CSC 320
Foundations of Computer Science

Lecture 1

Instructor: Dr. Ulrike Stege

Territory Acknowledgement

We acknowledge and respect the lək̓ʷəŋən peoples on whose traditional territory the university stands and the Songhees, Esquimalt and W̱SÁNEĆ peoples whose historical relationships with the land continue to this day.

This meeting will be recorded

"Please be aware our sessions are being screen-recorded to allow students who are not able to attend to watch later and will be posted in Brightspace."

Lectures & Tutorials

- **Lectures** are taught in-person
- Each lecture will be streamed via zoom
- Lectures are recorded and posted
- I **strongly recommend** to attend every lecture live (in-person or online)
- **Tutorials** are taught **online**
- Tutorials are not recorded
- I **strongly recommend** to attend the weekly tutorials (starting week of May 08)

If you are worried, feel **unwell** or have **symptoms**, **participate online**. We are working hard on providing a good hybrid experience for you.

Operation & Team

- Course on Brightspace; assessment on Crowdmark
- Regularly check Brightspace & Crowdmark for reading materials, lecture slides, deadlines, assessments
- Instructor: Dr. Ulrike Stege
- TAs: Sajed Karimy, Ivan Perez Martell, Luke Trinity, Stella Zarei

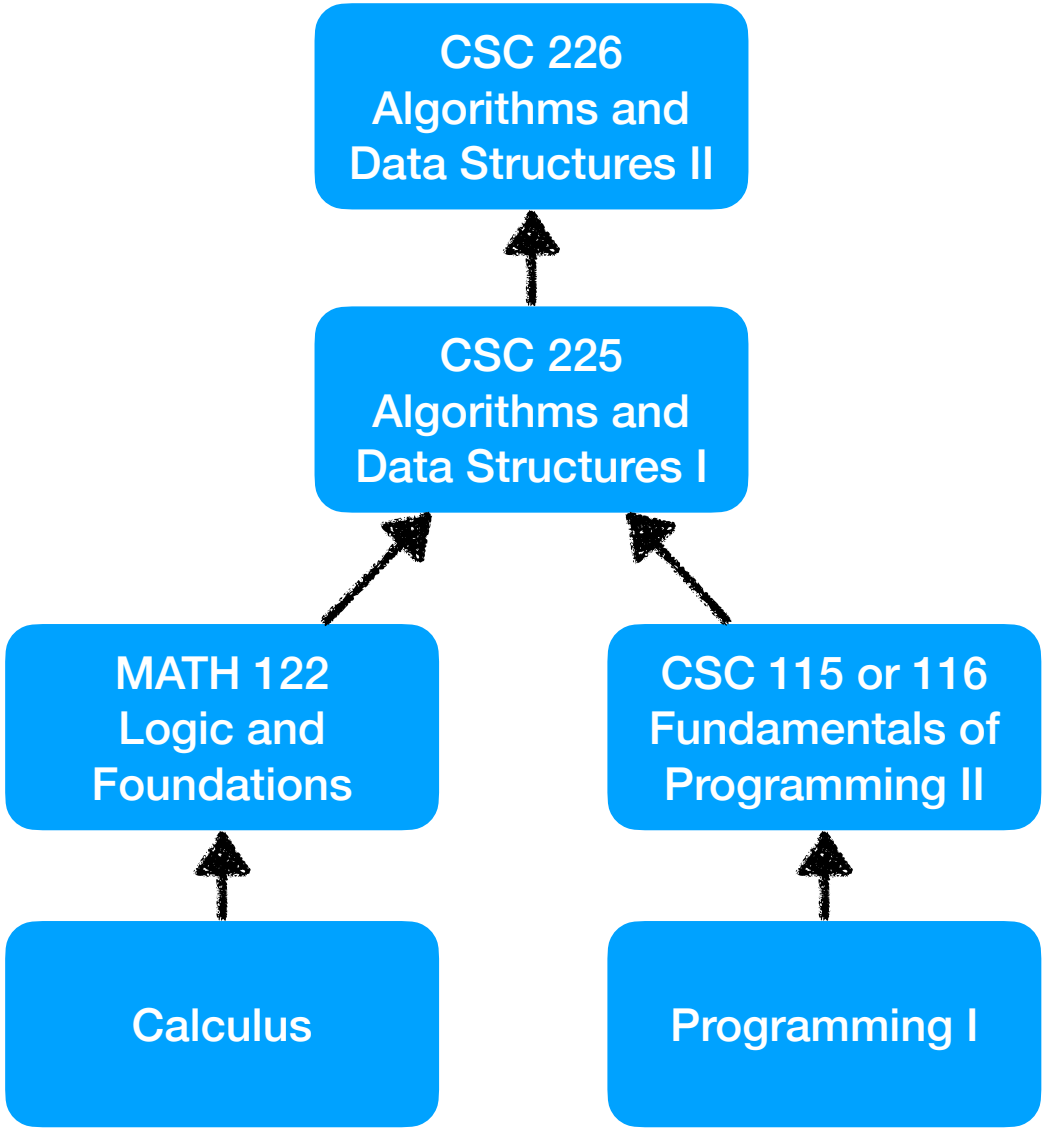
Office hours—in-person or online

First Office Hours: May 8

- Instructor's office hours (times: subject to change)
  - Zoom link on Brightspace (same as for lecture)
  - Mondays: 2:30pm—3:30pm
  - Thursday: 3:30pm—4:30pm
  - ECS 624

Poll: **http://etc.ch/qn45**

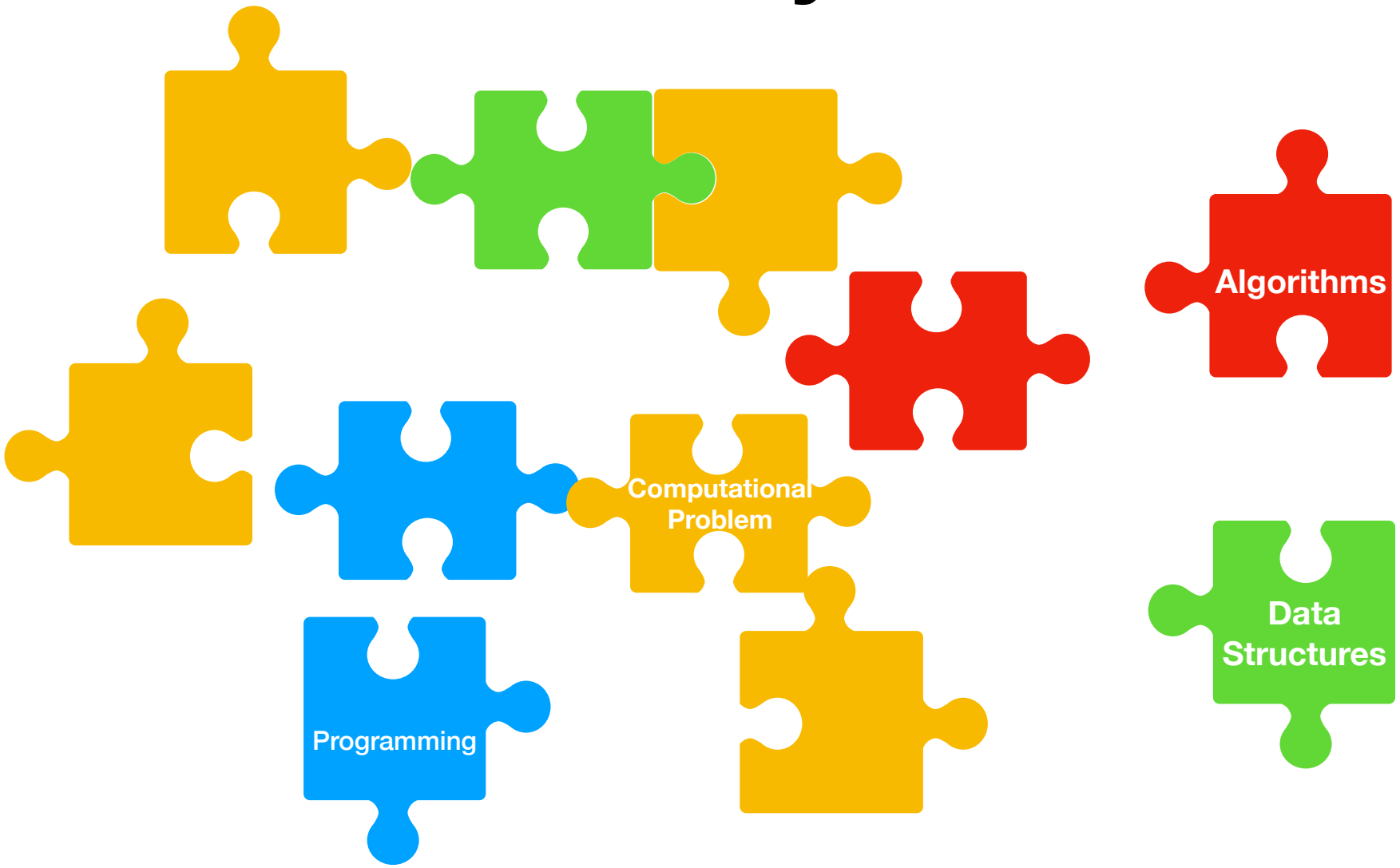
Pre-requisites for CSC 320—Foundations of Computer Science



Foundations of Computer Science: Why this course?

- What can we do with (classical) computers?
- What are the limitations of (classical) computing?

Foundations of Computer Science: Why this course?



Foundations of Computer Science—What's that?

- Study **fundamental nature** of (classical) **computation**
  - What **problems** are **solvable** using a (classical) **computer**?
- We **need to know** answers to
  - What exactly is a **problem**?
  - What exactly is a **solution to a problem**?
  - How do we **model** a **computer**?

Note

In our context, solving a problem means: answer the question posed in the problem exactly/ perfectly

Remaining Outline for today's lecture

- Expected Course and Learning Outcomes
- Administrative announcements
- Academic Integrity
- Some basics and background

CSC 320 Learning Outcomes

- An understanding of the following

  - What are the fundamental capabilities and limitations of computers?

  - What makes some problems computationally hard and others easy?

- Topics discussed are in the following areas

  - Automata theory

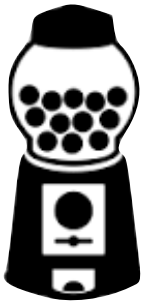
  - Computability theory

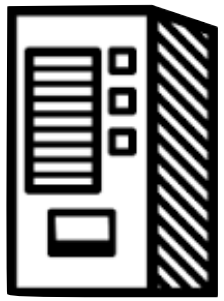
  - Complexity theory

CSC 225, 226: Theory of algorithms

Computational models from three perspectives

Automata Theory: Examples of Automata





- Finite automata
  - Used in text processing, compilers, hardware design, appliances, …
- Pushdown automata (or context-free grammars)
  - Used in programming languages, artificial intelligence
- Turing machines
  - Model of our “conventional” computer

What problems can and what problems cannot be solved?
What problems can be solved but are hard?
What problems are easy?

Some Terminology

- **Unsolvable** or **undecidable**: problems not solvable using our (standard/classical) computing model (independent of resource limitations)

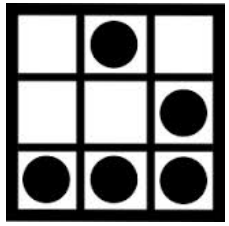
- **Easy** to solve: problems solvable in polynomial time (using our standard/classical computing model)

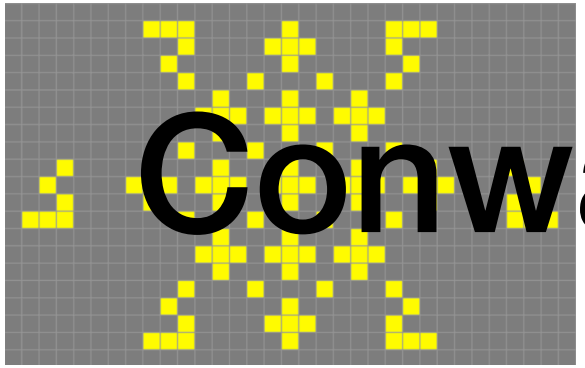
- **Hard** to solve: problems solvable (in theory, using our standard computing model) but (likely) not solvable in polynomial time (using our standard/classical computing model)

After completion of the course: You will be able to answer questions such as

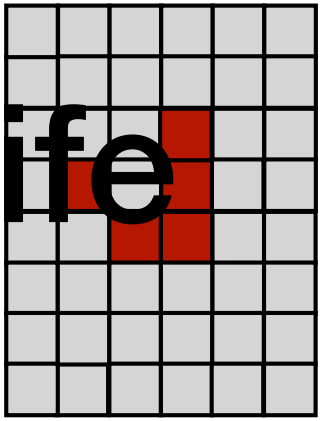
- Are there any problems that—no matter how powerful the computer/no matter how much time you have—your computer would not not be able to solve?

- Is it decidable (ie., is there a computation that allows us to decide), in **Conway's Game of Life**, whether given an initial pattern and another pattern, the latter pattern can ever appear when starting from the initial one?



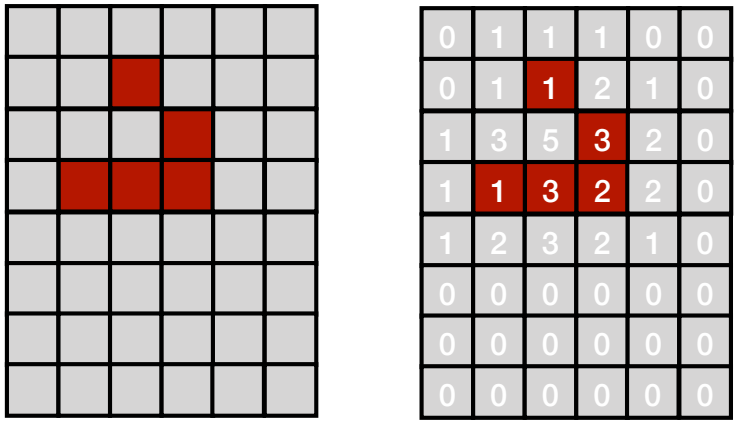


Conway's Game of Life

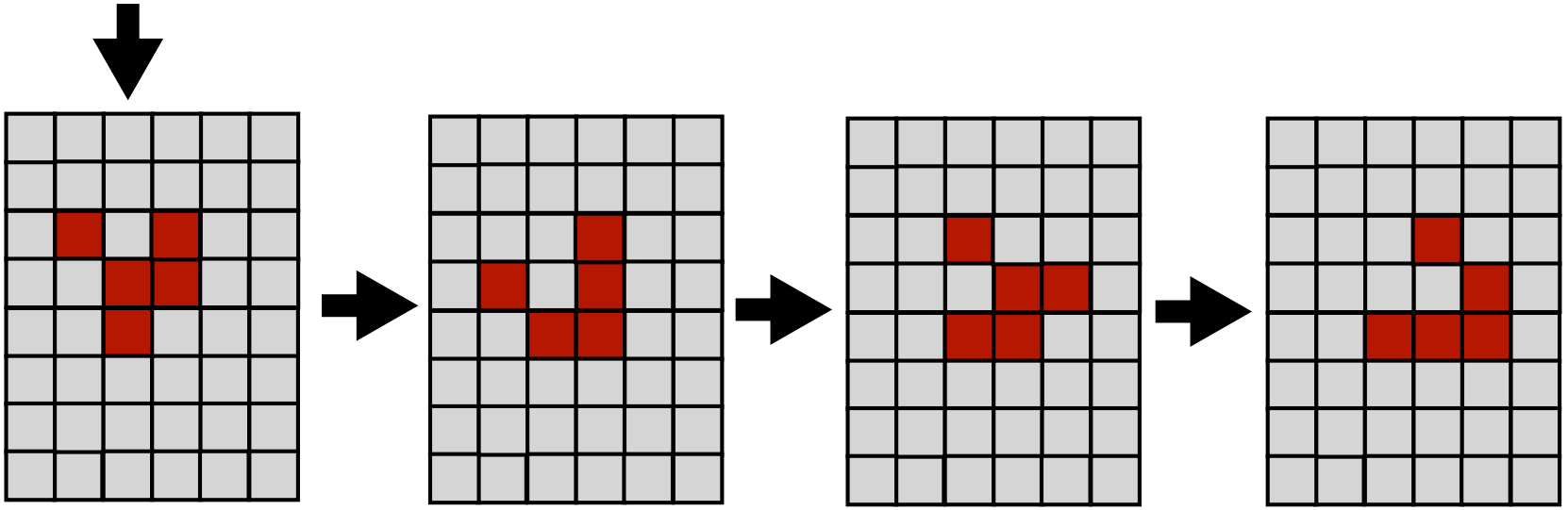


- Example of a cellular automaton
- Played on an infinite two-dimensional rectangular grid of cells
- Each cell can be either alive or dead
- Status of each cell c changes each turn of the game (*generation*) depending on the statuses of c's (eight) neighbors: cells that touch c, either horizontal, vertical, or diagonal from that cell
- Initial pattern: first generation
- Next generation: evolves from applying given *rules* simultaneously to *every cell* on the game board
  - If the cell is alive, then it stays alive if it has either exactly 2 or exactly 3 live neighbours
  - If the cell is dead, then it is "born" only if it has exactly 3 live neighbours

Example(s)

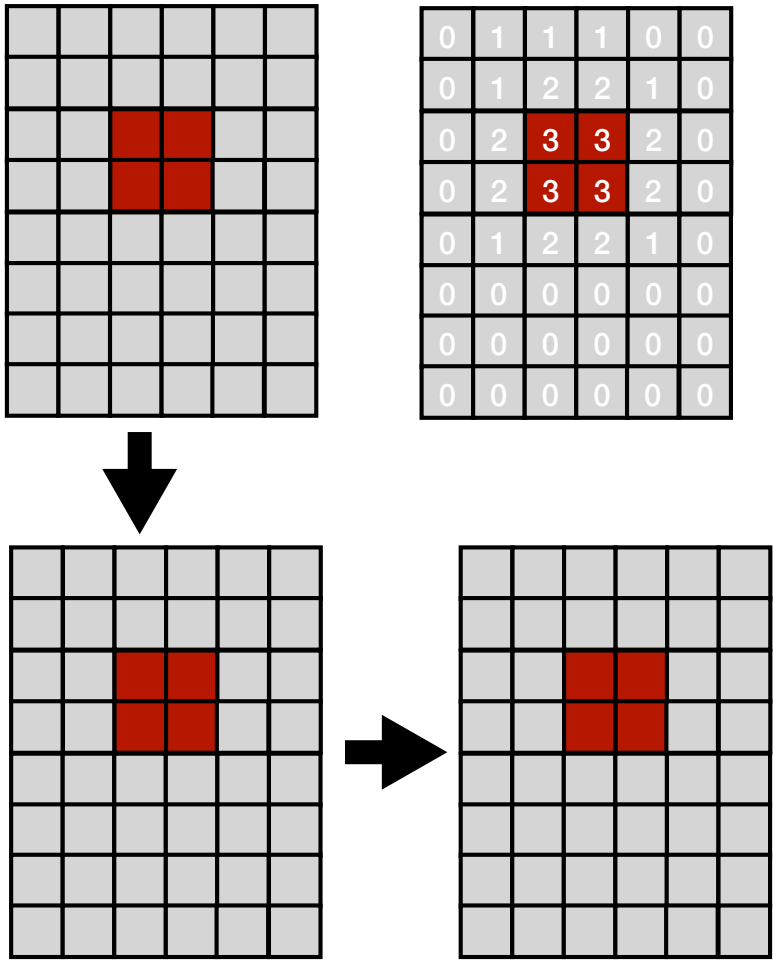


- If the cell is alive, then it stays alive if it has either exactly 2 or exactly 3 live neighbours
- If the cell is dead, then it is “born” only if it has exactly 3 live neighbours



Check out: https://playgameoflife.com

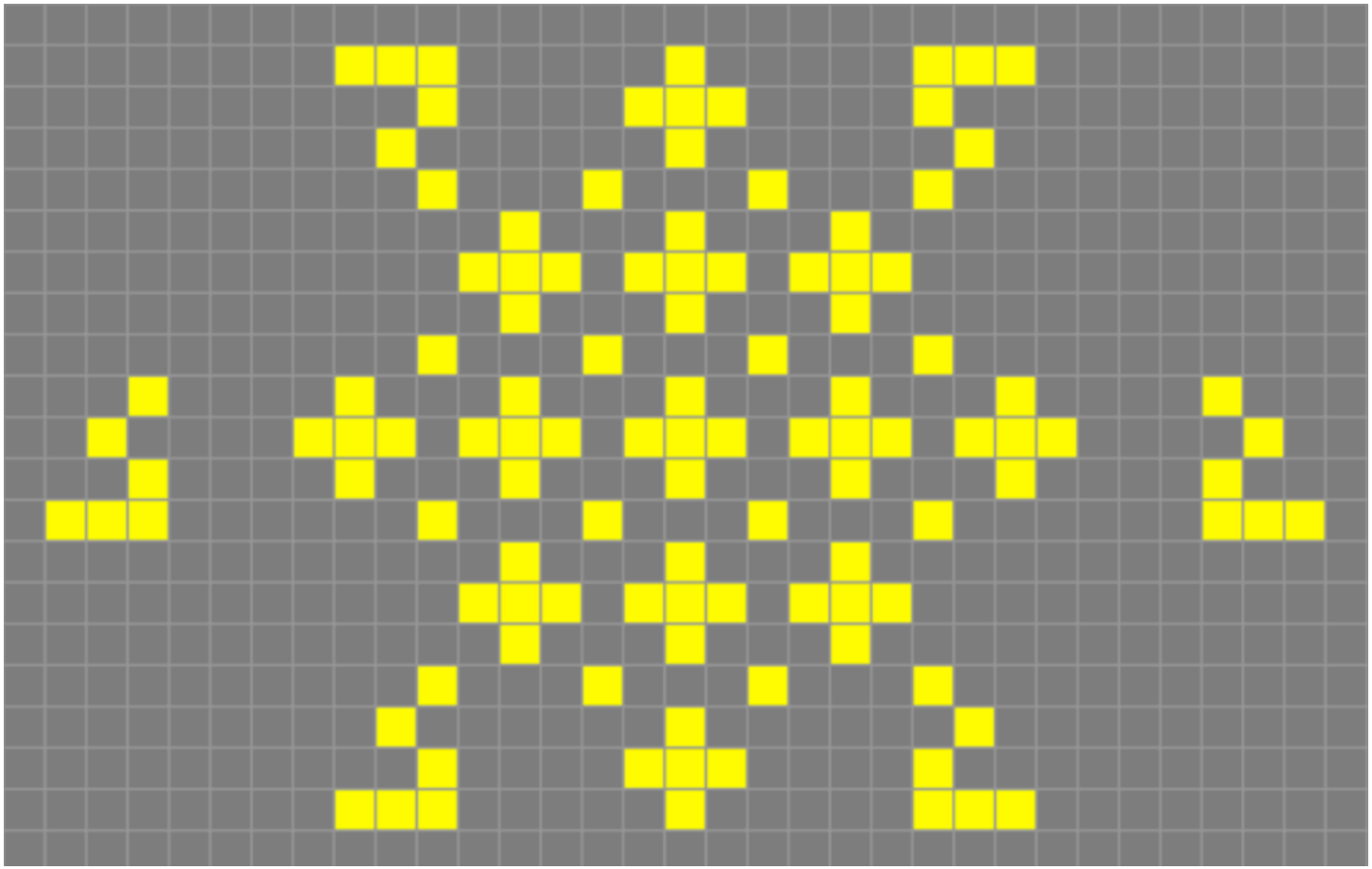
Example(s)



- If the cell is alive, then it stays alive if it has either exactly 2 or exactly 3 live neighbours
- If the cell is dead, then it is “born” only if it has exactly 3 live neighbours

Check out: https://playgameoflife.com

Try this one!

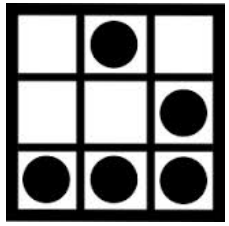


Check out: https://playgameoflife.com

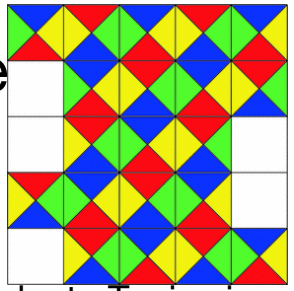
After completion of the course: You will be able to answer questions such as

- Are there any problems that—no matter how powerful the computer/no matter how much time you have—your computer would not not be able to solve?

- Is it decidable (ie., is there a computation that allows us to decide), in **Conway's Game of Life**, whether given an initial pattern and another pattern, the latter pattern can ever appear when starting from the initial one?



- Is it decidable, given a finite set of tile types, to determine whether there is an arrangement of them with adjacent sides matching that tiles the plane? (**The Tiling Problem**)



Tiling problem image from: Reus B. (2016) More Undecidable Problems. In: Limits of Computation. Undergraduate Topics in Computer Science. Springer.

You will be able to answer questions such as (cont'd):

- Is it possible to design an algorithm that is the perfect antivirus software, that is a software that decides for any current or future software whether or not the software can act like a virus?

- Is it possible to design a perfect debugger (an algorithm that that, among other bugs, catches infinite loops)?

- Is there an algorithm that schedules exams over a limited time period, such that no student is scheduled to take two or more exams at the same time?

If the answer is yes to one or more such questions, we ask whether there exists an algorithm that is *tractable*, that is it returns the solution in an *acceptable* amount of time.

Big (but unachievable) goals ….

- Universal debuggers
- Universal interpreters
- Universal malware detectors

Course Topics

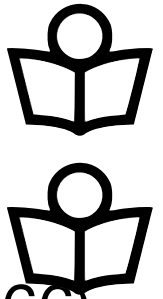
- Formal definitions of computations, languages and computability
- Models of computation
  - Finite automata, pushdown automata, Turing machines
  - Grammars
  - Deterministic and non-deterministic machines
- Undecidability, Decidability, Church-Turing Thesis
  - The Halting problem
  - Reductions
- Complexity theory
  - **P = NP?** NP-hardness, NP-completeness
  - Polynomial-time reductions
- If time permits: Dealing with NP-hardness; quantum computing

https://www.claymath.org/millennium-problems

Related Questions

- If at all: what type of problems can humans solve of the ones that are not solvable by (classical) computers?
- Can computers (eg: robots, conversational agents) act as humans?

Turing Machines, Turing Test, Conversational Agents



- Turing test (developed by A. Turing,1950)
- First conversational agent: ELIZA (J. Weizenbaum, 1966)
- Today: many conversational agents
  - IBM Watson Jeopardy
  - Siri
  - Cortana
  - Alexa
  - Google Assistant
  - Chatbots

More Admin

Where to find Information, Course Materials, Assessments

- Official Course Outline
- Brightspace
- Crowdmark

Deadlines; Assessment

10%

Quizzes

Quiz 1-8: 1% each
Quiz 9: 2%

25%

Assignments

Assignment 1-5: 5% each

25%

Midterms

Midterm 1: 10%
Midterm 2: 15%

40%

Final Exam

May

S	M	T	W	T	F	S
			3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	1	2	3
4	5	6	7	8	9	10

June

S	M	T	W	T	F	S
28	29	30	31	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	1
2	3	4	5	6	7	8

July

S	M	T	W	T	F	S
25	26	27	28	29	30	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	

Timed quizzes (~30 min)
Review before starting quiz

Assignments & Quizzes will take place on Crowdmark

Accommodation for potential difficulties keeping deadlines due to illness
Note: No other accommodation will be made

To accommodate for potential difficulties keeping deadlines such as illness, note below. Note that **no other accommodations** will be made.

Quizzes

There are 9 **timed** quizzes. Each quiz is available throughout a 24-hour window. Once you start the quiz, you will need to complete it in the given time window.

For quizzes 1 to 8, every student can miss at most one quiz. The weight of the missed quiz will be distributed evenly over the remaining quizzes. For other missed quizzes a grade of 0 will be assigned.

Assignments

There are 5 assignments.

A late penalty of 3% is imposed for every hour of handing in an assignment late. For assignments 1 to 4, every student can miss at most one assignment. The weight of the missed assignment will be distributed evenly over the remaining assignments. For other missed assignments, a grade of 0 will be assigned.

Midterms

There are 2 midterms. Everyone can miss at most one midterm. The weight of the missed midterm will be added onto the weight of the final exam. For another missed midterm, a grade of 0 will be assigned.

Textbook, Prerequisites

- Book: Introduction to the Theory of Computation
  by M. Sipser

Questions about the course, assignments, content, …?

- Consult in the following order
  1. Course outline/Brightspace/Lecture Notes
  2. Instructor in office hours or TA in tutorials
  3. If unresolved: email (ustege@uvic.ca)
     - **Always: CSC 320 in subject line**
     - Use your UVic preferred email

Questions about marking

- Grade change requests
  - Within one week of grades posted
  - In my office hours or email to me (ustege@uvic.ca)
  - Specific question/argument why your answer is correct/deserves your mark
    - Present your knowledge

Let’s get started …

Today's Theory of (Classical) Computation …

- is based on the **Church-Turing Thesis**

Fall 2018

Any real-world computation can be translated into an equivalent computation involving a Turing machine

- resource constraints (time, space) are ignored
- **Turing machine:** abstract model of (classical) computer

Today

Does Quantum Computing add computational power?

Assuming that any problem that can be computed can be translated into a Turing machine computation …

- … then there must be a Turing machine computation for every problem that we can compute
- What if there are **more problems** that we would like to compute **than** there are **Turing machine computations**?
- Then, there would be **problems** that are **not computable** by a Turing machine

We will learn that problems exist that are not computable/decidable by a Turing machine (eg: Halting problem)

To know how many different problems or Turing machine computations there are, we need to know how to count sets, and how to determine sets that are uncountable

MATH122 - Logic and Foundations

Description

Logic and quantifiers, basic set theory, mathematical induction and recursive definitions, divide and conquer recurrence relations, properties of integers, counting, functions and relations, countable and uncountable sets, asymptotic notation.

Countable and uncountable

- A set is **countable** if it is **finite** or **countably infinite**: the elements of a countable set can always be counted one at a time; *every element of the set is associated with a unique natural number*

- There exists a **bijection** between any **countably infinite** set and the **set of natural numbers** $\mathbb{N}$

- There exists a **bijection** between any **finite set** and a **finite subset of** $\mathbb{N}$

- A set that is neither finite nor countably infinite is **uncountable**

Let’s count some sets

- By definition $\mathbb{N}$ is countable
- Lets count the set of all integers $\mathbb{Z}$
  - How do we enumerate all elements in $\mathbb{Z}$?

$\mathbb{N}$	$\mathbb{Z}$
0	0
1	-1
2	1
3	-2
4	2
5	-3
6	3
…	…

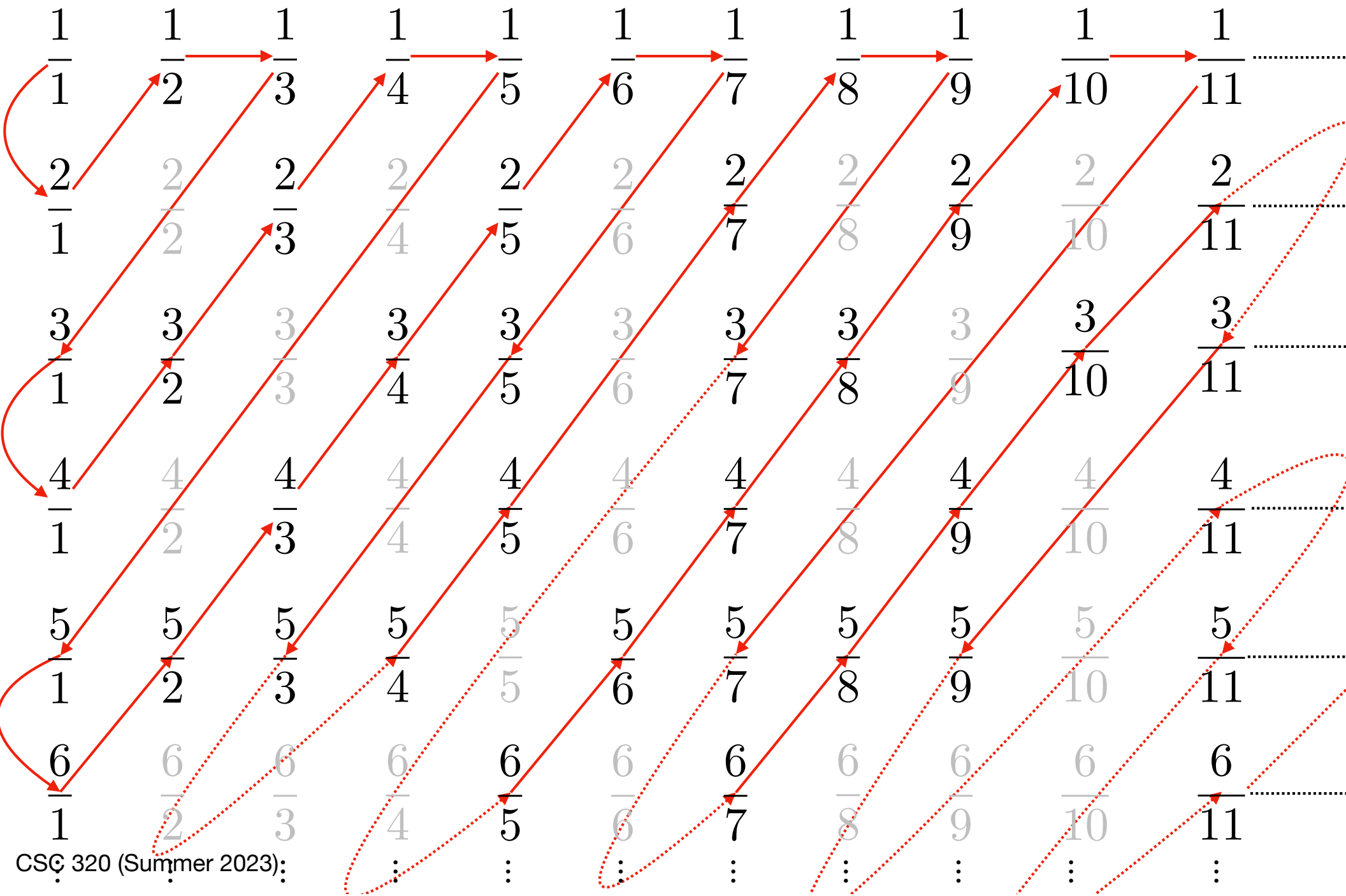
Counting the set of positive rational numbers $\mathbb{Q}^+ \setminus \{0\}$

- Let's first list them

Listing all positive rational numbers $\mathbb{Q}^+ \setminus \{0\}$

$\frac{1}{1}$	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{4}$	$\frac{1}{5}$	$\frac{1}{6}$	$\frac{1}{7}$	$\frac{1}{8}$	$\frac{1}{9}$	$\frac{1}{10}$	$\frac{1}{11}$ .....
$\frac{2}{1}$	$\frac{2}{2}$	$\frac{2}{3}$	$\frac{2}{4}$	$\frac{2}{5}$	$\frac{2}{6}$	$\frac{2}{7}$	$\frac{2}{8}$	$\frac{2}{9}$	$\frac{2}{10}$	$\frac{2}{11}$ .....
$\frac{3}{1}$	$\frac{3}{2}$	$\frac{3}{3}$	$\frac{3}{4}$	$\frac{3}{5}$	$\frac{3}{6}$	$\frac{3}{7}$	$\frac{3}{8}$	$\frac{3}{9}$	$\frac{3}{10}$	$\frac{3}{11}$ .....
$\frac{4}{1}$	$\frac{4}{2}$	$\frac{4}{3}$	$\frac{4}{4}$	$\frac{4}{5}$	$\frac{4}{6}$	$\frac{4}{7}$	$\frac{4}{8}$	$\frac{4}{9}$	$\frac{4}{10}$	$\frac{4}{11}$
$\frac{5}{1}$	$\frac{5}{2}$	$\frac{5}{3}$	$\frac{5}{4}$	$\frac{5}{5}$	$\frac{5}{6}$	$\frac{5}{7}$	$\frac{5}{8}$	$\frac{5}{9}$	$\frac{5}{10}$	$\frac{5}{11}$ .....
$\frac{6}{1}$	$\frac{6}{2}$	$\frac{6}{3}$	$\frac{6}{4}$	$\frac{6}{5}$	$\frac{6}{6}$	$\frac{6}{7}$	$\frac{6}{8}$	$\frac{6}{9}$	$\frac{6}{10}$	$\frac{6}{11}$ .....
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Counting the set of positive rational numbers $\mathbb{Q}^+ \setminus \{0\}$



Counting $\mathbb{Q}^+ \setminus \{0\}$

- We found a way to enumerate all rational numbers without getting stuck in infinity
  - Enumeration algorithm above defines bijection with $\mathbb{N}$
- **Note:** Counting row by row (or alternatively column by column) would never reach all numbers—since we would never leave the first row (column)

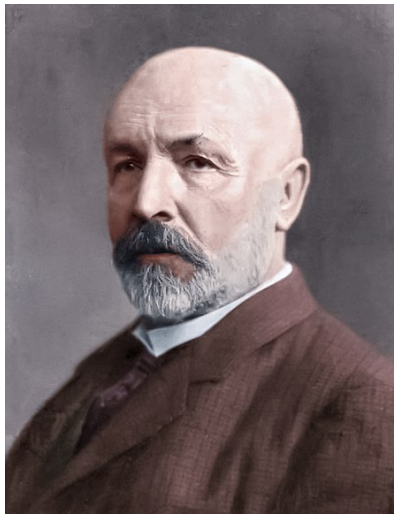
$\mathbb{R}$ numbers is uncountable
Cantor's Diagonalization Argument

Proof by contradiction. Assume: $\mathbb{R}$ is countable.

If $\mathbb{R}$ is countable then we can enumerate, eg, the set of all real numbers between 0 and 1:

$$
\begin{aligned}
x_1 &= 0.d_{11}d_{12}d_{13}d_{14}\ldots \\
x_2 &= 0.d_{21}d_{22}d_{23}d_{24}\ldots \\
x_3 &= 0.d_{31}d_{32}d_{33}d_{34}\ldots \\
x_4 &= 0.d_{41}d_{42}d_{43}d_{44}\ldots \\
&\vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots
\end{aligned}
$$

where $x_n = 0.d_{n1}d_{n2}d_{n3}d_{n4}\ldots$



George Cantor
1845—1918

Cantor's Diagonalization Argument (continued)

$$
\begin{array}{ll}
x_1 & 0.d_{11}d_{12}d_{13}d_{14}\ldots \\
x_2 & 0.d_{21}d_{22}d_{23}d_{24}\ldots \\
x_3 & 0.d_{31}d_{32}d_{33}d_{34}\ldots \\
x_4 & 0.d_{41}d_{42}d_{43}d_{44}\ldots \\
 & \vdots\quad\vdots\quad\vdots\quad\vdots\quad\vdots \\
x_n & = 0.d_{n1}d_{n2}d_{n3}d_{n4}\ldots \\
 & \vdots\quad\vdots\quad\vdots\quad\vdots\quad\vdots
\end{array}
$$

$c = x_1$? No: $c_1 \neq d_{11}$

Consider the number $c = 0.c_1c_2c_3c_4\ldots$ with $c_i \neq d_{ii}$ for each i

Clearly: $0 \leq c < 1$

c is not in above list

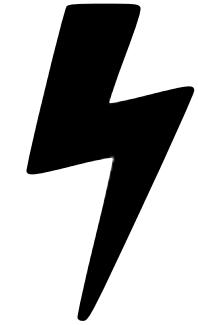
Cantor's Diagonalization Argument (continued)

$$
\begin{array}{lll}
x_1 & 0.d_{11}d_{12}d_{13}d_{14}\ldots & c \neq x_1 \\
x_2 & 0.d_{21}d_{22}d_{23}d_{24}\ldots & c \neq x_2 \\
x_3 & 0.d_{31}d_{32}d_{33}d_{34}\ldots & c \neq x_3 \\
x_4 & 0.d_{41}d_{42}d_{43}d_{44}\ldots & c \neq x_4 \\
 & \vdots\ \ \vdots\ \ \vdots\ \ \vdots\ \ \vdots & \\
x_n = & 0.d_{n1}d_{n2}d_{n3}d_{n4}\ldots & c \neq x_n \\
 & \vdots\ \ \vdots\ \ \vdots\ \ \vdots\ \ \vdots &
\end{array}
$$

$$
c = 0.c_1c_2c_3c_4\ldots \qquad c_i \neq d_{ii}
$$

Cantor's Diagonalization Argument (continued)

- Answer: No!
  $0.c_1c_2c_3c_4\ldots$ is not in the list as it is different from each list element! Therefore, since $0 \leq c \leq 1$, the list **does not** contain all real numbers between 0 and 1



- If we cannot even enumerate this subset of $\mathbb{R}$, then it is also impossible to enumerate $\mathbb{R}$

Recommended reading: https://www.cantorsparadise.com/hilberts-hotel-an-ingenious-explanation-of-infinity-1d1a79932080