

CSC 2053, SPRING 2014

**ALGORITHMS
AND
DATA STRUCTURES III**

M A PAPALASKARI

<http://www.csc.villanova.edu/~map/2053>

COURSE OVERVIEW

- ▶ outline
- ▶ why study algorithms?
- ▶ coursework
- ▶ resources

CSC 2053 course overview

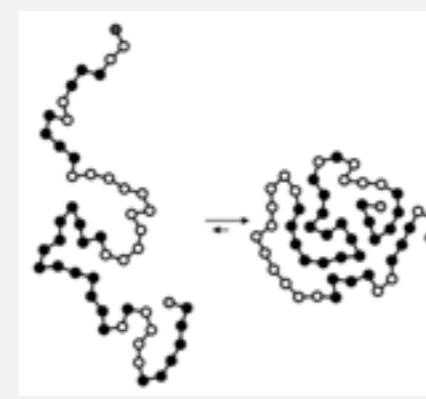
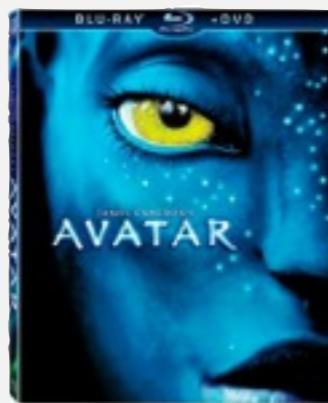
- Intermediate-level survey course.
- Third semester in intro CS sequence
- Programming and problem solving, with applications.
- **Algorithm:** method for solving a problem.
- **Data structure:** method to store information.

topic	data structures and algorithms
data types	stack, queue (review), bag, union-find, priority queue
searching	Binary search trees, hash tables
graphs	BFS, DFS, Prim, Kruskal, Dijkstra
strings	regular expressions, data compression

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, ...
- :

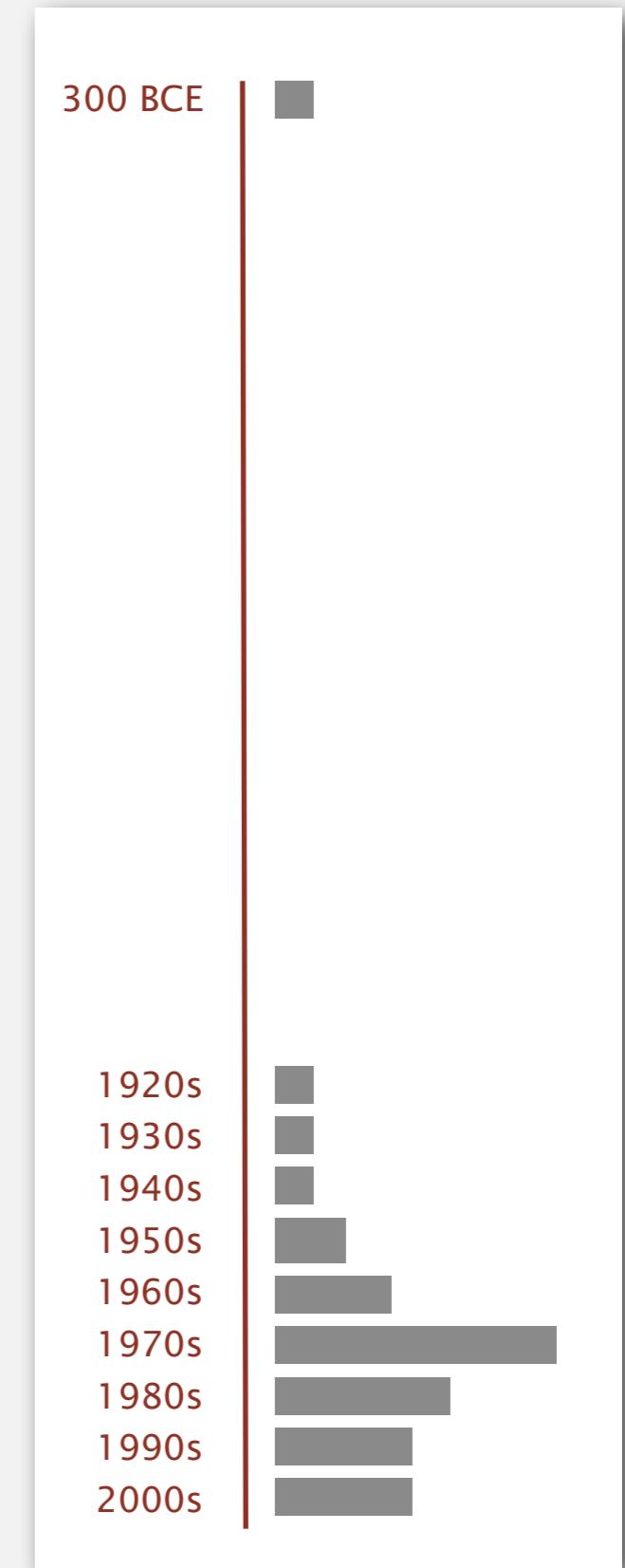
Google
YAHOO!
bing



Why study algorithms?

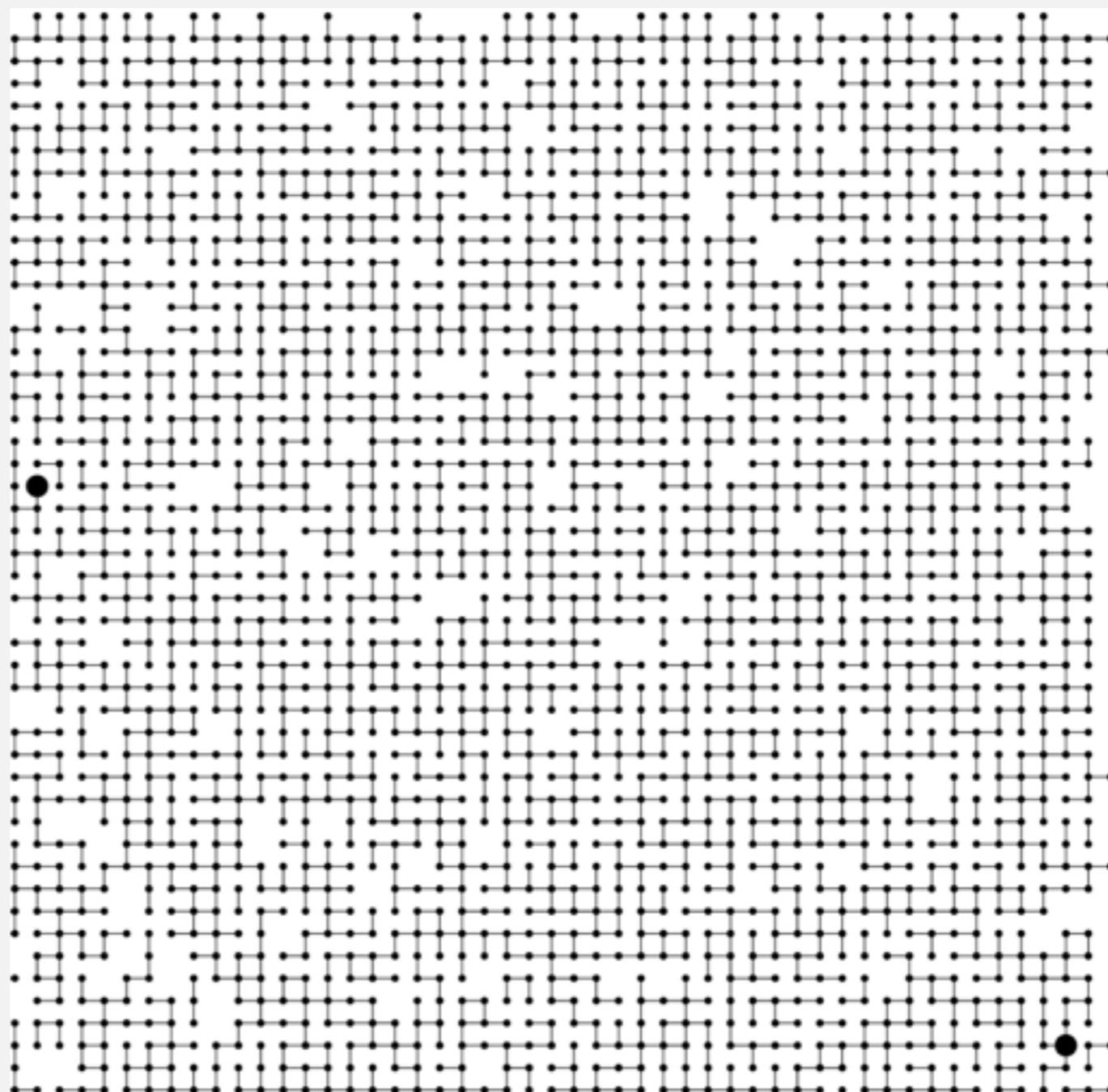
- Old roots, new opportunities.

- Study of algorithms dates at least to Euclid.
- Formalized by Church and Turing in 1930s.
- Some important algorithms were discovered by undergraduates in courses 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

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 WEVE 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 what may be the first algorithm ever recorded. And I suppose we could claim that the Dnael algorithm for estimating the start of summer is embodied in Stoschek. (That’s the name of the computer program that runs the algorithm.)

Like so many other things that technology affects, algorithms have advanced in startling and unexpected ways in the 20th century. In particular, the development of computational algorithms we chose for this issue have been essential for progress in communications, health care, manufacturing, economics, weather forecasting, and many other fields. And the field of robotics is still almost untouched. There are still very big challenges coming from more “traditional” tasks, too. For example, count efficiently the number of hairs on a person’s head. (Who else is going to do it?) Or calculate the trajectory of a large floating-point calculation likely to be correct. Think of the size of the problem. The answer is “a lot.” And the person who does it is “a very small, hot-headed person.” The answer is “large.”

Is there an analog for things such as large, multidisciplinary teams? “A large team,” you say. “A team.” “A team” invention creates its own necessity. Our need for powerful machines always exceeds their availability. Each significant computational task requires a different machine. And the larger, computation to be done. New algorithms are an attempt to bridge the gap between the demand for cycles and the available supply. The gap is growing. Moore’s Law, which states that the number of transistors on a chip doubles every 18 months, is slowing down. In effect, Moore’s Law changes the constant in front of the estimate of running time as a function of the number of transistors. The constant do not come along every 1.5 years, but when they do, they can change the exponent of the complexity.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

Computers have cracked many hard problems since 1 January 1900, but we are posing some even harder ones to the next century. In spite of a lot of good work, the question of communications, health care, manufacturing, economics, weather forecasting, and many other fields remains unanswered. There are still very big challenges coming from more “traditional” tasks, too. For example, count efficiently the number of hairs on a person’s head. (Who else is going to do it?) Or calculate the trajectory of a large floating-point calculation likely to be correct. Think of the size of the problem. The answer is “a lot.” And the person who does it is “a very small, hot-headed person.” The answer is “large.”

Is there an analog for things such as large, multidisciplinary teams? “A large team,” you say. “A team.” “A team” invention creates its own necessity. Our need for powerful machines always exceeds their availability. Each significant computational task requires a different machine. And the larger, computation to be done. New algorithms are an attempt to bridge the gap between the demand for cycles and the available supply. The gap is growing. Moore’s Law, which states that the number of transistors on a chip doubles every 18 months, is slowing down. In effect, Moore’s Law changes the constant in front of the estimate of running time as a function of the number of transistors. The constant do not come along every 1.5 years, but when they do, they can change the exponent of the complexity.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse, they can be terse, allusive, dense, and even

not have paid off.

If you’re interested in the poetry of computation, Just like verse,

Why study algorithms?

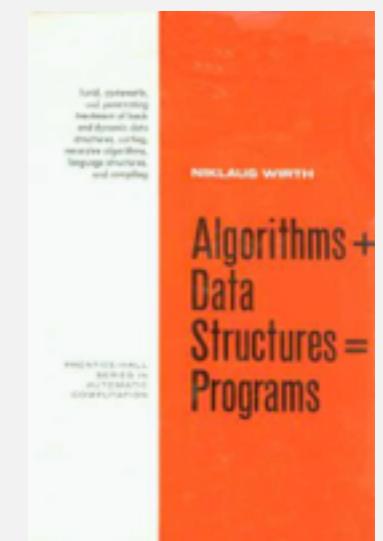
- To become a proficient programmer.

“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 mathematical models in scientific inquiry.

$$E = mc^2$$

$$F = ma$$

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

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

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.” — Avi Wigderson

Why study algorithms?

- For fun and profit.

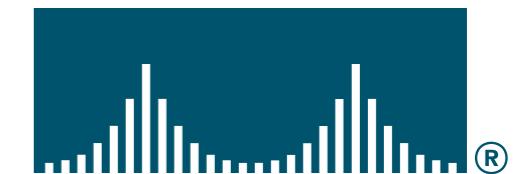
Google™



Apple Computer

facebook®

CISCO SYSTEMS



Morgan Stanley

IBM

Nintendo®



DE Shaw & Co

ORACLE®

P
PANDORA®

Akamai

YAHOO!

amazon.com

Microsoft®

PIXAR
ANIMATION STUDIOS

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?



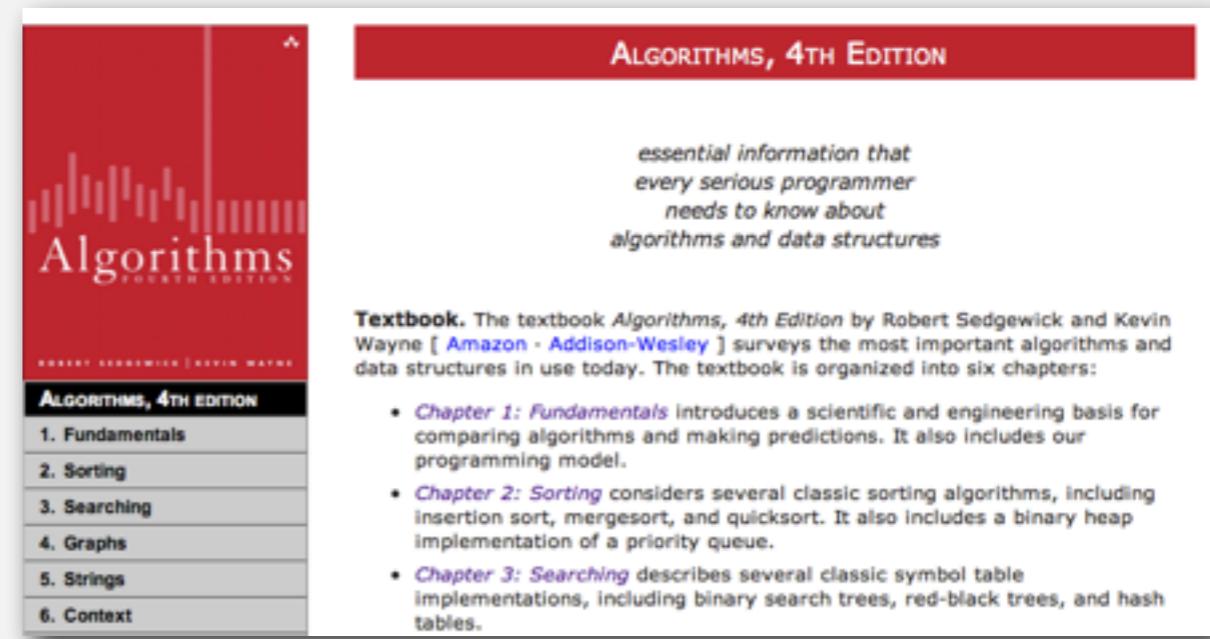
- Course website.

- Course info.
- Schedule
- Programming assignments.
- Lecture slides.

<http://www.csc.villanova.edu/~map/2053/s14>

- Booksite.

- Brief summary of content.
- Download code from book.



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

Where to get help?

- Piazza. Online discussion forum.

- Low latency, low bandwidth.
- Mark solution-revealing questions as private.



<http://piazza.com/villanova/spring2014/csc2053>

- Office hours.

- High bandwidth, high latency.
- See web for schedule.



<http://www.csc.villanova.edu/~map/2053/s14>

- Helpdesk.

- Grad TAs in Mendel G85.
- For help with technical issues & debugging.
- See schedule.

<http://www.princeton.edu/~cos226>