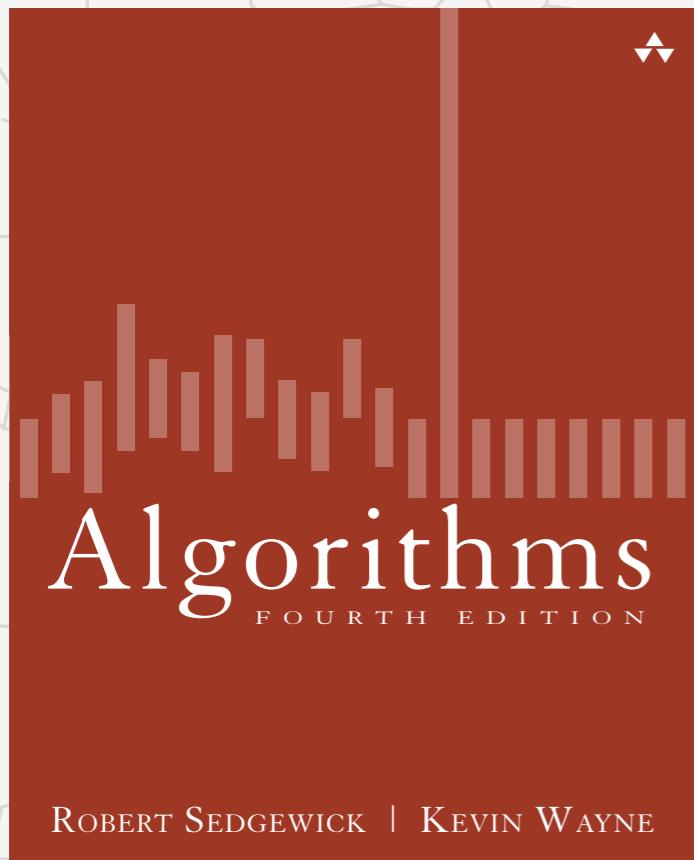


Algorithms

ROBERT SEDGEWICK | KEVIN WAYNE



ALGORITHMS, PARTS I AND II

- ▶ **overview**
- ▶ ***why study algorithms?***
- ▶ **resources**

<http://algs4.cs.princeton.edu>

Course overview

What is this course?

- Intermediate-level survey course.
- Programming and problem solving, with applications.
- **Algorithm:** method for solving a problem.
- **Data structure:** method to store information.

topic	data structures and algorithms	
data types	stack, queue, bag, union-find, priority queue	
sorting	quicksort, mergesort, heapsort	part 1
searching	BST, red-black BST, hash table	
graphs	BFS, DFS, Prim, Kruskal, Dijkstra	
strings	radix sorts, tries, KMP, regexps, data compression	part 2
advanced	B-tree, suffix array, maxflow	

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, ...

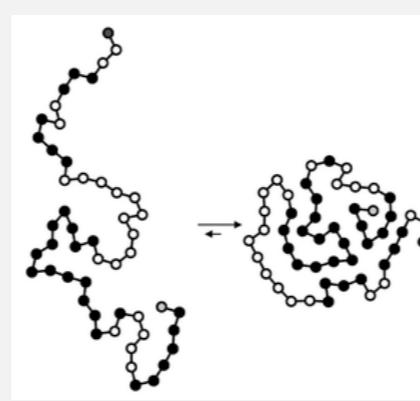
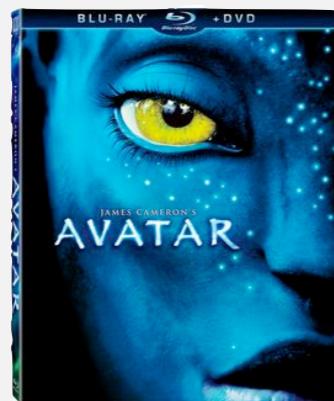
Multimedia. MP3, JPG, DivX, HDTV, face recognition, ...

Social networks. Recommendations, news feeds, advertisements, ...

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

:

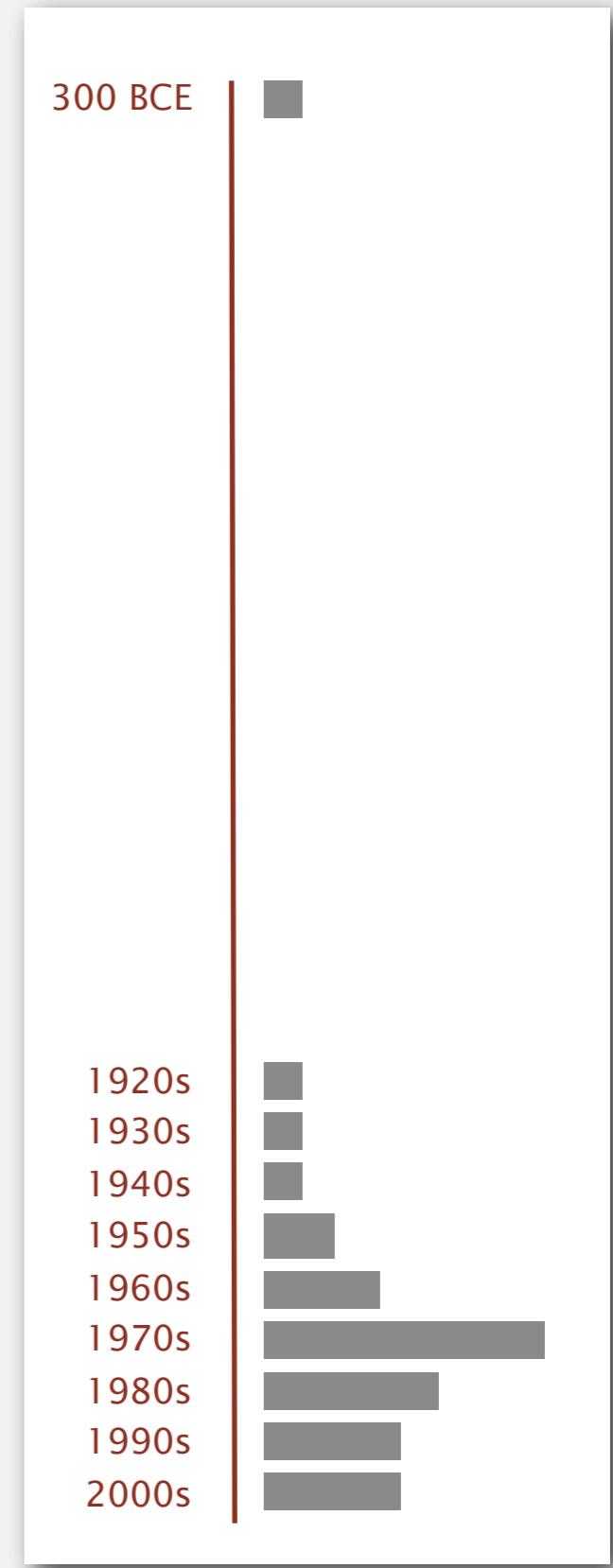
Google
YAHOO![®]
bing[™]



Why study algorithms?

Old roots, new opportunities.

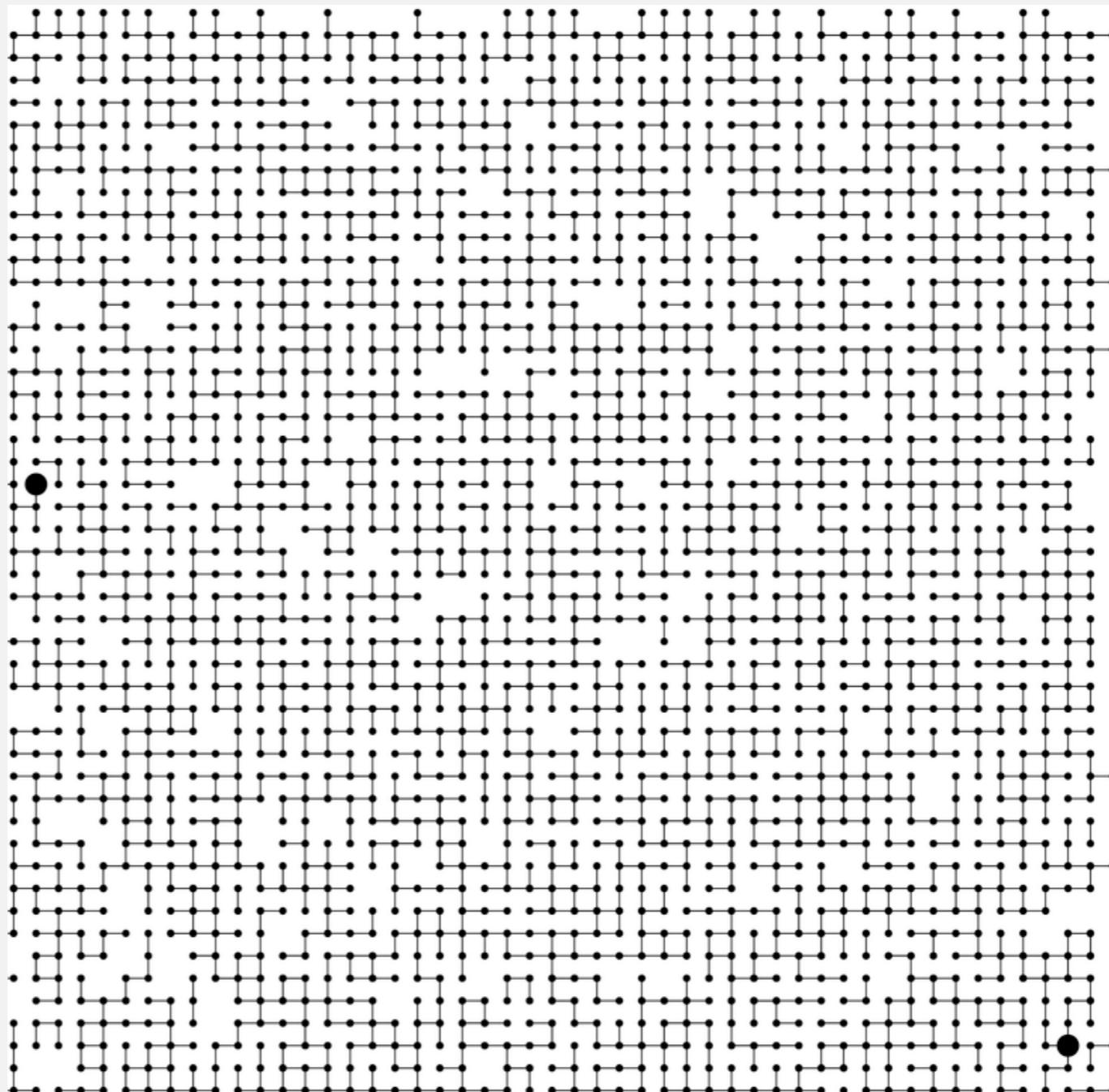
- Study of algorithms dates at least to Euclid.
- Formalized by Church and Turing in 1930s.
- Some important algorithms were discovered by undergraduates in a course like this!



Why study algorithms?

To solve problems that could not otherwise be addressed.

Ex. Network connectivity. [stay tuned]

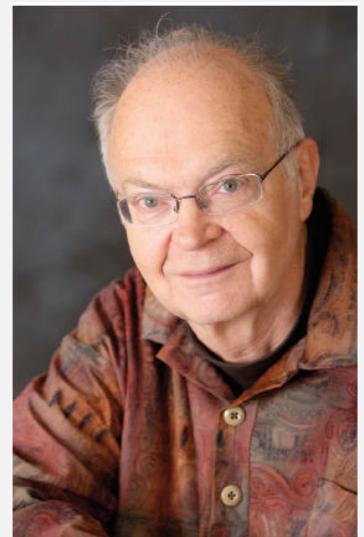


Why study algorithms?

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

“ An algorithm must be seen to be believed. ” — Donald Knuth



FROM THE
EDITORS

THE JOY OF ALGORITHMS

Francie 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. Sumerian cuneiform, one of the most ancient written records, contains parity algorithm descriptions for reckoning in base 60. And the first algorithmic procedure for estimating the sum of a series is embedded in Euclid's *Elements*. (That's really hard hardware!) Like so many other fields, algorithms have been flourishing in unexpected ways in the 20th century—at least it looks that way to us now. The algorithms have been applied to science, engineering, medicine, communications, health care, manufacturing, economics, weather prediction, defense, and fundamental science. Consider, please, that in less than eight hours I sat on the Maryland Shore when someone asked, "What's first a cat or a dog?" and I was able to answer without any speculation about the observed behavior of sex gals, someone who had just asked me the right answer—“A very hungry person.”

The flip side to “accuracy is the mother of invention” is “invention creates its own necessity.” Our need for powerful machines has driven the development of new algorithms. Our population booms that suggest the need, usually much larger, competition to be done. New algorithms are an attempt to make the best of what we have. We have become accustomed to getting the Moore’s Law factor of two every 18 months. In effect, Moore’s Law has become the algorithm of choice. It is a function of size, a function of problem size. Important new algorithms do not come along every 1, 3 years, but when they do, they change the world.

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 new light on some aspect of computing. A colleague recently claimed that he’d done only 15 minutes of productive work in his office over the last year. He was referring to time spent in the 15 minutes during which he sketched out a fundamental optimization algorithm. He regarded the previous years of thought and investigation as a sunk cost that might or might not ever paid off.

Researchers have cracked many hard problems since I Janus-edited my first issue of CISE in 1987. Now, in the next century, in spite of a lot of good work, the question of how to extract information from extremely large masses of data is still almost as hard as ever. In fact, it is getting harder. We have more data, more “big data,” to analyze. We have the way that *the sun* functions. The added computational cost is very small, but the added confidence in the answer is large. What is the next step? What is the next algorithmic optimization?

At even deeper level is the issue of reasonable methods of solving specific cases of “impossible” problems. What is the best way to search for a needle in a haystack? Is there a better way to attack the problem? What is the best way to attempt to answer any practical question. Are there efficient ways to attack them?

As we move into the next century, things will be ripe for another revolution in our understanding of the foundations of computational theory. Questions already arising from quantum computing, for example, will force us to reassess the notion of random numbers seem to require that we somehow tie together theories of computing, logic, and the nature of the physical world.

The new century is going to be very useful for us, but it is not going to be dull either! ■

2 COMPUTING IN SCIENCE & ENGINEERING

Why study algorithms?

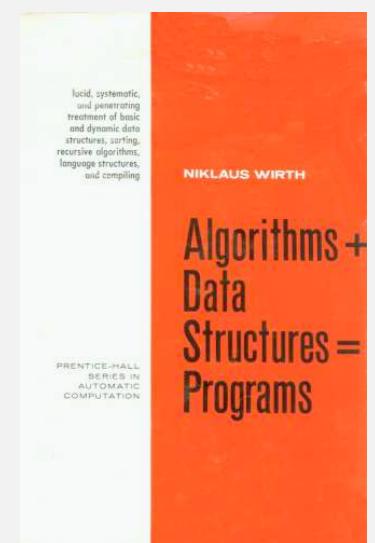
To become a proficient programmer.

“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?

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

Computational models are replacing math models in scientific inquiry.

$$E = mc^2$$

$$F = ma$$

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

20th century science
(formula based)

$$F = \frac{Gm_1 m_2}{r^2}$$

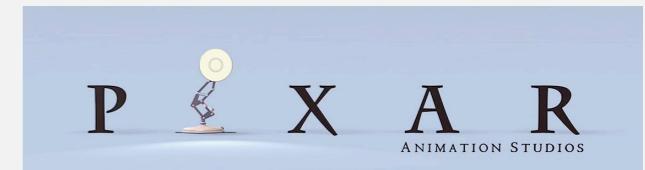
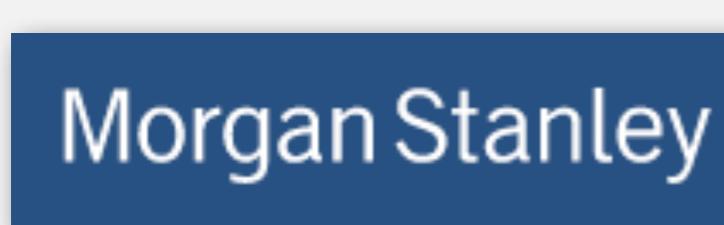
```
for (double t = 0.0; true; t = t + dt)
    for (int i = 0; i < N; i++)
    {
        bodies[i].resetForce();
        for (int j = 0; j < N; j++)
            if (i != j)
                bodies[i].addForce(bodies[j]);
    }
```

21st century science
(algorithm based)

“Algorithms: a common language for nature, human, and computer.” — Avi Wigderson

Why study algorithms?

For fun and profit.



Why study algorithms?

- Their impact is broad and far-reaching.
- Old roots, new opportunities.
- To solve problems that could not otherwise be addressed.
- For intellectual stimulation.
- To become a proficient programmer.
- They may unlock the secrets of life and of the universe.
- For fun and profit.

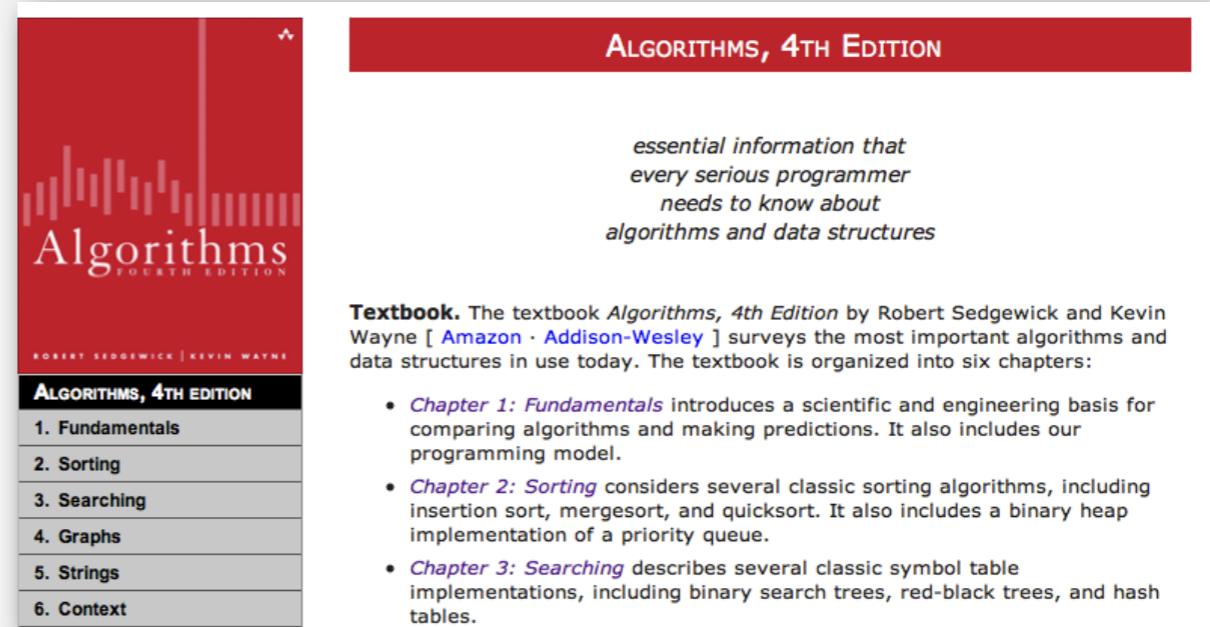
Why study anything else?



Resources

Booksite.

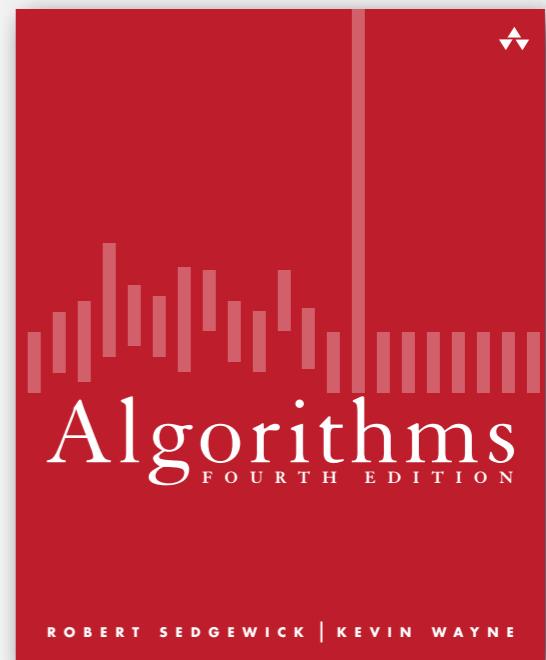
- Lecture slides.
- Download code.
- Summary of content.



<http://algs4.cs.princeton.edu>

Textbook (optional).

- *Algorithms, 4th edition* by Sedgewick and Wayne.
- More extensive coverage of topics.
- More topics.



ISBN 0-321-57351-X

Prerequisites

Prerequisites.

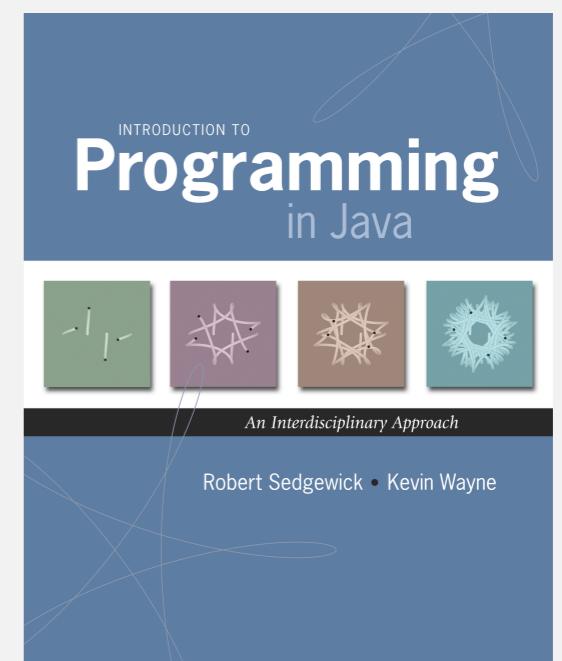
- Programming: loops, arrays, functions, objects, recursion.
- Java: we use as expository language.
- Mathematics: high-school algebra.

Review of prerequisite material.

- Quick: Sections 1.1 and 1.2 of *Algorithms, 4th edition*.
- In-depth: *An Introduction to programming in Java: an interdisciplinary approach* by Sedgewick and Wayne.

Programming environment.

- Use your own, e.g., Eclipse.
- Download ours (see instructions on web).



Quick exercise. Write a Java program.

ISBN 0-321-49805-4

<http://introcs.cs.princeton.edu>