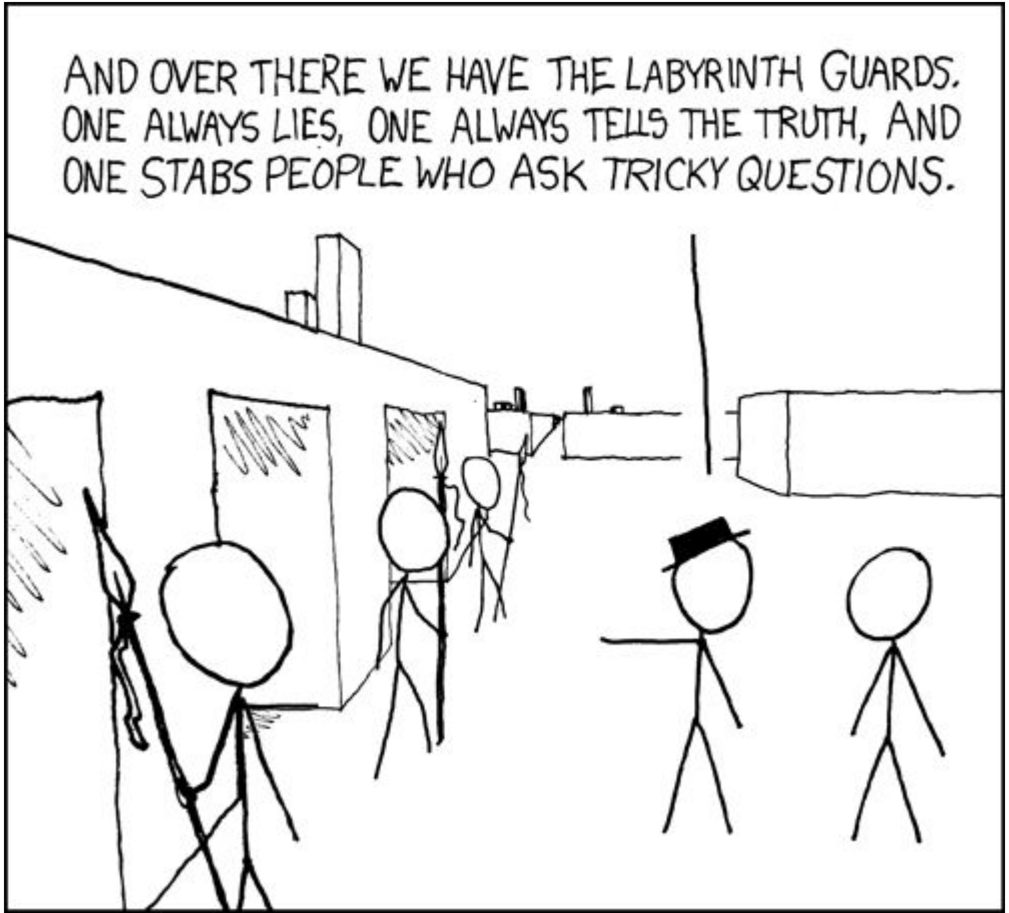
University of Colorado **Boulder**

CSCI 2824: Discrete Structures

Fall 2018

Tony Wong


AND OVER THERE WE HAVE THE LABYRINTH GUARDS. ONE ALWAYS LIES, ONE ALWAYS TELLS THE TRUTH, AND ONE STABS PEOPLE WHO ASK TRICKY QUESTIONS.

Lecture 1: Course Introduction and Intro. to Binary

What does “Discrete Structures” mean?

What does “Discrete Structures” mean?

It’s the computer sciency way of saying…   “**discrete math**”

What does “Discrete Structures” mean?

It’s the computer sciency way of saying…   “**discrete math**”

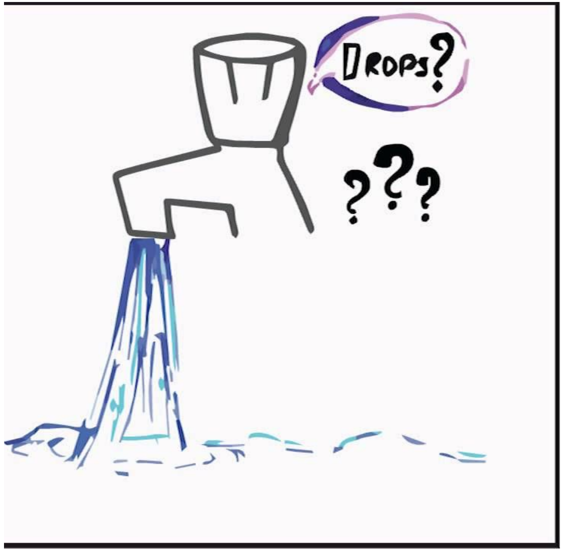
Okay… then what is “**discrete math**”?

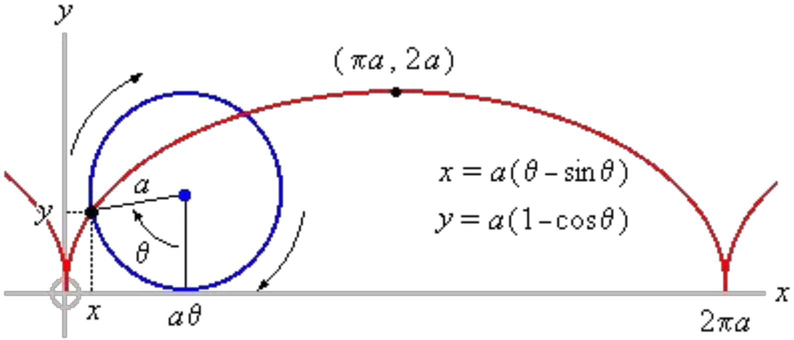
What does “Discrete Structures” mean?

It’s the computer sciency way of saying… “**discrete math**”

Okay… then what is “**discrete math**”?

Well, there’s *continuous* math
- like derivatives and integrals
- or the flow of water out of a faucet





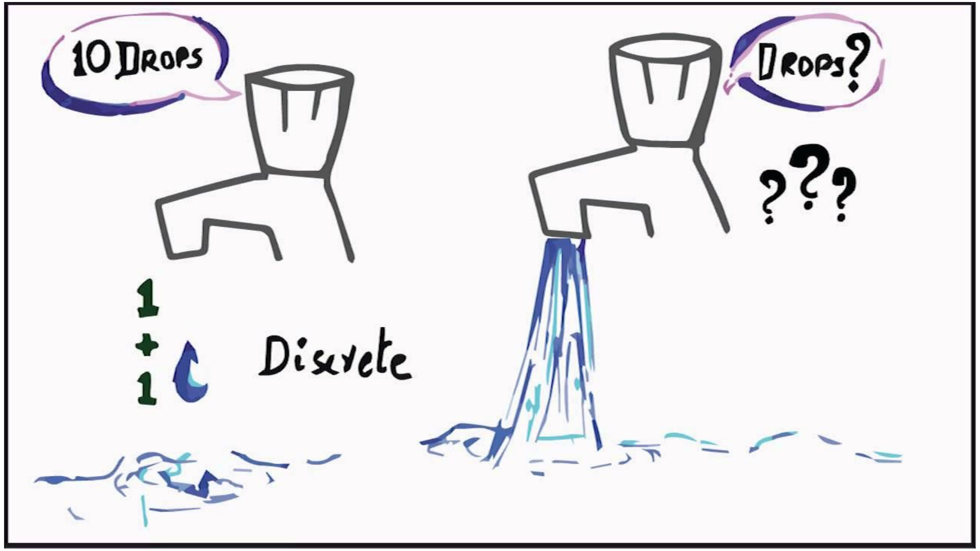
What does “Discrete Structures” mean?

It’s the computer sciency way of saying… “**discrete math**”

Okay… then what is “**discrete math**”?

Well, there’s *continuous* math
- like derivatives and integrals
- or the flow of water out of a faucet

and then there’s *discrete* math
- like counting, sorting, enumeration
- or individual droplet of water



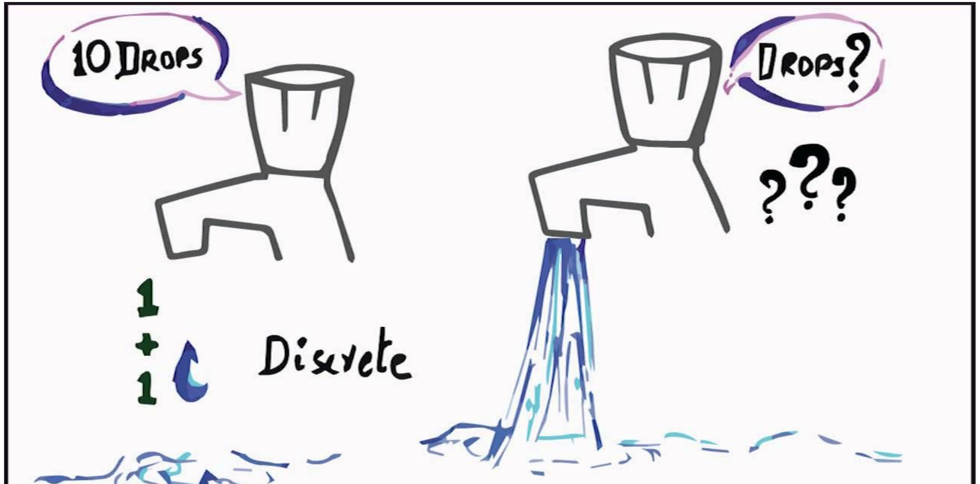
What does “Discrete Structures” mean?

It’s the computer sciency way of saying… “**discrete math**”

Okay… then what is “**discrete math**”?

Well, there’s *continuous* math
- like derivatives and integrals
- or the flow of water out of a faucet

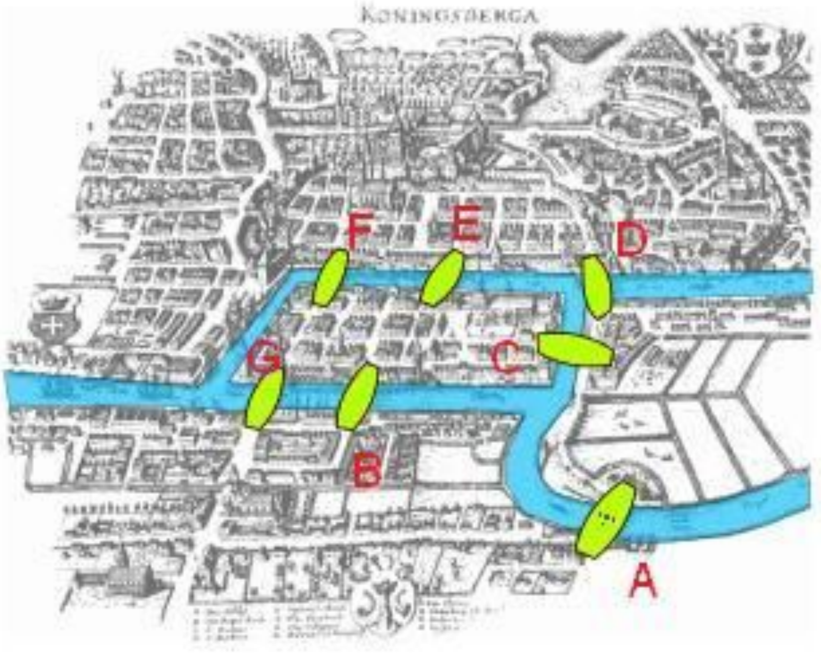
and then there’s *discrete* math
- like counting, sorting, enumeration
- or individual droplet of water



Graph theory: Bridges of Konigsberg

The city of Konigsberg has two islands formed by a river with seven bridges connecting the islands and the mainland.

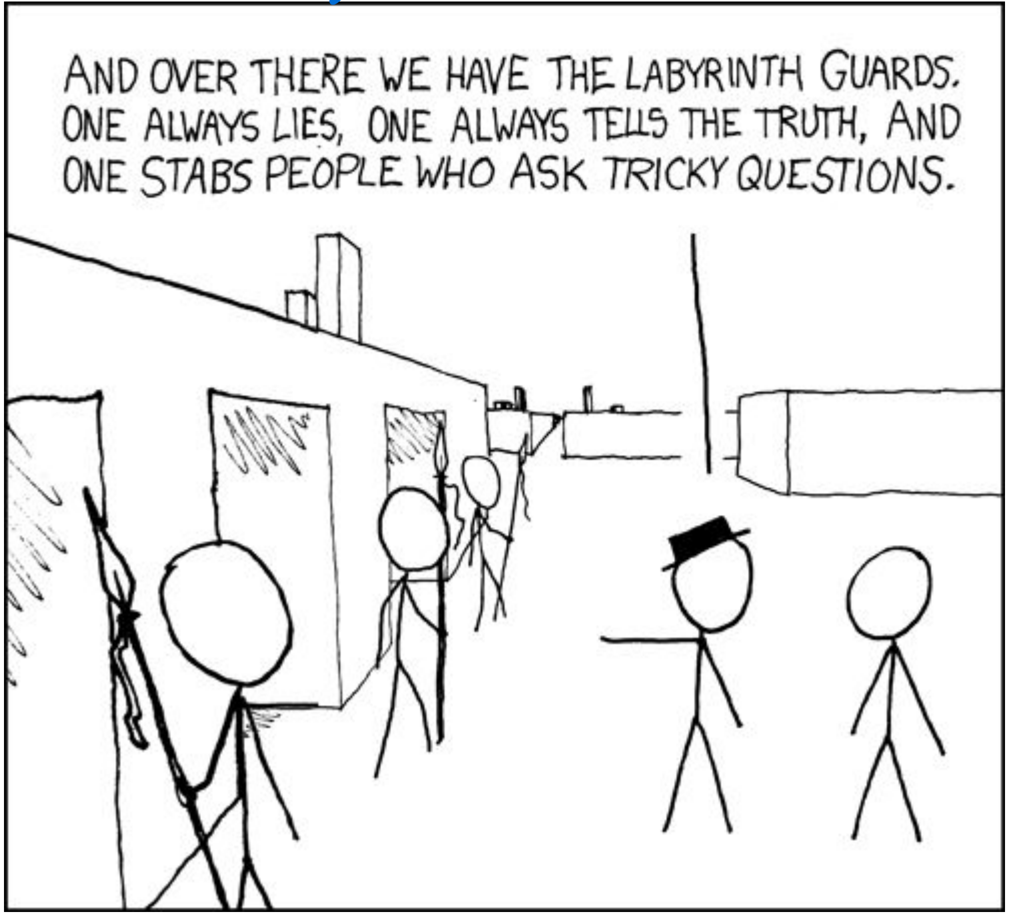
Is there a path that traverses each bridge exactly once?



Logic

“Two doors” riddle:

xkcd

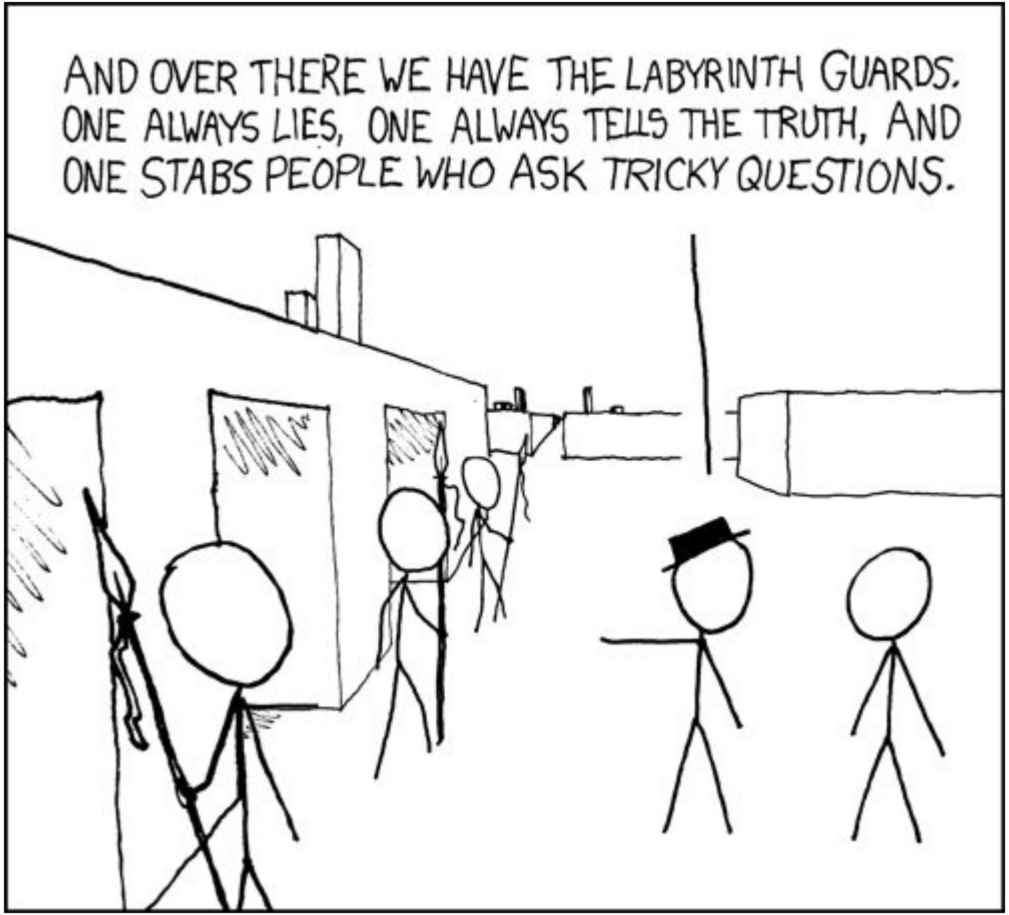

AND OVER THERE WE HAVE THE LABYRINTH GUARDS. ONE ALWAYS LIES, ONE ALWAYS TELLS THE TRUTH, AND ONE STABS PEOPLE WHO ASK TRICKY QUESTIONS.

Logic

“Two doors” riddle:

- Two doors, guarded by two guards.
- One door goes where you want to, but the other leads to certain death.
- One always lies, and one always tells the truth.

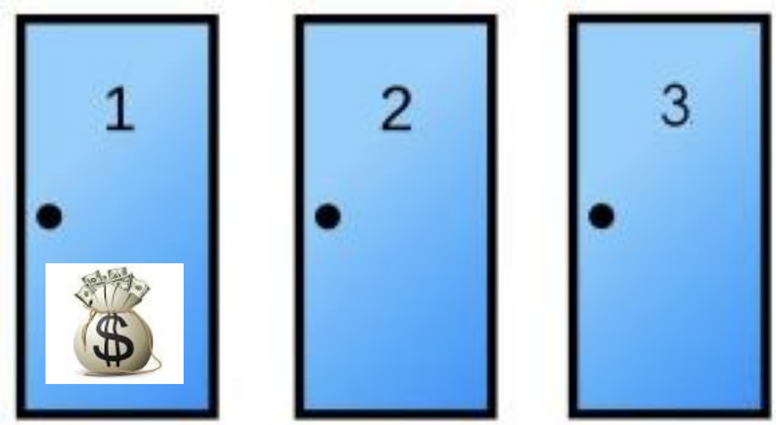
- **How can you ask** <u>**only one of them**</u> <u>**only one question**</u> **to discover which door is which?**


AND OVER THERE WE HAVE THE LABYRINTH GUARDS. ONE ALWAYS LIES, ONE ALWAYS TELLS THE TRUTH, AND ONE STABS PEOPLE WHO ASK TRICKY QUESTIONS.

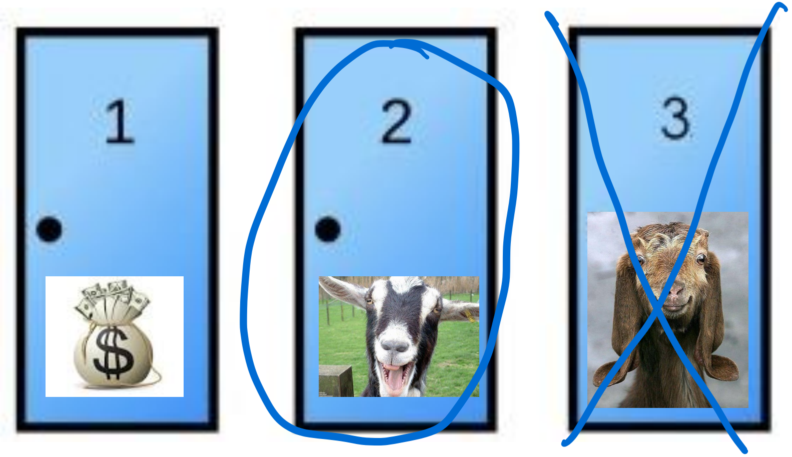
Discrete Probability

Three doors problem

- There are three doors.
- One has a nice prize behind it…



Discrete Probability

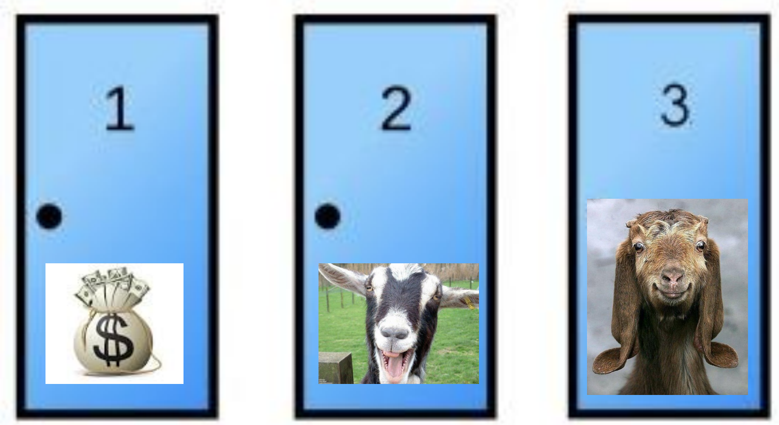


Three doors problem

- There are three doors.
- One has a nice prize behind it…
- … and the other two have goats.
- You get to pick a door and will be awarded the prize behind it.
- Then the host reveals a goat behind one of the other two doors.
- You now have the option to stick with your original door or switch.

- **Should you stick with your original door or switch? Or does it not matter?**

Discrete Probability



Three doors problem

- There are three doors.
- One has a nice prize behind it…
- … and the other two have goats.
- You get to pick a door and will be awarded the prize behind it.
- Then the host reveals a goat behind one of the other two doors.
- You now have the option to stick with your original door or switch.

- **Should you stick with your original door or switch? Or does it not matter?**

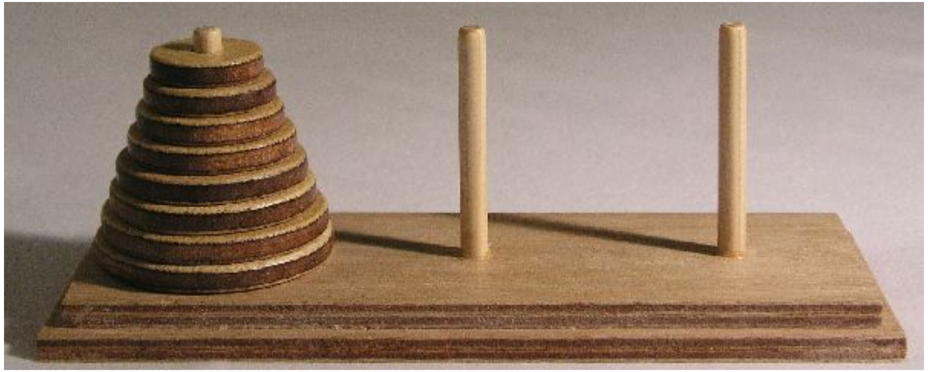
Real applications: (just some vague examples)

- *Complexity* of algorithms (100 prisoners and a light bulb)
- Anything related to data science

Recursion

Tower of Hanoi

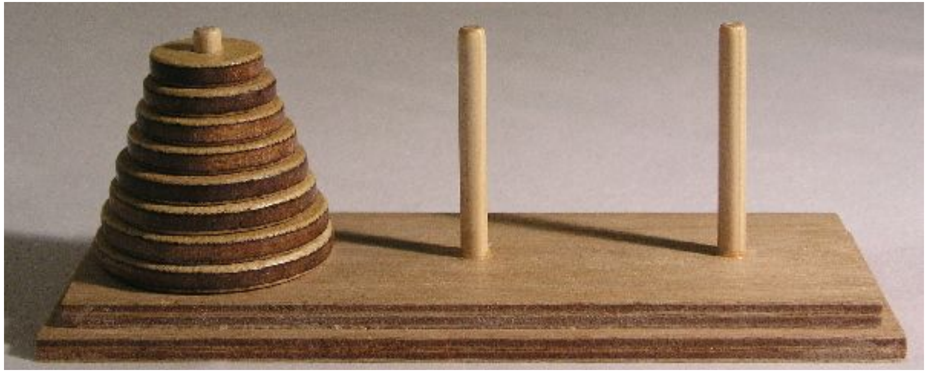
Move the stack of discs from one peg to another without placing a larger disc on top of a smaller one.



Recursion

Tower of Hanoi

Move the stack of discs from one peg to another without placing a larger disc on top of a smaller one.



Real applications:

- Recursions often yield more readable code, as well as elegant solutions
- Recursions are often frowned upon because of stack memory issues
- Still, recursions can shed some light on how a nice “for” loop can work

Changing gear: Course Logistics

What you need…

- Mathematical curiosity…

  … or at least not math-averse.

- Do some programming
  - Expect most homework assignments to include some **Python** component

Python 3

Changing gear: Course Logistics

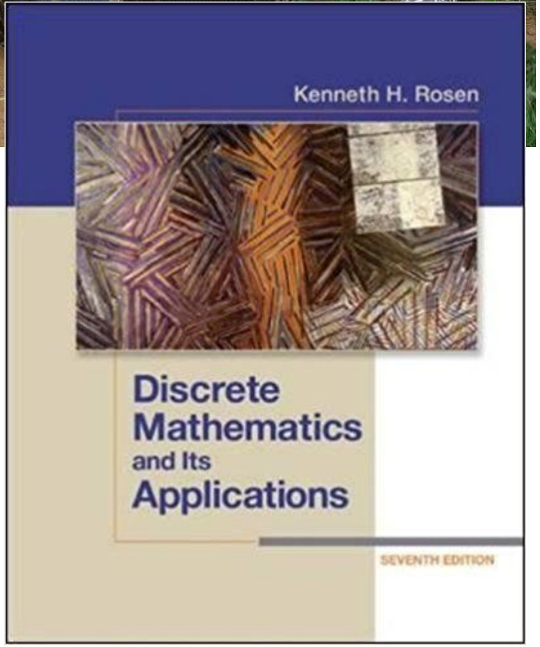
What you need…

- Mathematical curiosity…

  … or at least not math-averse.

- Do some programming
  - Expect most homework assignments to include some **Python** component

- *Some* form of the book
  - Old/international versions/PDF are okay, but make sure you have matched the appropriate sections



Changing gear: Course Logistics

moodle.cs.colorado.edu

Structure:

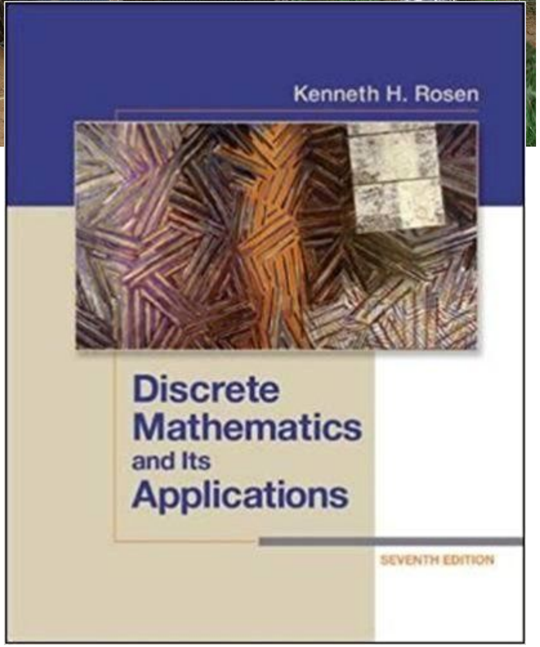
- **Weekly homework assignments** (30%)
  - Half online via Moodle, half written
  - Moodle enrollment keys:

    Tony's class: *csci2824-Tony*

    Rachel's class: *csci2824-Rachel*

searching "csci 2824"

Fall 2018?



Kenneth H. Rosen

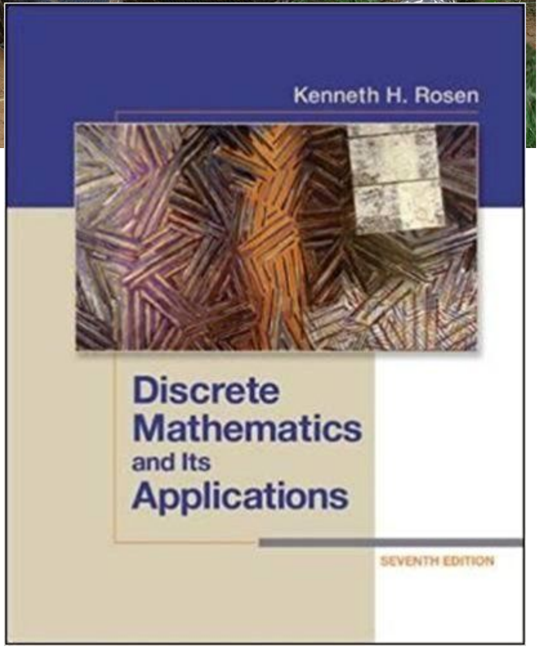
Discrete Mathematics and Its Applications

SEVENTH EDITION

Changing gear: Course Logistics

Structure:

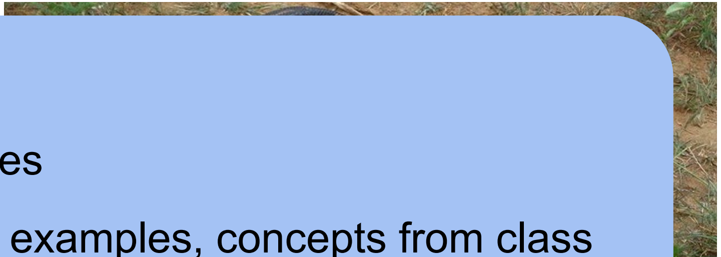
- **Weekly homework assignments** (30%)
  - Half online via Moodle, half written
- **“Quizlets”** (10%)
  - (Moodle)
- **2 midterm exams** (20% each) ← evenings on Weds! of weeks 6 & 11
- **Cumulative final exam** (20%)
  - **Need exam average of at least 55% for a C- in class**

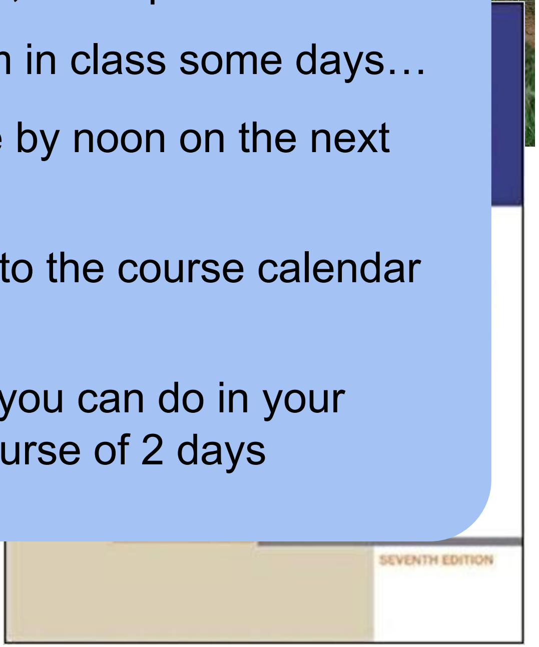


Changing gear: Course Logistics

Structure:

- **Weekly homework assignments** (3
  - Half online via Moodle, half writt
- **“Quizlets”** (10%)
  - (Moodle)
- **2 midterm exams** (20% each)
- **Cumulative final exam** (20%)
  - **Need exam average of at least**





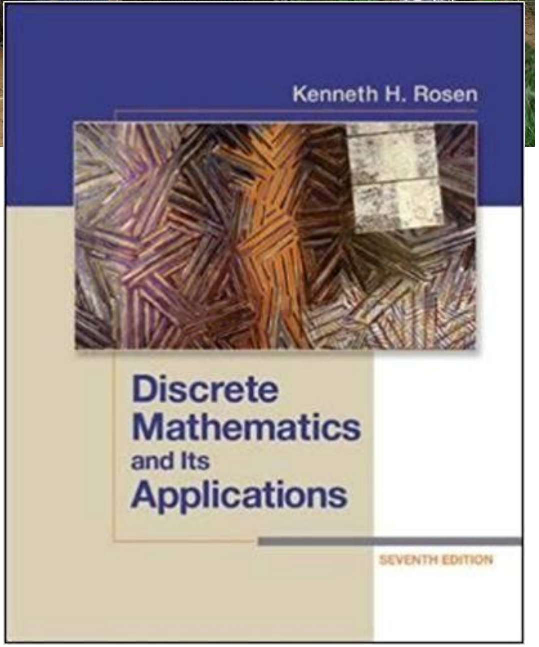
Quizlets:

- 5-15 minutes
- Build off of examples, concepts from class
- We’ll announce them in class some days…
- … and they’ll be due by noon on the next class meeting day
- We’ll also add them to the course calendar once assigned
- Like a pop quiz that you can do in your pajamas over the course of 2 days

Changing gear: Course Logistics

Structure:

- **Weekly homework assignments** (30%)
  - Half online via Moodle, half written
- **"Quizlets"** (10%)
  - (Moodle)
- **2 midterm exams** (20% each)
- **Cumulative final exam** (20%)
  - **Need exam average of at least 50% for a C- in class**
- Keep track of things via the course webpage (**Piazza**)
- Read the **syllabus** and **schedule** for more details!
  - Especially regarding due date and collaboration policies
  - Let us know about special needs as soon as possible



Kenneth H. Rosen

Discrete Mathematics and Its Applications

SEVENTH EDITION

Communicating via Piazza

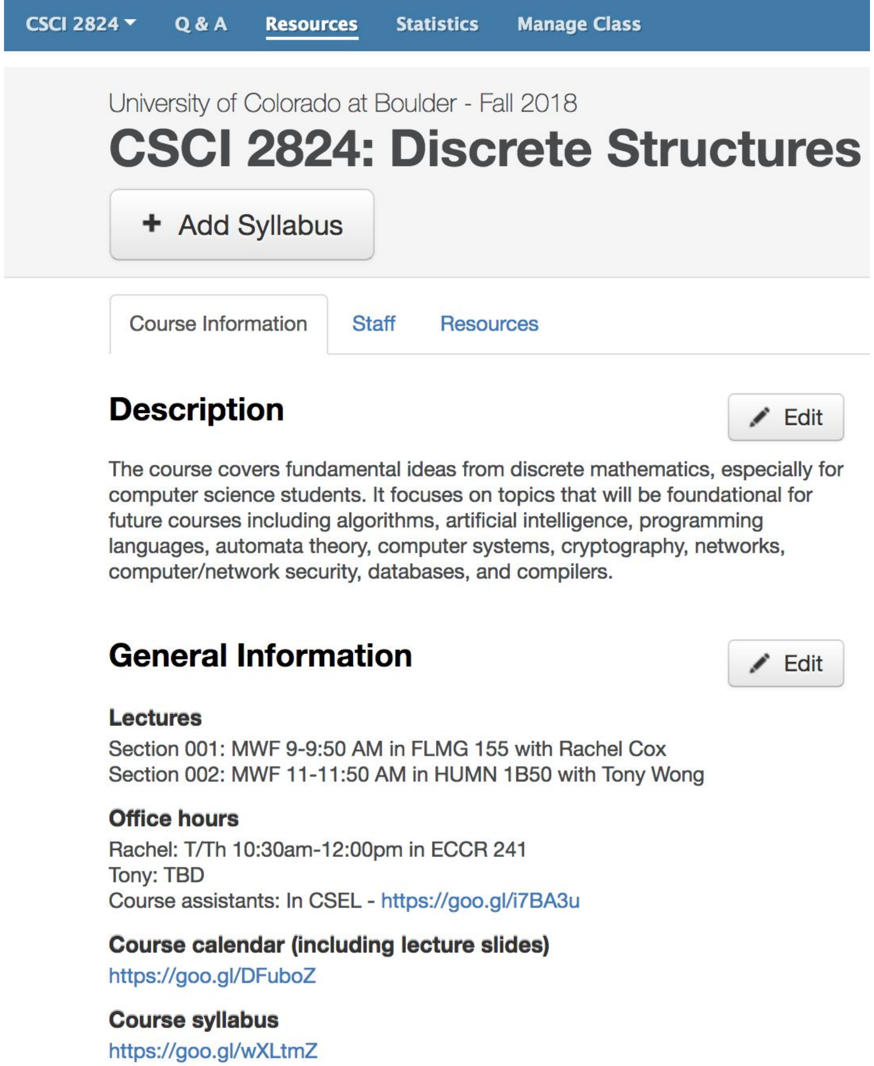
We will use Piazza to manage all communications:

https://piazza.com/colorado/fall2018/csci2824

- Announcements and course materials (notes, supplemental material, the syllabus and other logistical notes) will be posted here

* Send me & Rachel Piazza messages instead of email

→ keeps things organized



University of Colorado at Boulder - Fall 2018

CSCI 2824: Discrete Structures

Description

The course covers fundamental ideas from discrete mathematics, especially for computer science students. It focuses on topics that will be foundational for future courses including algorithms, artificial intelligence, programming languages, automata theory, computer systems, cryptography, networks, computer/network security, databases, and compilers.

General Information

Lectures

Section 001: MWF 9-9:50 AM in FLMG 155 with Rachel Cox
Section 002: MWF 11-11:50 AM in HUMN 1B50 with Tony Wong

Office hours

Rachel: T/Th 10:30am-12:00pm in ECCR 241
Tony: TBD
Course assistants: In CSEL - https://goo.gl/i7BA3u

Course calendar (including lecture slides)

https://goo.gl/DFuboZ

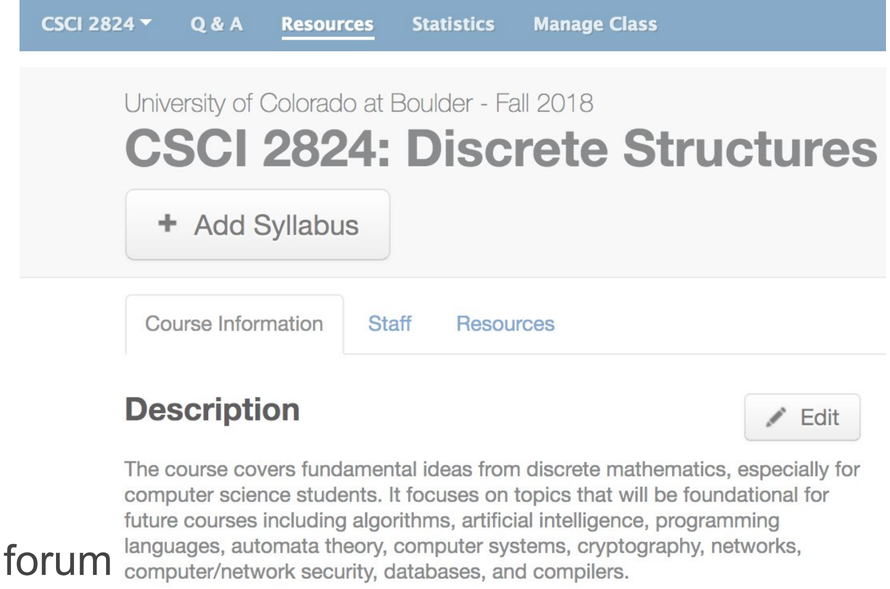
Course syllabus

https://goo.gl/wXLtmZ

Communicating via Piazza

We will use Piazza to manage all communications:

https://piazza.com/colorado/fall2018/csci2824



University of Colorado at Boulder - Fall 2018

CSCI 2824: Discrete Structures

Description

The course covers fundamental ideas from discrete mathematics, especially for computer science students. It focuses on topics that will be foundational for future courses including algorithms, artificial intelligence, programming languages, automata theory, computer systems, cryptography, networks, computer/network security, databases, and compilers.

- Announcements and course materials (notes, supplemental material, the syllabus and other logistical notes) will be posted here
- **You** (yes, **you**!) can ask questions in an open forum
  - and responses can be posted by other students or instructors/CAs/graders
- Discuss assigned work (but of course do **not** post answers/vital code)
- Read the other questions before posting your own!
- If you have a question specific to one instructor, be sure to use the appropriate salutation (e.g., “Hey Rachel, …” or “Hey Tony, …”)

“private note“

Do not send us email! Send us message via Piazza instead.

Expectations

- Come to class **prepared**, at least skim the material beforehand
- Be willing to **contribute**
    - To classroom discussion
    - As well as on Piazza
      (students helping one another is one of the reasons why we use it!)
- Be **on time**. If you are late, do not be disruptive
- If you use a **laptop, sit in the back** so you do not distract others (unless you have a documented special need)
    - More on this in a moment…
    - Handwritten notes and notes on a tablet are okay anywhere.

Expectations

Let’s talk laptops.

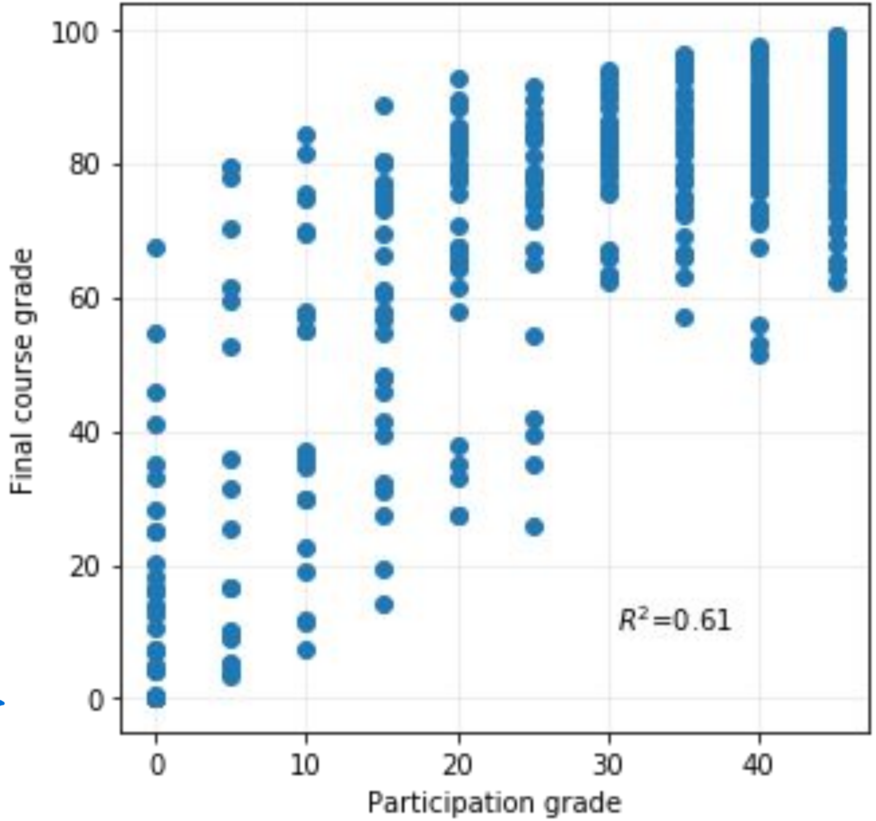
“Results showed that students who used laptops in class spent considerable time multitasking and that the laptop use posed a significant distraction to both users and fellow students. Most importantly, the level of laptop use was negatively related to several measures of student learning, including self-reported understanding of course material and overall course performance.”

http://www.sciencedirect.com/science/article/pii/S0360131506001436

Also: http://journals.sagepub.com/doi/pdf/10.1177/0956797616677314

And: http://www.sciencedirect.com/science/article/pii/S0272775716303454 (... and others…)

Expectations: Attendance



almost all failing grades

almost no failing grades

About Me

- Call me **Tony**. Or **Dr. Wong** if you're more comfortable with that
- Second year teaching in **CS**
- Before this: Postdoc at **Penn State**. And taught **Earth Science**

  Grad student in **Applied Math**. And taught Calc/Diff Eq
- Research interests:
  - Stats: Uncertainty quantification, Markov chains, (Bayesian) model calibration
  - Applications: Storm surge/sea-level projections, climate risk management

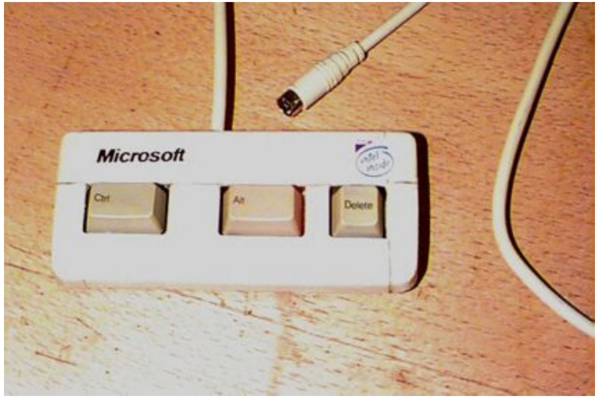
- **Office:** ECOT 623
- **Office hours:** W 3-4, Th 10-12

  and by appointment ← esp. if you want to discuss grades/etc.

Enough chit chat. Let’s get to work!
Binary Arithmetic

There is a nice overview of binary arithmetic available here:

http://www.cs.colorado.edu/~srirams/courses/csci2824-spr14/binaryLecture.html

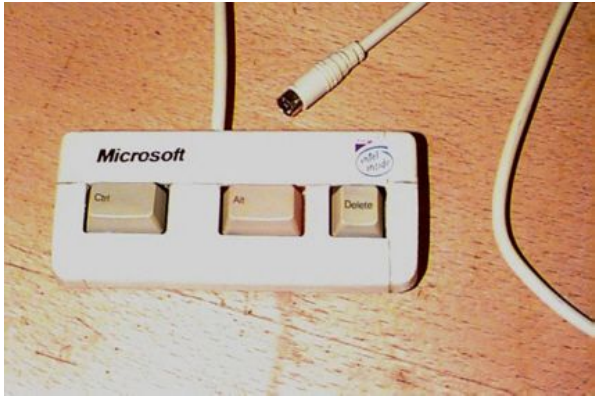


Enough chit chat. Let’s get to work!
Binary Arithmetic

There is a nice overview of binary arithmetic available here:

http://www.cs.colorado.edu/~srirams/courses/csci2824-spr14/binaryLecture.html

- We are most used to using **decimal arithmetic** (base-10)
  - Counting on your fingers or toes?
- **Why** do we care?
  - Computers are good are knowing whether something is “on” (1) or “off” (0)
  - Two options → base-2 system



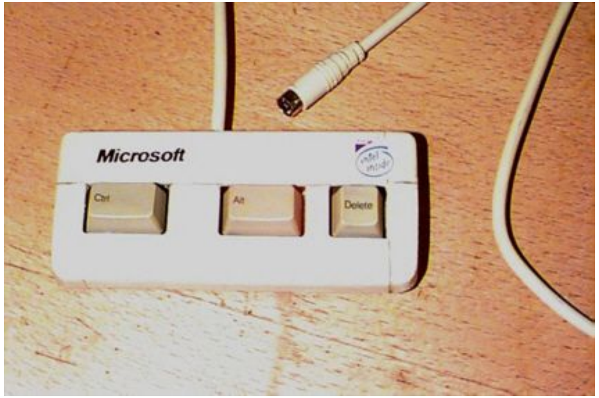
Enough chit chat. Let’s get to work!
Binary Arithmetic

There is a nice overview of binary arithmetic available here:

http://www.cs.colorado.edu/~srirams/courses/csci2824-spr14/binaryLecture.html

- We are most used to using **decimal arithmetic** (base-10)
  - Counting on your fingers or toes?
- **Why** do we care?
  - Computers are good are knowing whether something is “on” (1) or “off” (0)
  - Two options → base-2 system
- **Example:** holding down Ctrl (0/1)? Alt (0/1)? Del (0/1)?

⇒ Abort only if (111)



For example:

The decimal number 12045 implies that we have:

recipe: each digit = how much of each "ingredient"

powers of 10

$1 \rightarrow 1 \times 10{,}000 \quad (10^4)$

$+ \quad 2 \rightarrow 2 \times 1{,}000 \quad (10^3)$

$+ \quad 0 \rightarrow 0 \times 100 \quad (10^2)$

$+ \quad 4 \rightarrow 4 \times 10 \quad (10^1)$

$+ \quad 5 \rightarrow 5 \times 1 \quad (10^0)$

For another example:

The binary number 11011 follows the same pattern, but with **powers of 2** instead of 10.

So what is this in decimal form?

For example:

The decimal number 12045 implies that we have:

$$\begin{array}{l} \ 5 \times 10^{0} \\ + 4 \times 10^{1} \\ + 0 \times 10^{2} \\ + 2 \times 10^{3} \\ \underline{+ 1 \times 10^{4}} \\ = 12045 \end{array}$$

For another example:

The binary number 11011 follows the same pattern, but with **powers of 2** instead of 10.

So what is this in decimal form?

For another example: ↓↓↓

The binary number 11011 follows the same pattern, but with **powers of 2** instead of 10.

So what is this in decimal form?

$$1 \cdot 2^0 + 1 \cdot 2^1 + 0 \cdot 2^2 + 1 \times 2^3 + 1 \times 2^4$$

$$= 1 + 2 + 0 + 8 + 16$$

$$= \boxed{27}$$

For another example:

The binary number 11011 follows the same pattern, but with **powers of 2** instead of 10.

So what is this in decimal form?

We have:

$1 \text{ x } 2^0$

$+ 1 \text{ x } 2^1$

$+ 0 \text{ x } 2^2$

$+ 1 \text{ x } 2^3$

$\underline{+ 1 \text{ x } 2}^4$

$= 1 + 2 + 0 + 8 + 16$

$= 27$

Some general rules:

1. We chop off leading 0s
    a. Just like the decimal number 0123 is written as 123, the binary number 0011011 is written as 11011.
2. Gosh, it would be cumbersome to keep writing “the decimal number *this* and the binary number *that*”. **Surely** there must be an easier way!

Some general rules:

Quizlet 0: Watch Airplane

1. We chop off leading 0s
   a. Just like the decimal number 0123 is written as 123, the binary number 0011011 is written as 11011.
2. Gosh, it would be cumbersome to keep writing “the decimal number *this* and the binary number *that*”. **Surely** there must be an easier way!
   a. There is! We write the base (10 for decimal, 2 for binary, e.g.) as a **subscript** next to the number.
   b. So from the previous example,

$$11011_2 = 27_{10}$$

Converting the number N from decimal to binary

- Start from right (the ones place) and move to the left.
- If the first digit is odd, then the first bit is a 1. Otherwise, it's a 0.
  - Reset N → (N-1) / 2 if N was odd.
  - Reset N → N/2 if N was even.
- Move to the next bit to the left.
- Repeat procedure until N either is 0 or 1.

Example: Convert the number 47 from decimal to binary.

7 odd → first bit is 1

→ procede w/ $\frac{47-1}{2} = \underline{\underline{23}}$

3 odd → second bit is 1

→ procede w/ $\frac{23-1}{2} = \underline{\underline{11}}$

1 odd → third bit is 1

→ go w/ $\frac{11-1}{2} = 5$

HERE

Converting the number N from decimal to binary

- Start from right (the ones place) and move to the left.
- If the first digit is odd, then the first bit is a 1. Otherwise, it's a 0.
  - ○ Reset N → (N-1) / 2 if N was odd.
  - ○ Reset N → N/2 if N was even.
- Move to the next bit to the left.
- Repeat procedure until N either is 0 or 1.

Example: Convert the number 47 from decimal to binary.

First digit (7) is odd, so first bit is 1.
Proceed with (47-1) / 2 = 23

First digit (3) is odd, so second bit is a 1.
Proceed with (23-1) / 2 = 11

First digit (1) is odd, so third bit is a 1.
Proceed with (11-1) / 2 = 5

5 is odd, so fourth bit is a 1
Proceed with (5-1) / 2 = 2

2 is even, so sixth bit is a 0
Proceed with 2/2 = 1

End with a 1: $\mathbf{47_{10} = 101111_2}$

Converting the number N from decimal to binary

Example: Convert 2824 from decimal to binary.

Converting the number N from decimal to binary

Example: Convert 2824 from decimal to binary.

- First digit is even, so first bit is a 0.
    - Next iteration, we have 2824/2 = 1412.

Converting the number N from decimal to binary

Example: Convert 2824 from decimal to binary.

- First digit is even, so first bit is a 0.
    - Next iteration, we have 2824/2 = 1412.
- First digit is even, so second bit is a 0.
    - Next iteration, we have 1412/2 = 706.
- First digit is even, so third bit is 0.
    - Next iteration, we have 706/2 = 353.
- First digit is odd, so fourth bit is 1.
    - Next iteration, we have (353-1)/2 = 176.
- First digit is even, so fifth bit is 0.
    - Next iteration, we have 176/2 = 88.
- First digit is even, so sixth bit is 0.
- … eventually, find $2824_{10} = 101100001000_2$

Introduction and Binary Intro.

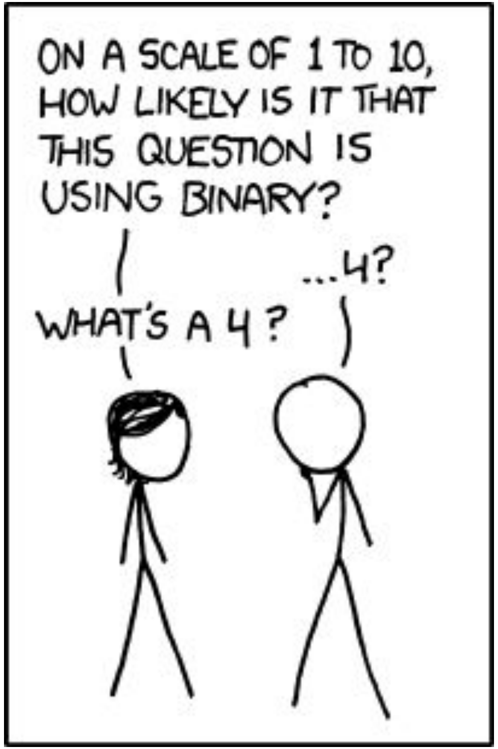
Recap:

Today, we…

- Confirmed that we are in the right classroom at the right time
- Learned what the goals and logistics for this class are
- Learned how to represent decimal numbers as binary, and vice versa

Next time:

- We **dig deeper into binary**, and
- Play with Python!


ON A SCALE OF 1 TO 10, HOW LIKELY IS IT THAT THIS QUESTION IS USING BINARY?
WHAT'S A 4?
...4?

Bonus material!