INTRO TO QUANTUM COMPUTING
WEEK #1

CLASSICAL COMPUTING

AMIR KARAMLOU

10/18/2020

JAY GAMBETTA

IBM Fellow and VP, IBM Quantum

COURSE LOGISTICS

Reminders

CODE OF CONDUCT

The Coding School (TCS) is committed to fostering a respectful, inclusive environment where all participants feel safe and can do their best learning. We are committed to values of diversity and inclusion. A core part of our organization's mission is to make the future of STEM more representative and welcoming to all. Therefore, we take these responsibilities seriously and have high expectations for our learning community. All TCS members are required to read and agree to follow the Code of Conduct outlined below.

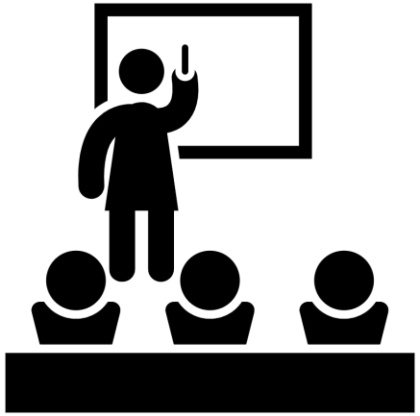
Examples of behaviors that contribute to a positive learning environment include:

- Using welcoming and inclusive language at all times.
- Being respectful of differing viewpoints, experiences, and cultures.
- Showing empathy toward others.
- Being an active listener and engaged participant.
- Valuing the contributions of all members of the TCS community.
- Respectfully pointing out unintentionally racist, sexist, or biased comments and jokes when they happen.

COURSE NORMS

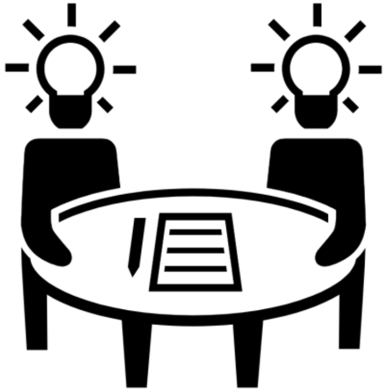
- Failure is a part of the process
- You will not understand everything - that is okay (we would be concerned if you did!)
- Curiosity, effort, and engagement are valued over perfection
- Every student is on a level playing field, and we can all learn from one another
- Students are not in competition with one another and should be supportive, not competitive
- We actively promote diversity in our courses and want all students to celebrate the different backgrounds and experiences of our students
- We ask students to be present, engaged, and supportive of one another

COURSE COMPONENTS



Lecture

- Sundays: 2pm-4pm EST
- Breakdown: 1 hour of theory and 1 hour tutorial
- ***Recordings will be available for those unable to attend***



Lab

- Options: Tuesday - Saturday
- 1 hour weekly with a TA: review homework, go over practice problems, answer questions
- Will be sent out this week
- Recordings will be available but for those attending a section, attendance will be recorded

Homework

- Weekly problem sets based on lecture
- Will be graded (Pass/Fail)
- Unlimited submissions before deadline!
- Released: after lecture
- Due: 11:59 pm EST following Sunday and answers will be released right after submission deadline

COURSE SYLLABUS

Semester 1 focuses on the foundational math, programming, and physics concepts required to learn quantum computing. For some, this may be a review. We aim to relate as much of the content to quantum computing as possible. Topics include:

- Classical Computing
- Quantum Computing in the Abstract
- Math: Introduction to Vectors and Complex Numbers, Probability
- Math for Quantum Mechanics
- Introduction to Python Programming

Semester 2 focuses on quantum mechanics, quantum information, and quantum algorithms. Topics include:

- Quantum Mechanics
- The Qubit and Bloch Sphere
- Gates, Measurements and Quantum Circuits
- Quantum Key Distribution
- Superdense Coding + Quantum Teleportation
- Deutsch-Josza Algorithm, Grover's Algorithm, VQE & QAOA
- Experimental Metrics and Implementation

TECHNOLOGIES

DISCORD

Platform for communication + self-study groups

- ALL COMMUNICATION FROM QxQ so check official channels
- **If you're not on Discord, join now**
- **Live verify:** 10am-8pm ET TODAY

piazza

Platform for all academic content-related questions

- WILL BE SEND OUT THIS WEEK

canvas

Platform for all course materials and homework submissions

- WILL NOT BE SEND OUT THIS WEEK

KEY TAKE-AWAYS

- **Discord is the main source of communication and is the official means that QxQ will communicate with students. Make sure you are checking the official channels**
  - We will be updating the access of university/workforce students

- **If you have not already confirmed your spot, do so NOW.**

- **Labs start this week.** Fill out the form with your time preferences. If you just got accepted, you'll receive the email today.
  - If you got the wrong form (i.e. you are a HS student and received the university/workforce form, then follow the instructions on the form.)

- **Piazza will be introduced this week.** Canvas will not.

- **Homework will be released in Discord after the lecture.**

- **Friends can still apply. We've opened a waitlist.**

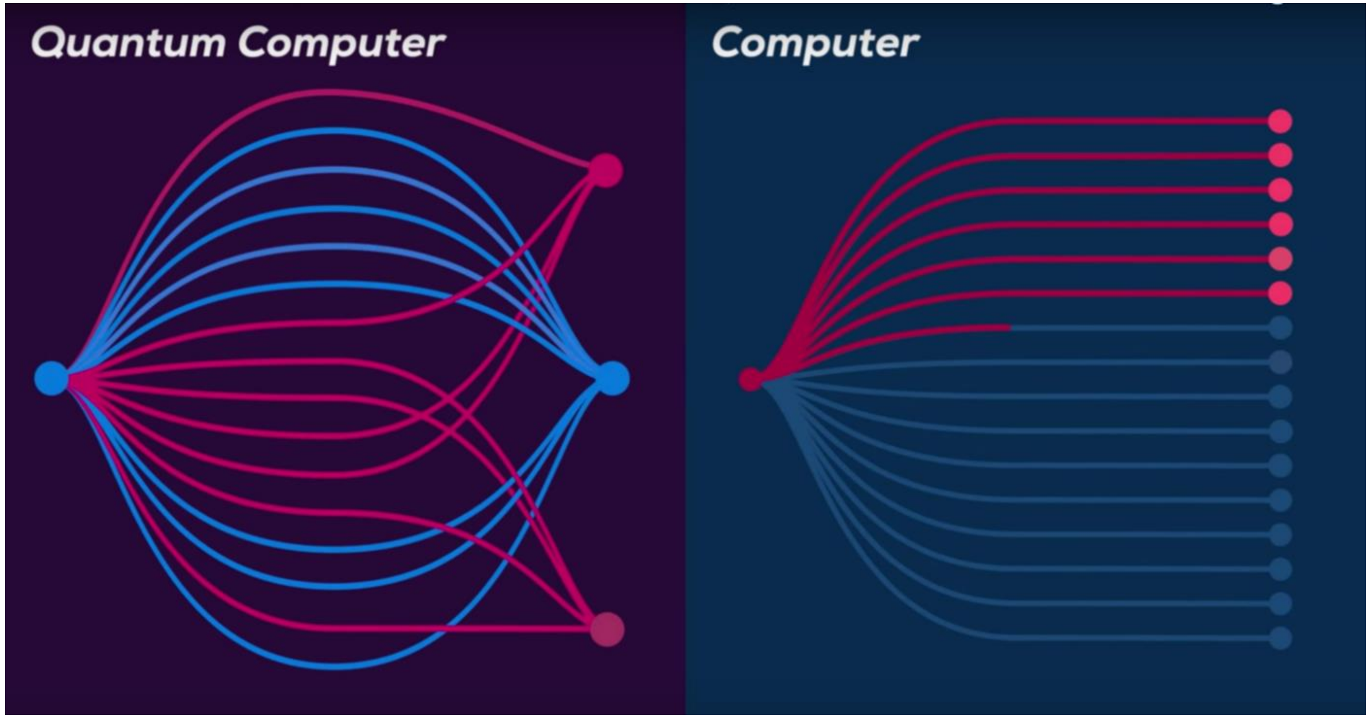
- **Do NOT message/LinkedIN/Discord your instructors or TAs.**

- **You are committed to completing this course through mid-May 2021.** If you're going to drop, this is your last chance.

TOPICS COVERED

Classical Computing

- History of classical computing
- Binary Representation
- Boolean Logic
- Algorithm Thinking



LECTURE OBJECTIVES

Why are we learning about *classical computing* when this course is about *quantum computing*?

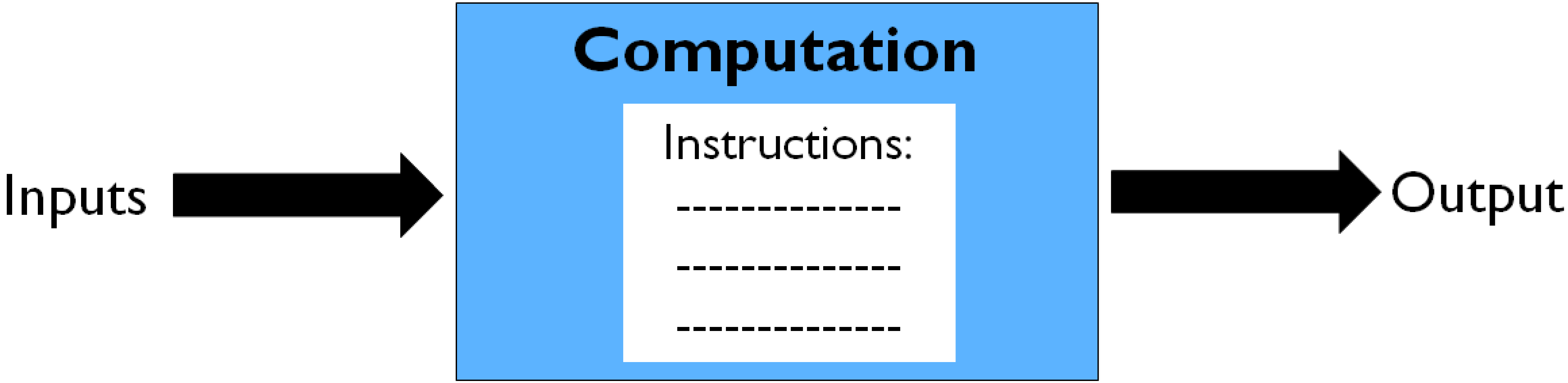
- **Bits**
- **Boolean Logic**
- **Universal Computation**
- **Reversible Computation**

History of Computing

WHAT IS COMPUTATION?

Computation:

A mathematical calculation that maps inputs to an output based on a set of instructions



A BIT-SIZED HISTORY OF COMPUTING

BASE-REPRESENTATIONS

learning to think like a computer

DECIMALS

- **Decimal number system is based on numerical digits 0-9**
  - **Also referred to as base-10**
- **Base** determines how numbers get represented and how we perform arithmetic operations

EXAMPLE

6 = 6

(6 x 10^0)

36 = 30 + 6

(3 x 10^1) + (6 x 10^0)

536 = 500 + 30 + 6

(5 x 10^2) + (3 x 10^1) + (6 x 10^0)

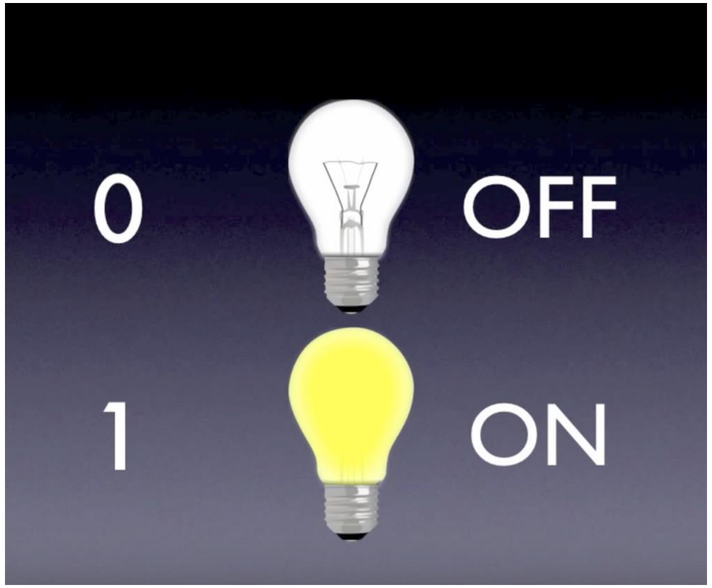
PRACTICE

1) 28

2) 432

BINARY

- **While it's common to use base-10 for counting, there is no mathematical reason to prefer one base over another.**
- **Base-2** is one of the most important bases for performing computation
- Base-2 is **binary**, meaning there are only two possible digits (0 and 1)
  - Also referred to as a **bit (binary digit)**



- **We can describe any number with a combination of bits**
- **All of the operations in a classical computer happen by manipulating bits**

CONVERTING BITS

Does 1010 in binary mean anything to you? *Probably not.*

- 1010 in decimal:

  $$(1 \times 10^3) + (0 \times 10^2) + (1 \times 10^1) + (0 \times 10^0)$$

- 1010 in binary:

  $$(1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (0 \times 2^0) = 8 + 2 = 10$$

PRACTICE

Convert the following bits into decimals:

1) 1100

2) 10011

PRACTICE

Convert the following decimal numbers into binary representation:

1) 33

2^0	1
2^1	2
2^2	4
2^3	8
2^4	16
2^5	32
2^6	64

BITS: ARITHMETIC OPERATORS

How computers compute

ADDING BITS

Binary addition:

- Similar to the decimal addition we're used to
- Bits carry over when the sum becomes larger than 2

- 0 + 0 =
- 0 + 1 =
- 1 + 0 =
- 1 + 1 =

PRACTICE

MULTIPLYING BITS

Binary multiplication:

- 0 x 0 =
- 0 x 1 =
- 1 x 0 =
- 1 x 1 =

PRACTICE

WHY BITS?

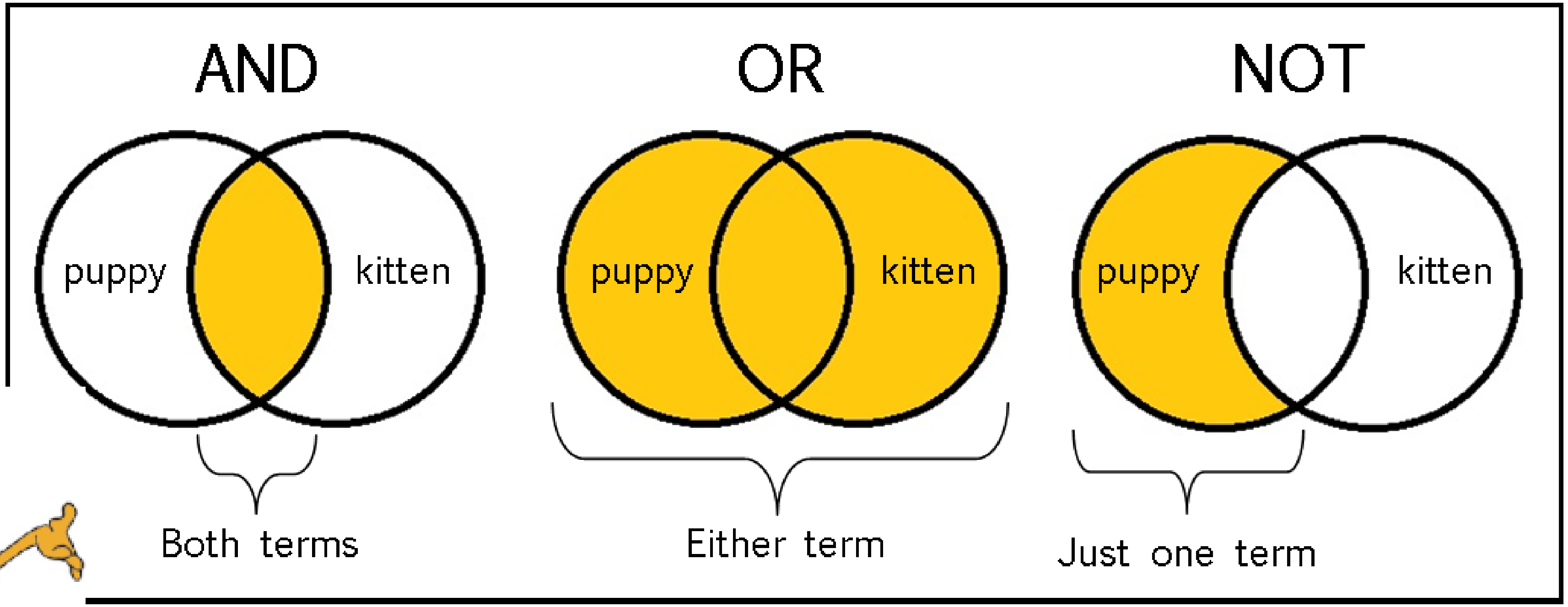
Easier for building hardware

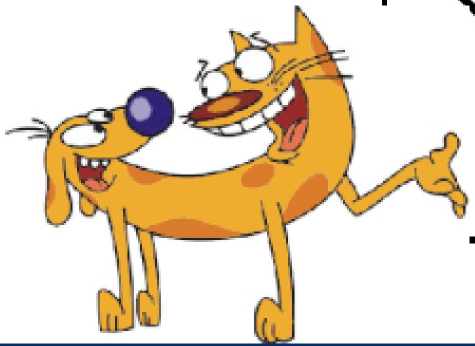
Fast operations

BOOLEAN LOGIC

WHAT IS BOOLEAN LOGIC?

- **Boolean logic:** Maps input bit(s) to output bit(s)

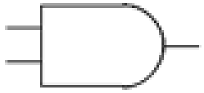
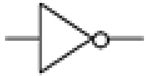
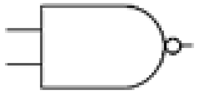
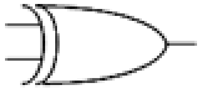




LOGIC GATES + TRUTH TABLES

LOGIC GATES

- **Logic gates:** Maps input bit(s) to output bit(s)

Gate	Symbol	Operator
and		$A \cdot B$
or		$A + B$
not		$\bar{A}$
nand		$\overline{A \cdot B}$
nor		$\overline{A + B}$
xor		$A \oplus B$

TRUTH [HURTS] TABLES

The truth table tells us the *output* of a logical operation based on its *inputs*

Truth Table

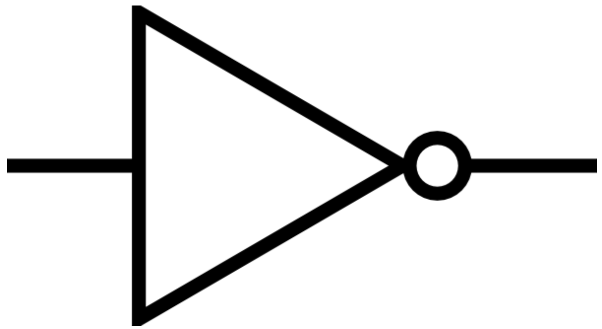
Input A	Input B	Output
0	0	0
1	0	1
0	1	1
1	1	1

GATES: 1 BIT

- **NOT**
- **FANOUT**

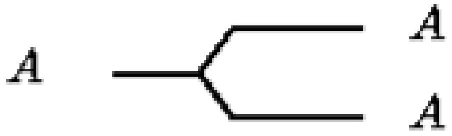
GATES: NOT

NOT: Flips the bit



GATES: FANOUT

FANOUT: **Copies the bit**



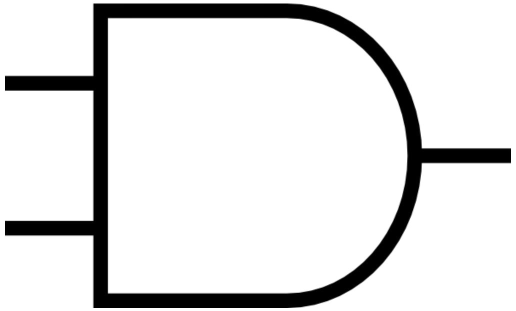
GATES: 2 BIT

- **AND**
- **OR**
- **XOR**

GATES: AND

AND: outputs 1 if both input bits are 1

outputs 0 otherwise

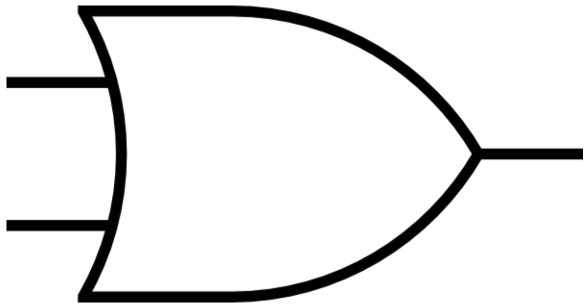


PRACTICE

GATES: OR

OR: outputs 1 if either of the input bits is 1

outputs 0 if neither of the input bits is 1

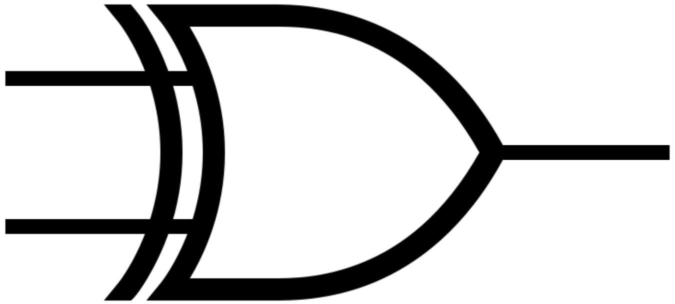


PRACTICE

GATES: XOR

XOR: outputs 1 if either input bits are 1, but not both

outputs 0 if neither or both bits are 1



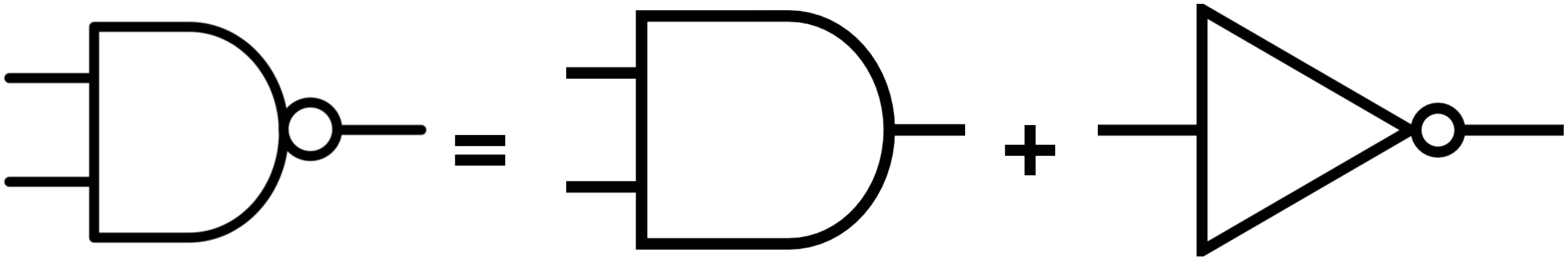
PRACTICE

UNIVERSALITY

Any computation operation can be made by using a combination of:

{NOT, AND, OR, FANOUT}

NAND: NOT AND



EXERCISE

Let's make a binary adder with the gates we have!

REVERSIBILITY

Given the output of a gate, we can determine what the inputs are.

EXAMPLE

Is NOT reversible?

Is AND reversible?

REVERSIBILITY

Given the output of a gate, we can determine what the inputs are.

- **Reversible gate:** preserves all the information
- **Non-reversible gate:** loses some information

THIS WEEK

Lab:

- **Labs start this week!**
- **In your lab section this week, you will cover: boolean algebra and logic circuits**

Homework:

- **Will be available on Discord in #course-announcements**
- **Topics will cover boolean algebra and logic circuits**
- **You will not submit your homework this week; you will submit it once we get everyone's onboarded to Canvas (week of Oct. 25)**

Important Notes:

- **Make sure you:**
  - **Sign up for your lab time**
  - **Confirm your acceptance (if you haven't already)**
  - **Sign up and get verified Discord**
  - **Sign up for Piazza (will be sent between today and Tuesday)**

FOLLOW US!

@thecodingschool

@the_coding_school

@the-coding-school

@qubitbyqubit

#QubitbyQubit

END OF LECTURE 1