CS 441: Discrete Structures for Computer Science

PhD. Nils Murrugarra-Llerena
nem177@pitt.edu

Who am I?

B.S. Computer Science at **National University of Trujillo**

M.S. in Computer Science at **University of São Paulo** in **AI**

Who am I?

PhD in Computer Science at **University of Pittsburgh** in **Computer Vision**

Research scientist at **Snap Inc.**

Who am I?

Assistant professor at
Weber State University

Teaching Assistant Professor at **University of Pittsburgh**

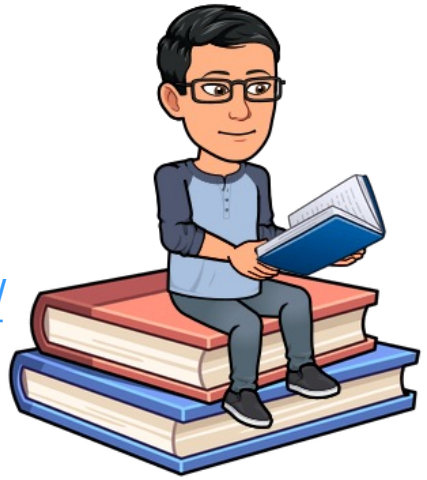
[Students' presentations]

Name, hobbies, and mention one thing that you expect to learn in this course ☺

Syllabus

Syllabus

Course intro: Syllabus



- Contact Information
  - Prof. Nils Murrugarra
  - nem177@pitt.edu
  - Please, add prefix “[CS 441]” in all emails.
  - Website: https://nineil.github.io/courses/fall24_cs441/

- Lectures:
  - [Section A] Mon/Wed: 1:00pm - 2:15pm @ SENSQ 5502
  - [Section B] Mon/Wed: 3:00pm - 4:15pm @ LAWRN 207

- Office hours:
  - Tue/Wed 9:00 am – 11:00 am [SENSQ 5419]

Course intro: Textbook

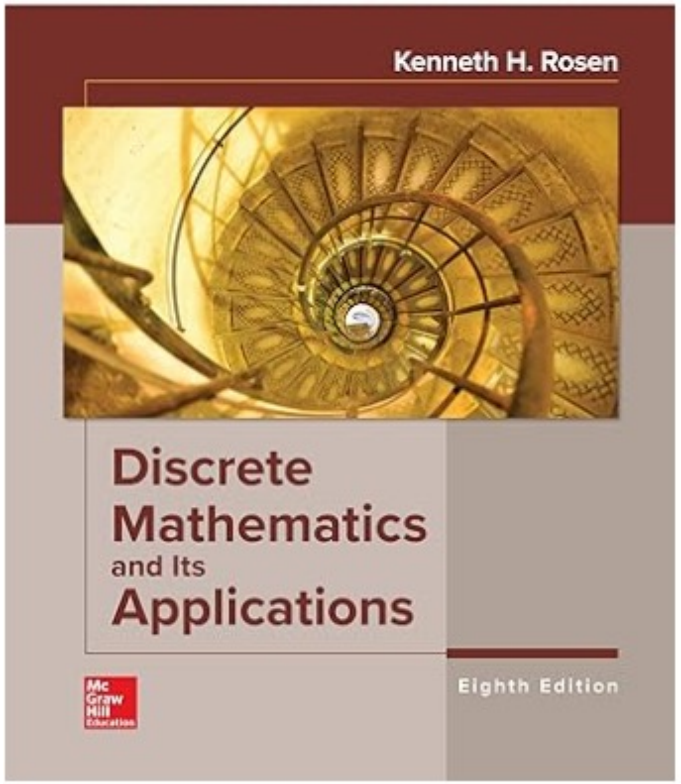
Discrete Mathematics and its Applications

Edition: 8th

By Kenneth Rosen

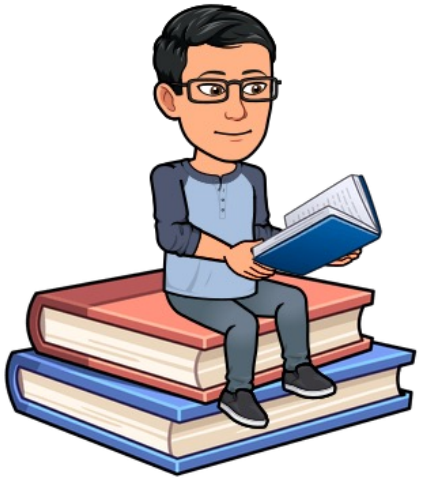
ISBN: 1260091996

Year: 2018



Course intro: What to expect?

- Material is based on previous iterations of *CS 441: Discrete Structures for Computer Science* from Prof. William Garrison.
- Exams mainly cover this material
- We will do around 11 to 13 assignments



Course intro: What to expect?

- There will be a lot of work!
- However, you will learn a lot :). Please, ask questions in class and use office hours as needed.
- I would like to help you much as possible.

Course intro: What to expect?

H/T Kirk Pruhs

Review Syllabus

Canvas Link:

https://canvas.pitt.edu/courses/273358/files/17495577?module_item_id=5014615

Questions?

Course overview

- What ***is*** discrete mathematics?
- Why is a math course part of the computer science curriculum?
- Will I really ever use this stuff again?
- How to succeed in this course?

What is discrete mathematics?

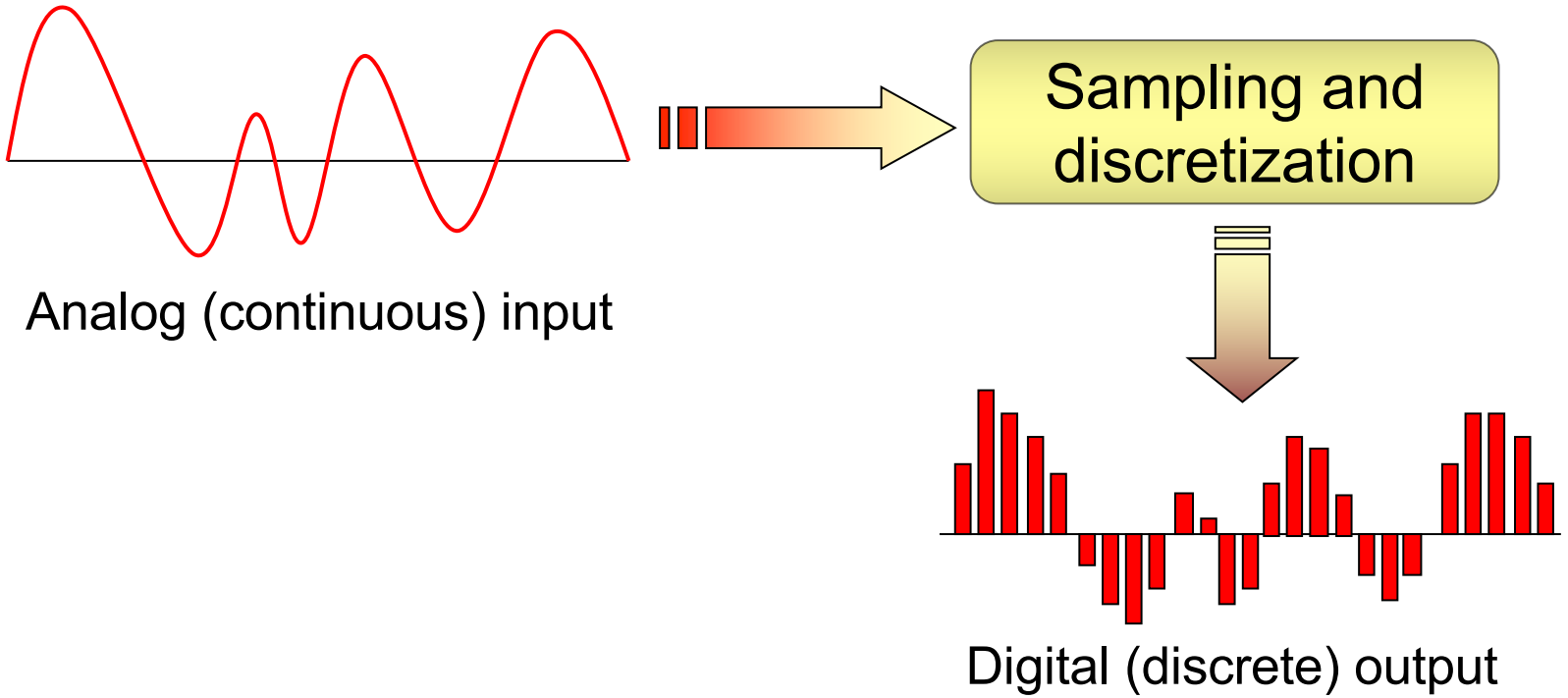
- Discrete mathematics is the study of *distinct* objects or structures and their relationships to one another

- For example:
  - How many ways can a valid password be chosen?
  - Can traffic flow between two computers in a network?
  - How can we transform messages to hide their contents?
  - How do we parse a given sequence of commands?

- By contrast, continuous mathematics (e.g., calculus) studies objects and relationships that vary continuously
  - e.g., position, velocity, and acceleration of a projectile

Why study discrete math?

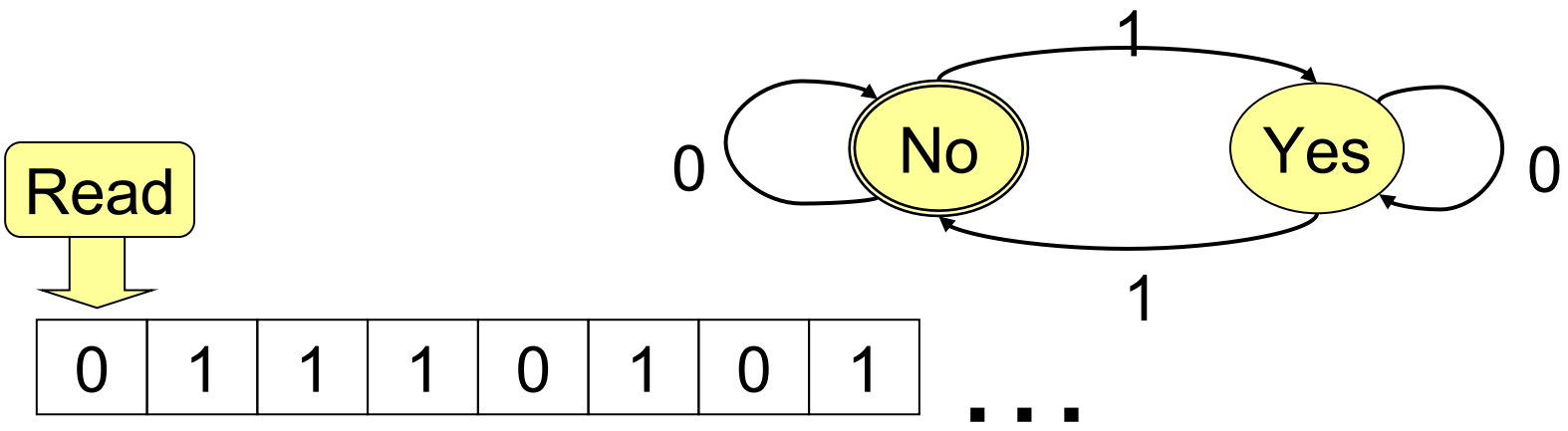
Reason 1: Computers do not process continuous data



Why study discrete math?

Reason 2: Computers aren't actually all that smart, they are just deterministic functions that map discrete inputs to discrete outputs

Example: Does a given string contain an odd number of 1s?



Why study discrete math?

In general: Discrete mathematics allows us to better understand computers and algorithms

```
function fib(int n)
  if(n == 0 || n == 1)
    return 1;
  else
    return fib(n-1) + fib(n-2);
```

```
function fib(int n)
  int first = 0;
  int second = 1;
  int tmp;
  for(i = 1 to n)
    tmp = first + second;
    first = second;
    second = tmp;
  end for
  return first;
```

Tentative Syllabus

- Logic and proofs
- Sets
- Functions
- Algorithms and analysis
- Integers, modular arithmetic, cryptography
- Induction
- Relations

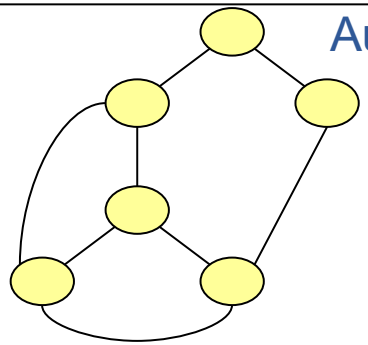
Are these topics really useful?

Logic and proofs

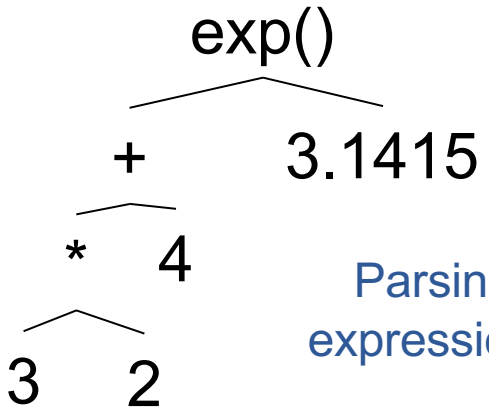
```
grant(X, projector) :- role(X, presenter), located(X, 104)
located(adam, 104)
role(adam, presenter)

=> ?grant(adam, projector)
=> true
```

Automated reasoning, AI, security



Verifying data structures and hardware

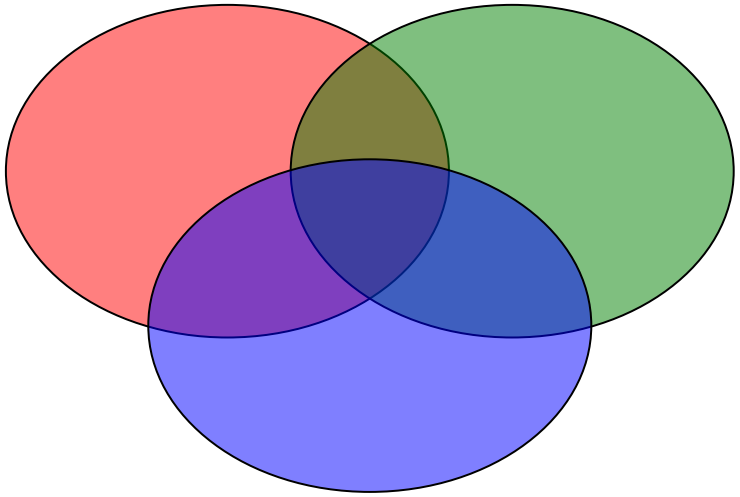


Parsing expressions

```
function fib(int n)
  int first = 0;
  int second = 1;
  int tmp;
  for(i = 1 to n)
    tmp = first + second;
    first = second;
    second = tmp;
  end for
  return first;
```

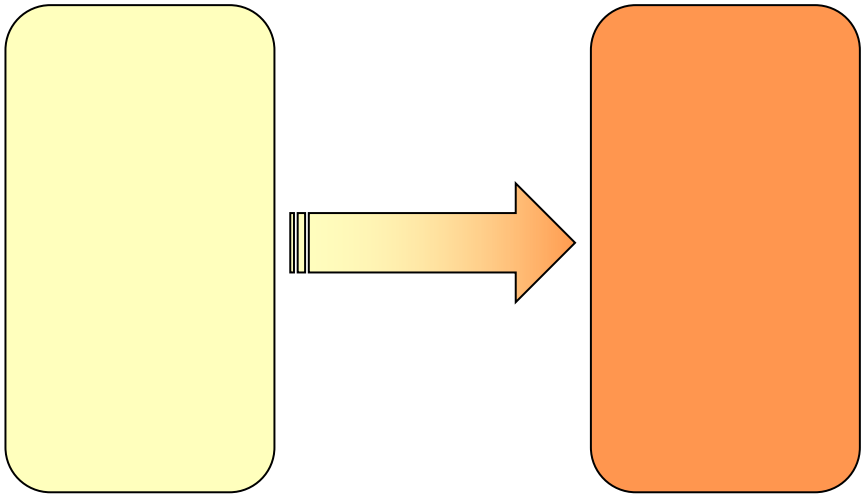
Algorithm and protocol analysis

Sets

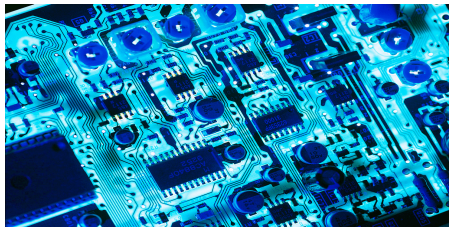


Sets define collections of objects…

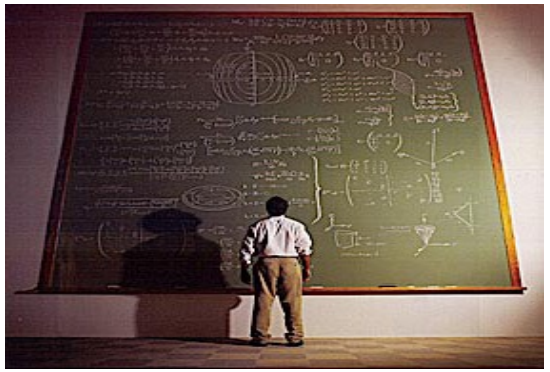
… and give us a means of reasoning about the relationships between objects



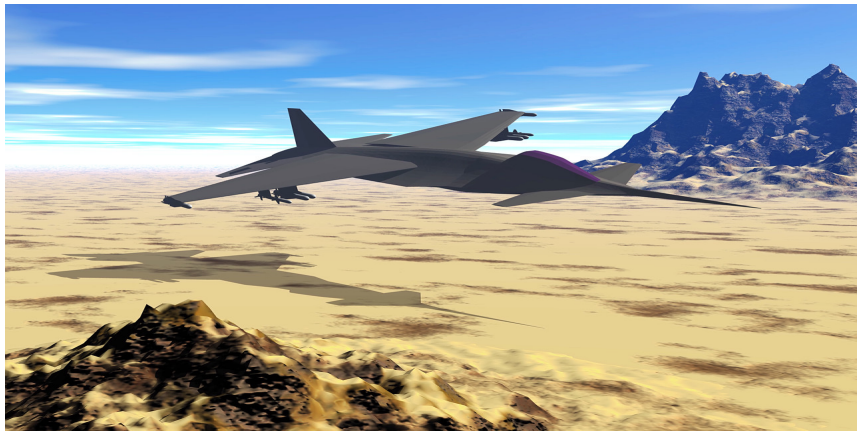
Functions



Hardware design

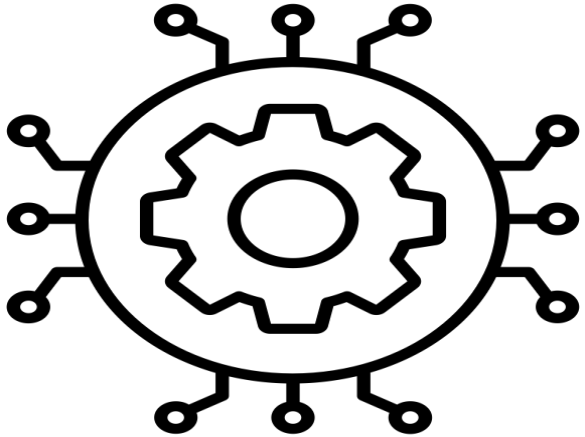


Theory of computation



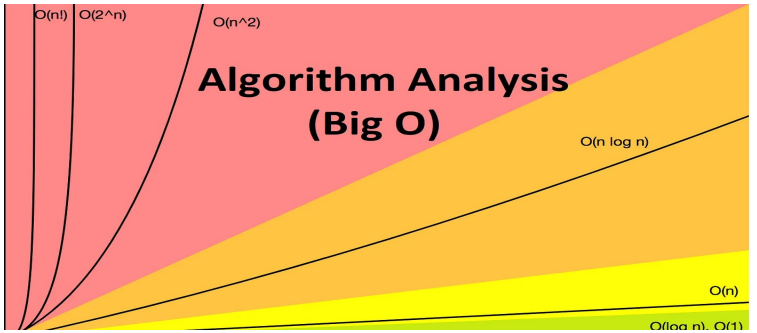
Computer graphics

Algorithms and analysis



Studying algorithms helps us write better code…

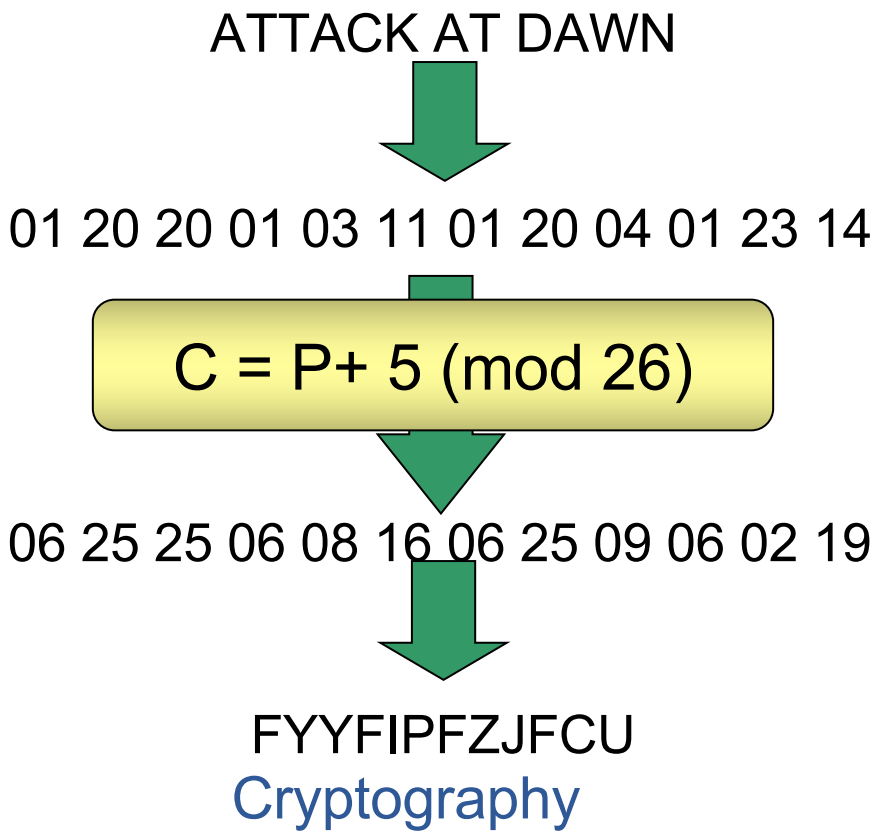
… and algorithm analysis helps us determine which approaches scale best



Integers and Modular Arithmetic

```
  0111 0101 0110 1011
+ 0101 1001 1110 0001
  1100 1111 0100 1100
```

Binary arithmetic and bitwise operations

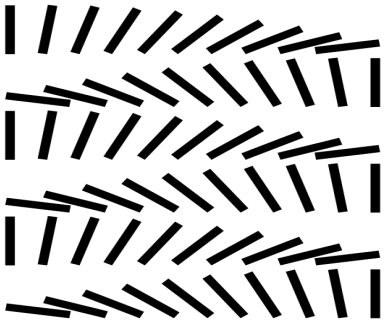


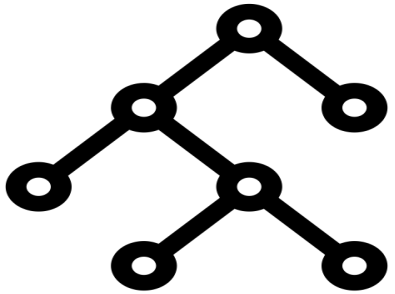
Cryptography

Induction

Induction is a proof technique that helps us reason about infinite objects…

… and processes…

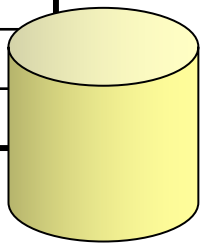




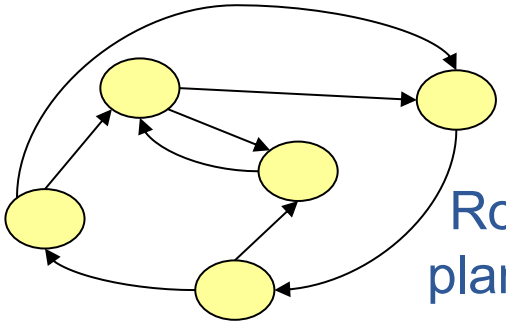
… and data structures!

Relations

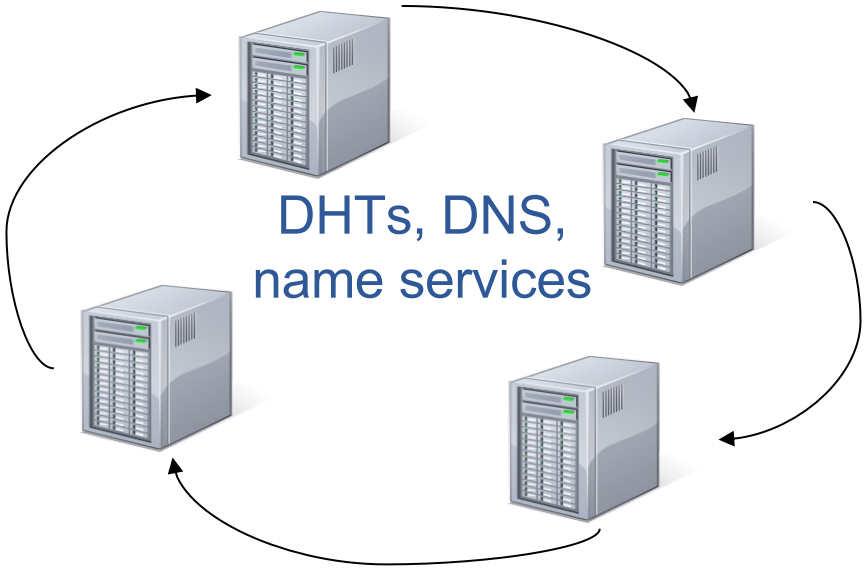
Name	Age	Phone
Alice	19	555-1234
Danielle	33	555-5353
Zach	27	555-3217
Charlie	21	555-2335



Relational databases



Route planning



DHTs, DNS, name services

Syllabus, redux

- Logic and proofs
- Sets
- Functions
- Algorithms and analysis
- Integers, modular arithmetic, cryptography
- Induction
- Relations

Are these topics really useful?

Yes

Mastering discrete mathematics requires practice!

- Succeeding in this class requires practicing the skills that we will acquire, thinking critically, and asking questions
  - We are practicing clear and precise communication in the language of mathematics and logic—be specific!

- Keys to success:
  - Attend class and take notes
  - Do your homework
  - Work extra problems when you're unsure
    - Solutions to odd-numbered exercises are provided in textbook
  - Go to your recitation every week
  - Take advantage of office hours

What should I do now?

1. Check your Canvas notification settings
2. Read the chapter for next lecture
3. Decide if you need Inclusive Access, and opt out if not
4. Install Top Hat if you plan to use the mobile app
    - Then, wait to be added to the course on TH
5. Watch for a Gradescope invitation, where you'll submit recitation and homework assignments
6. Email me if you have any special circumstances that you may need accommodated

Final thoughts

- Our goal is to prepare you to be stronger computer scientists by:
  - Exploring the formal underpinnings of computer science
  - Developing critical thinking skills
  - Articulating ties between theory and practice

- Next: Propositional logic