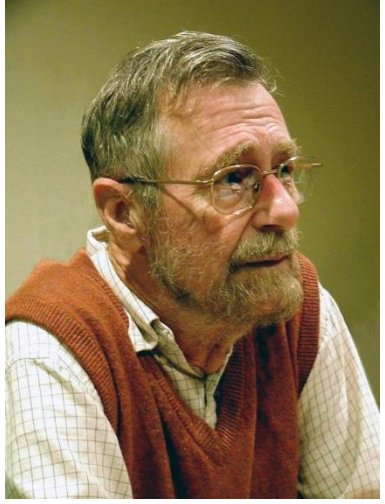
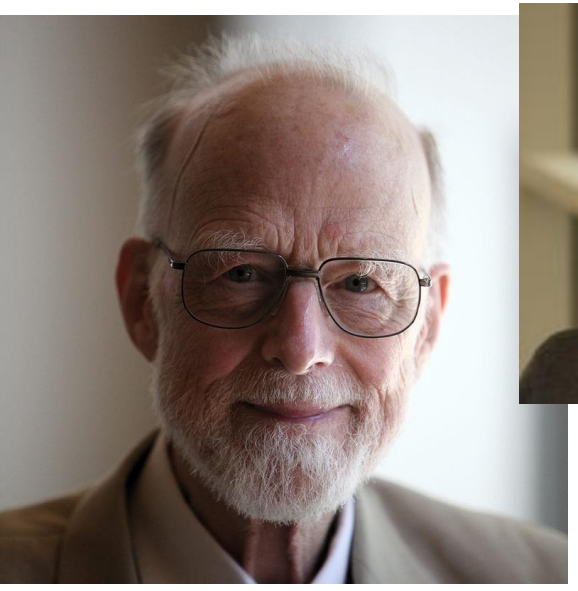
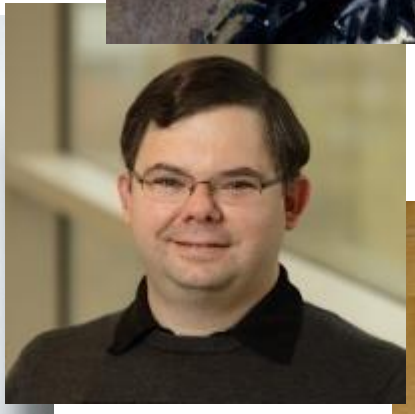
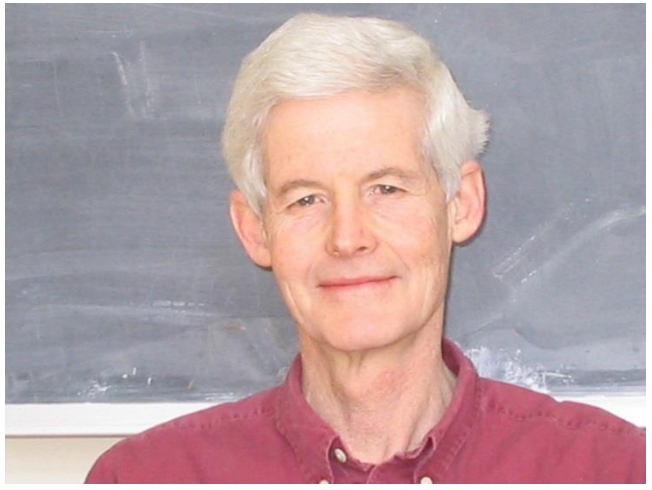
CS4102 Algorithms

Fall 2018



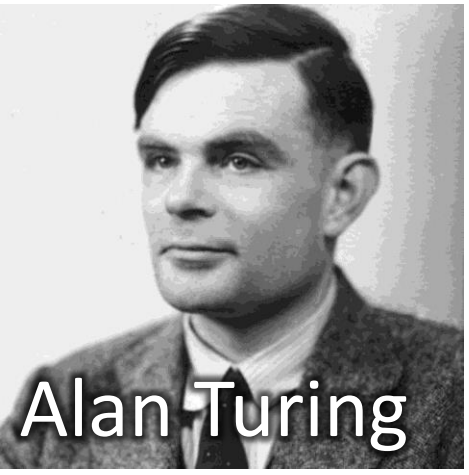


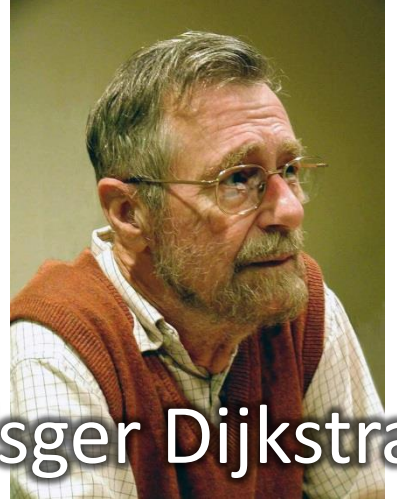


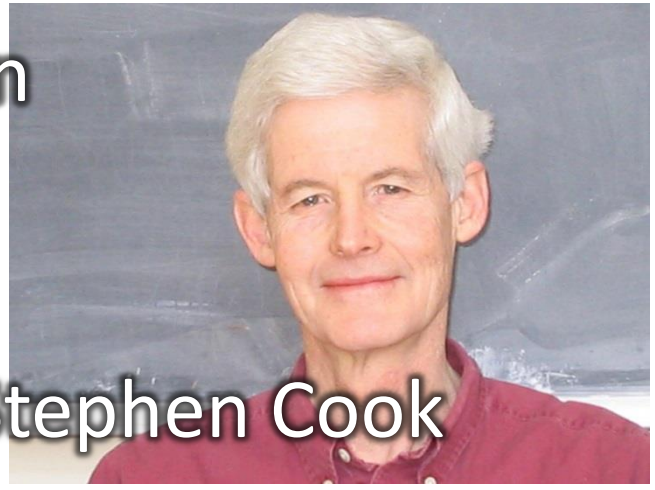


CS4102 Algorithms

Fall 2018



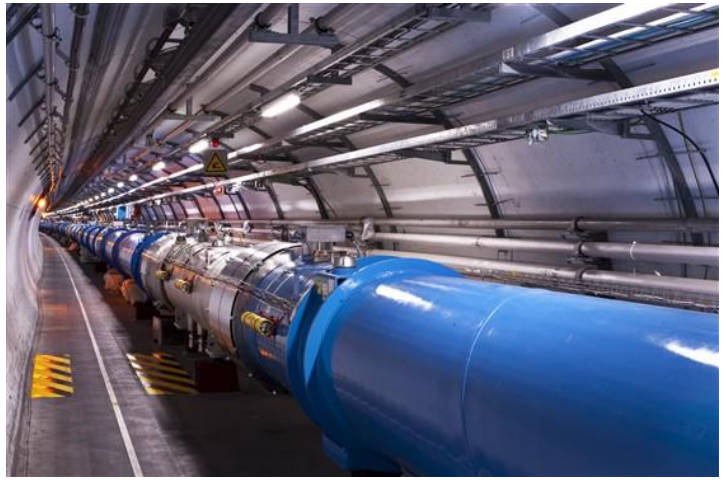




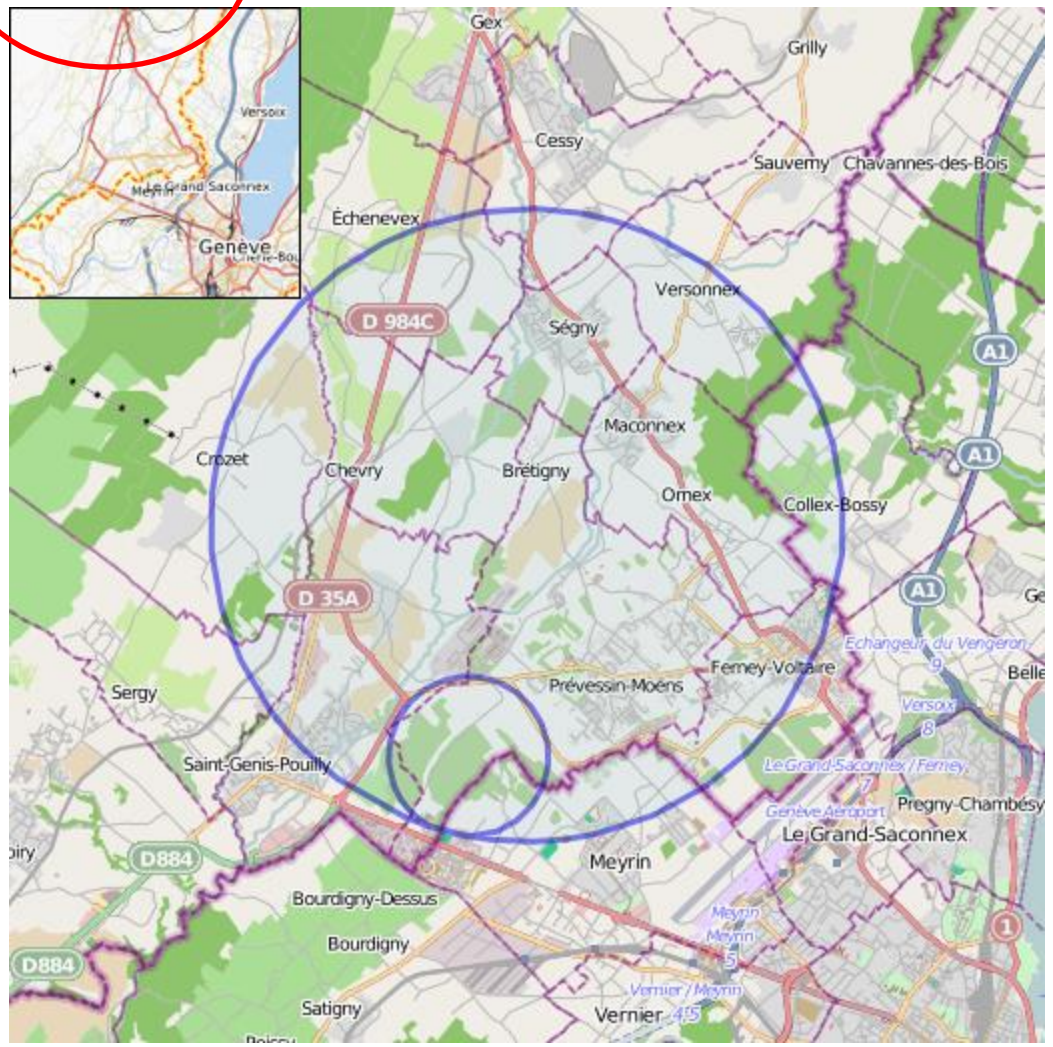
What is an algorithm?

- In mathematics and computer science, an algorithm is a self-contained sequence of actions to be performed. Algorithms can perform calculation, data processing and automated reasoning tasks. [Wikipedia]
- Motivating example

Need an accurate approximation → π



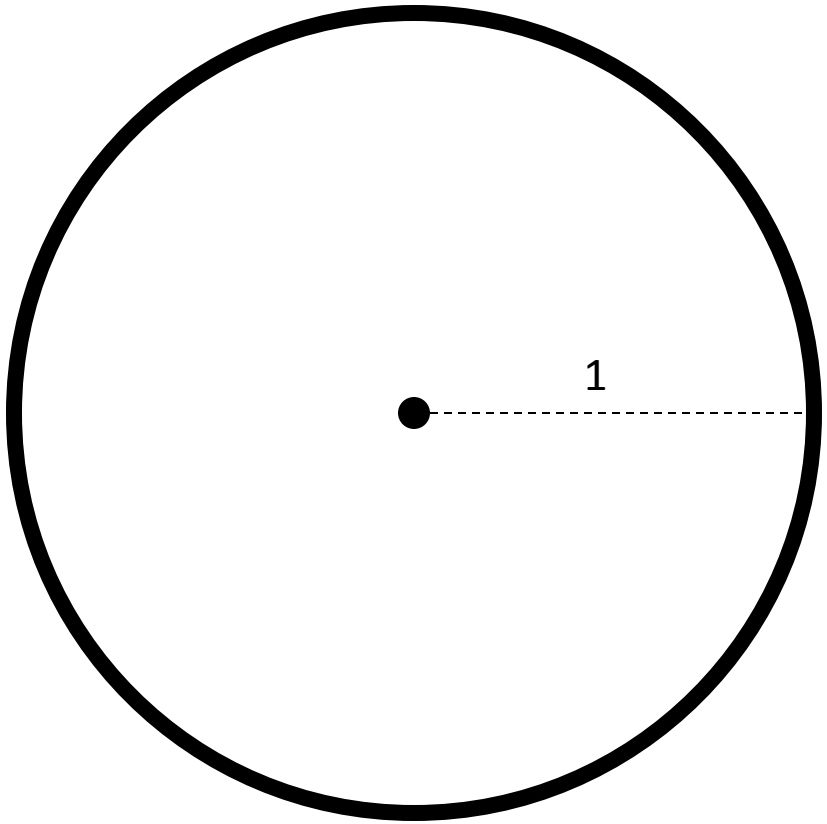
How much concrete do I need?



4.3km (2.7mi) diameter

π Approximation Algorithm

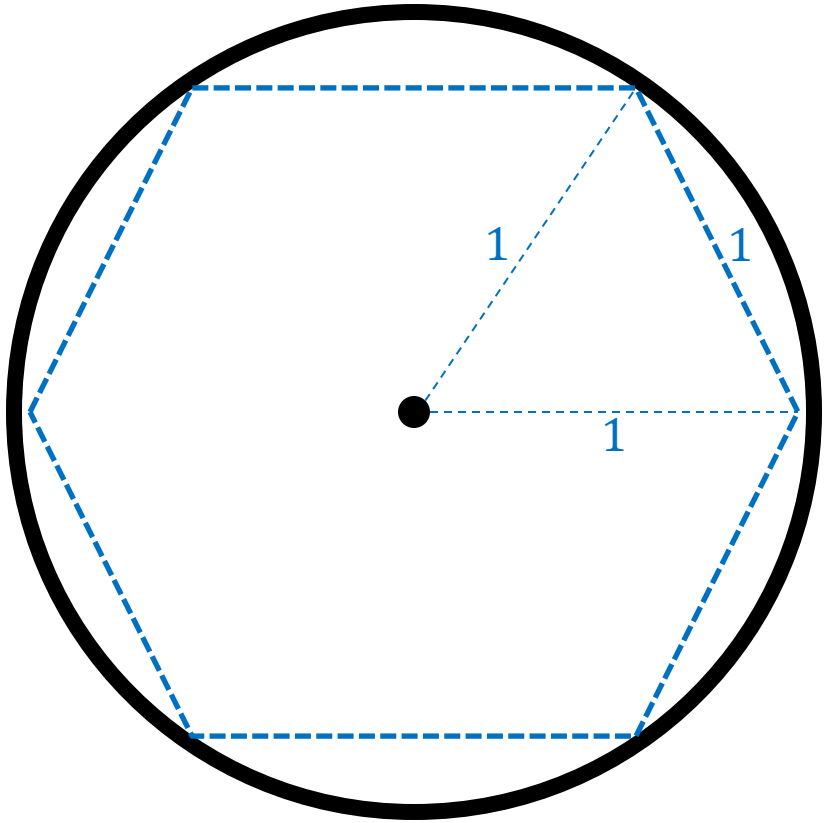
$\pi = 3.14159265359...$



Circumference = 2π

π Approximation Algorithm

$\pi = 3.14159265359\ldots$



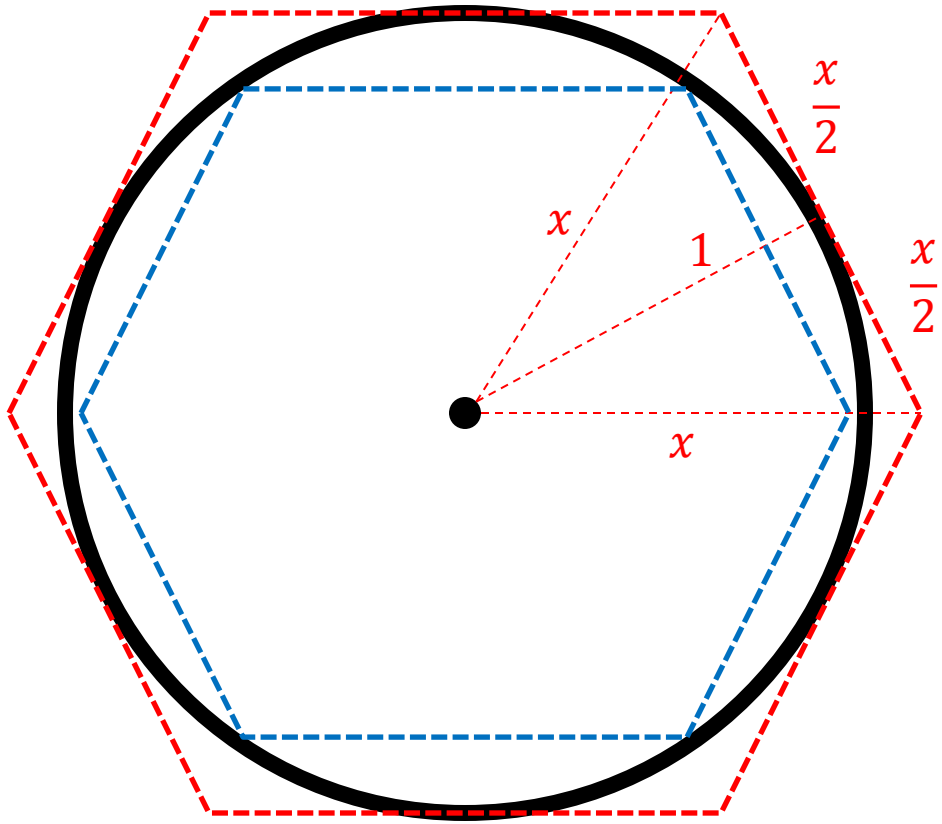
$2\pi >$ Perimeter $= 6$

π Approximation Algorithm

$\pi = 3.14159265359...$ 1 digit correct

Solve for x

$$x = \frac{2}{\sqrt{3}}$$

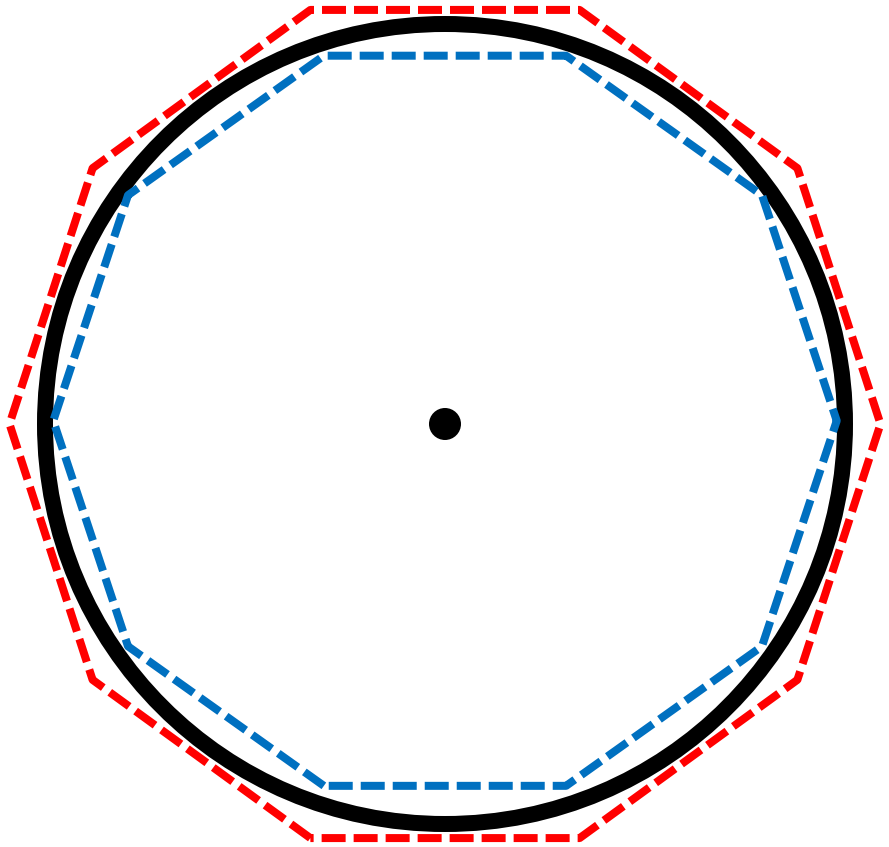


$$\frac{12}{\sqrt{3}} = \text{Perimeter} > 2\pi > \text{Perimeter} = 6$$

$$3.46 > \pi > 3$$

π Approximation Algorithm

$\pi = 3.14159265359...$ 3 digits correct



$$6 + \frac{20}{70} = \text{Perimeter} > 2\pi > \text{Perimeter} = 6 + \frac{20}{71}$$

$$3.14285 > \pi > 3.14084$$

How to analyze this approach?

- How fast do we “converge”?
- How much work is needed to do better?

Better π Approximation (Ramanujan)

$$\frac{1}{\pi} = \frac{2\sqrt{2}}{9801}\sum_{k=0}^{\infty}\frac{(4k)!\,(1103+26390k)}{(k!)^4\,396^{4k}}$$

$\pi = 3.14159265358979323846264338327950288419716939937510582$

$k = 0$

$\pi \approx 3.1415927$

8 digits per iteration!

$k = 1$

$\pi \approx 3.14159265358979 38$

Goals

- Create an awesome learning experience
- Instill enthusiasm for problem solving
- Give broad perspective on Computer Science
- Have fun!

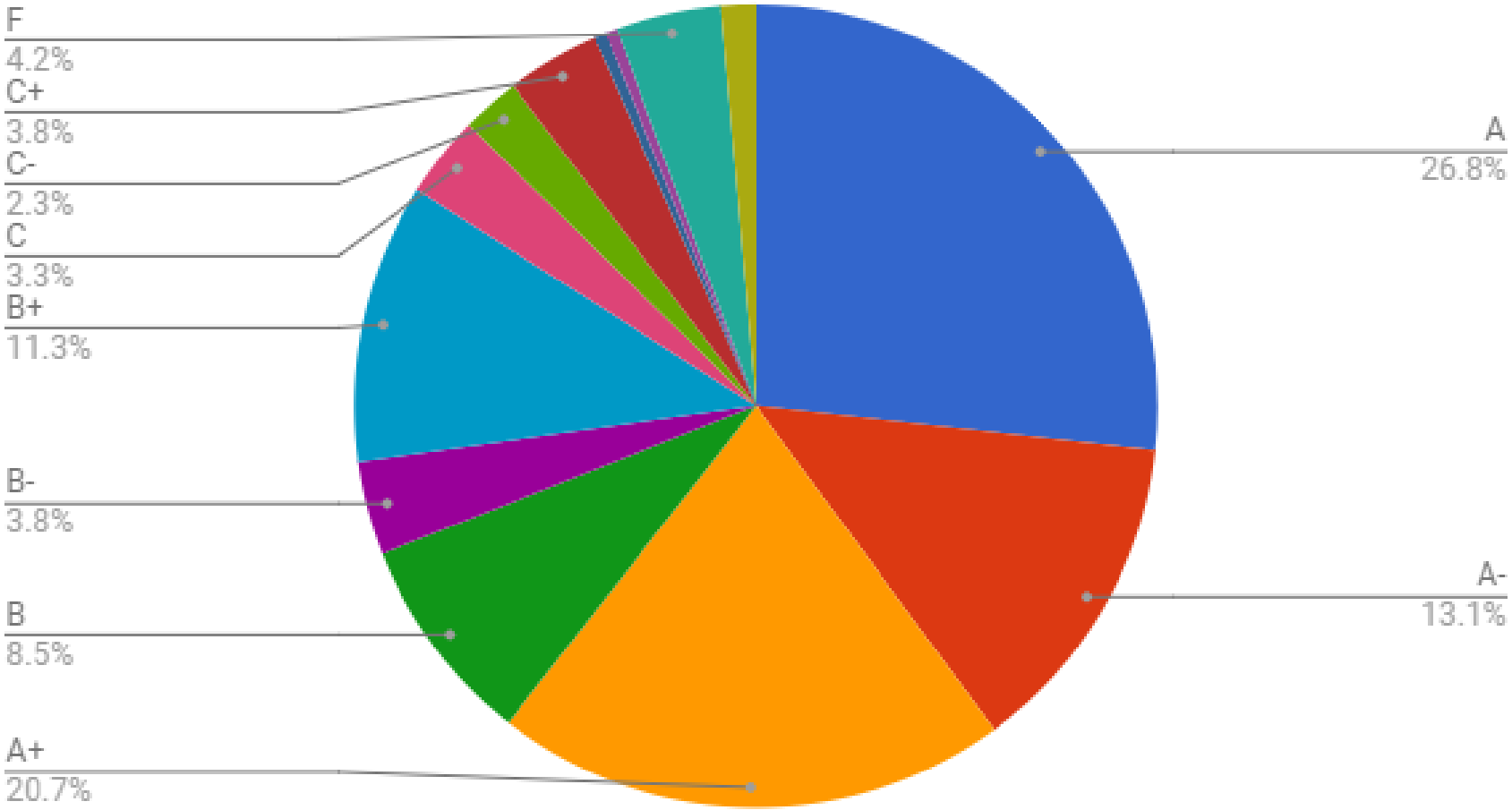
Warning

- This will be a very difficult class
  - Hard material
  - “Holy Grail” of computer science
  - Useful in practice
  - Job Interviews
- Lots of opportunities to succeed!

Hopefully not you…

- He is very reasonable in what he expects from students and makes the course appropriately hard and time consuming.
- The professor was extremely helpful both in and out of class and does not appear to give hard homework for the sake of defeating his students.
- The vast majority of the time he was able to explain the hard concepts effectively and made me more motivated and confident about the rigorous topics during the semester.
- The class is very difficult. It's not easy, but it's important material.
- The homework felt brutally difficult compared to the lecture and midterm, but was probably so for a reason.
- This class was the perfect balance of being true to the difficult course material but entirely fair
- The class was pretty difficult, but I felt like I was given more than enough resources to succeed (and do better than I actually did lol)
- He was also incredibly flexible with his assignments knowing full well that his assignments were incredibly difficult
- The homework was very difficult and I think a little too hard to get an A in
- GETTING AN A IN THIS CLASS IS HARDER THAN SNEAKING INTO THE BASE OF NSA WITH OR WITHOUT GETTING SHOT

While difficult, students have done well…



Office Hours

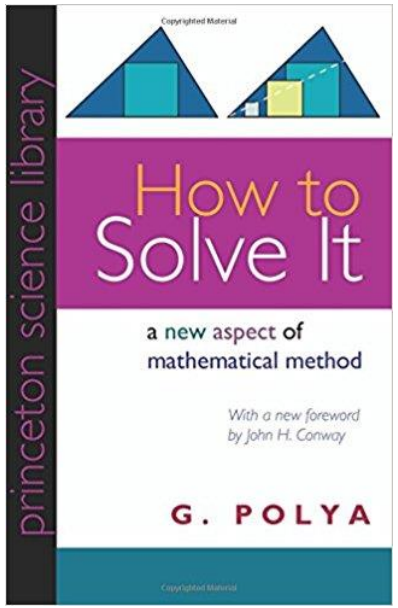
- Prof. Hott
  - Rice 210
  - Th 11am-12:30pm (this class)
  - W 2pm-3:30pm (2110)
  - By appointment
- Prof. Brunelle
  - Rice 209
  - W 4pm-6pm
  - By appointment

Requirements

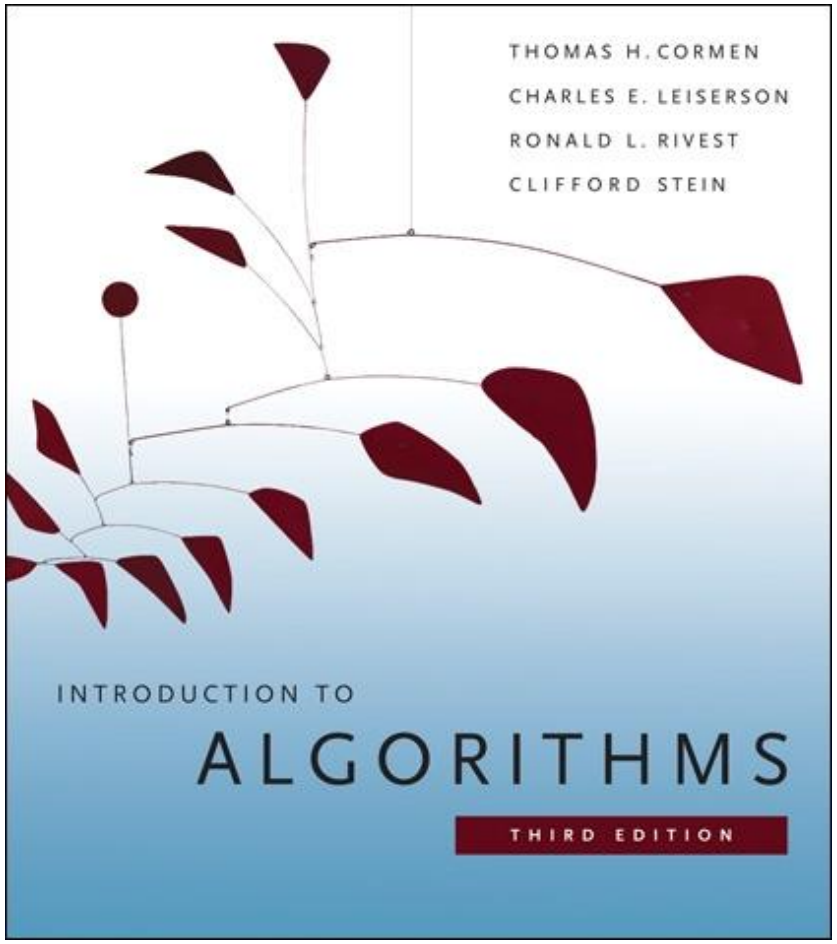
- Discrete Math (CS 2102)
- Data Structures (CS 2150)
- Derivatives, series (Calc I)
- Tenacity
- Inquisitiveness
- Creativity

Textbook

- No textbook required
- Highly recommended:

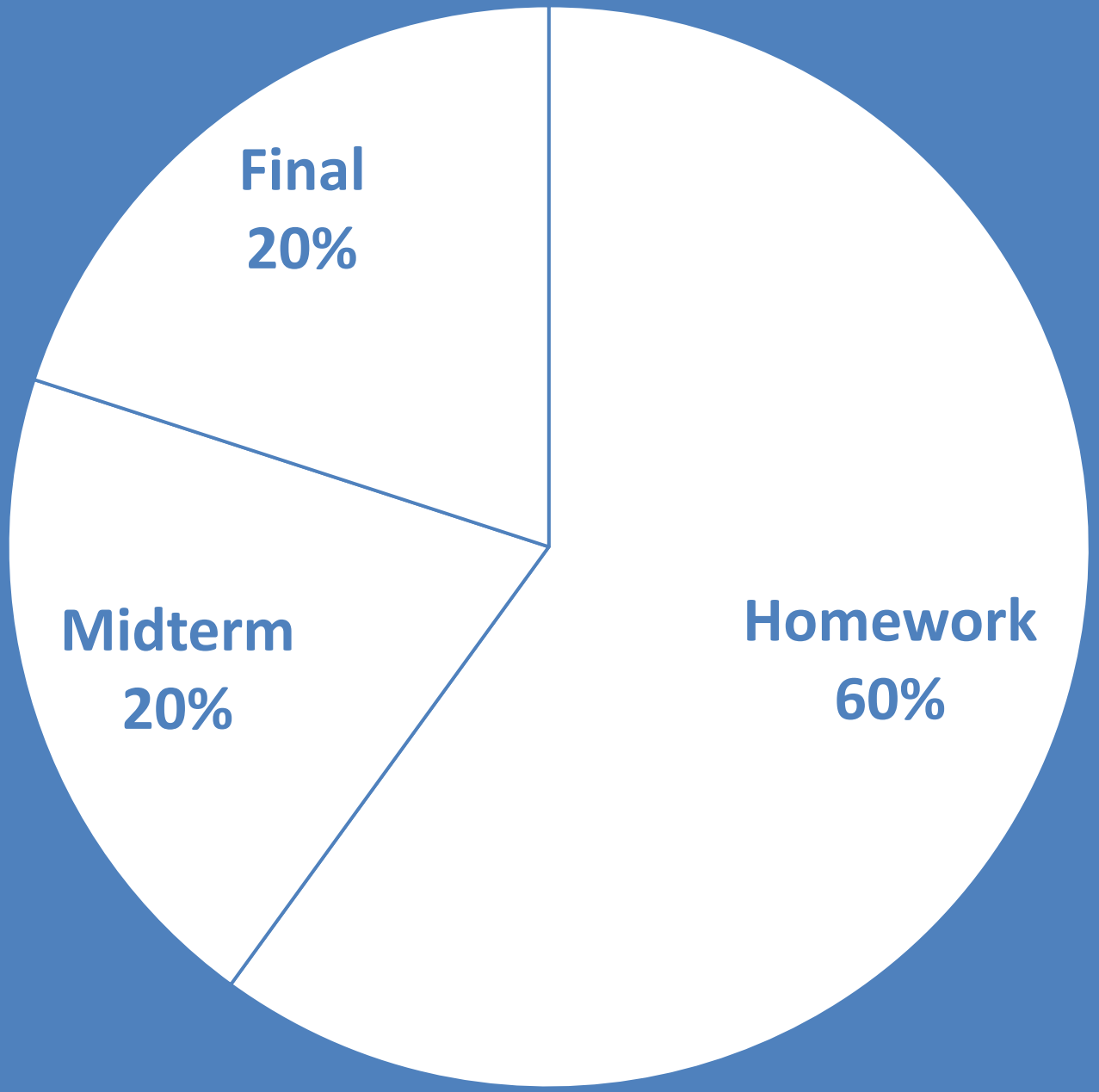


Polya. *How to Solve It.*



Cormen et al. (CLRS) *Introduction to Algorithms*. Third Edition.

Grade Breakdown



10% Extra Credit

Homework

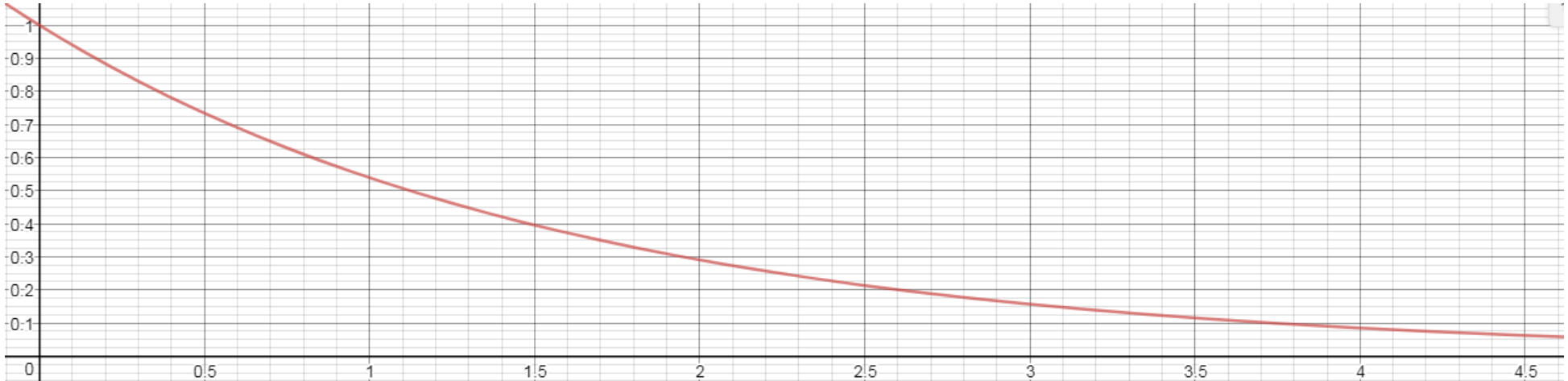
- 11 assignments total
- Mix of written and programming assignments
- Written:
  - 2/3 of all assignments
  - Must be typeset in LaTeX (tutorial is HW0)
  - Submit as a **pdf** and a **zip** folder containing tex file and any supplements
    - Submissions without both attachments (pdf, zip) will not be graded
- Programming:
  - 1/3 of all assignments
  - Must implement in Python or Java

Academic Integrity

- Collaboration Encouraged!
  - Groups of up to 5 per assignment
  - List your collaborators
- Write-ups/code written independently
- Be able to explain any solution you submit!
- DO NOT seek published solutions online

Late Policy

- $grade = grade_{earned}\, e^{-\frac{1}{\phi} days}$
- Exponential decay
- Extra credit for the radioactive isotope with half-life closest to your homework's
- Accepted until solutions posted



Exams

- Midterm
  - Est. October 16
  - In-class / take-home hybrid
- Final
  - Registrar's official date/time
  - Monday 12/10, 2-5pm (9:30am section)
  - Monday 12/17, 9am-12pm (2pm section)
- You **must** show up to the correct section's exam
  - midterm and final

Regrades

- Conducted in person w/ Prof Hott or Prof Brunelle
  - Tuesday 4pm-5pm
  - By appointment

Extra credit

- Given for extraordinary acts of engagement
  - Good questions/comments
  - Quality discussions
  - Problem solving session attendance
  - Analysis of current events
  - References to arts
  - Extra credit projects
  - Slide corrections
  - Etc. Just ask!
- Email: **extra.credit.cs4102@gmail.com**

Feedback

- I am not a course dictator, I am a civil servant
- I’m open to any suggestion to help you learn
- Let me know!
  - In person
  - Email
  - Piazza
  - Anonymous feedback (as long as it’s constructive)

Attendance

- How many people are here today?
- Naïve algorithm
  1. Everyone stand
  2. Professor walks around counting people
  3. When counted, sit down
- Run time?
  - Class of n students
  - O(n)
- Other suggestions?

Better Attendance

1. Everyone Stand
2. Initialize your “count” to 1
3. Greet a neighbor who is standing: share your name, full date of birth(pause if odd one out)
4. If you are older: give “count” to younger and sit.
   Else if you are younger: add your “count” with older’s
5. If you are standing and have a standing neighbor, go to 3