Quantum Software Development

EE-193 / CS-150 | Spring 2024 | Tufts University

Lecture 1: Course Intro, Syllabus, Math Refresher, Quantum Information, Quantum Logic Gates

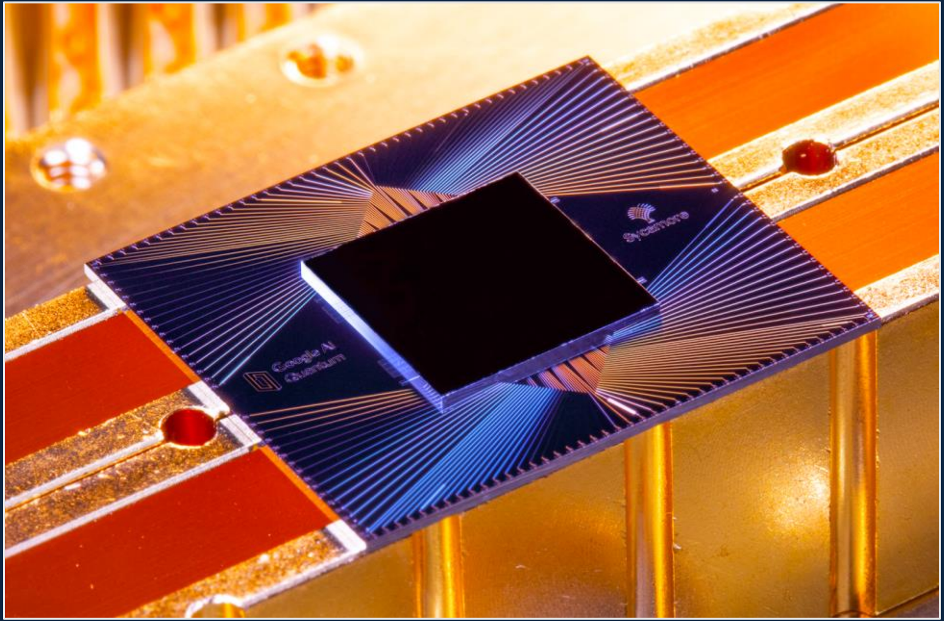
Tufts University School of Engineering

MITRE | SOLVING PROBLEMS FOR A SAFER WORLD®

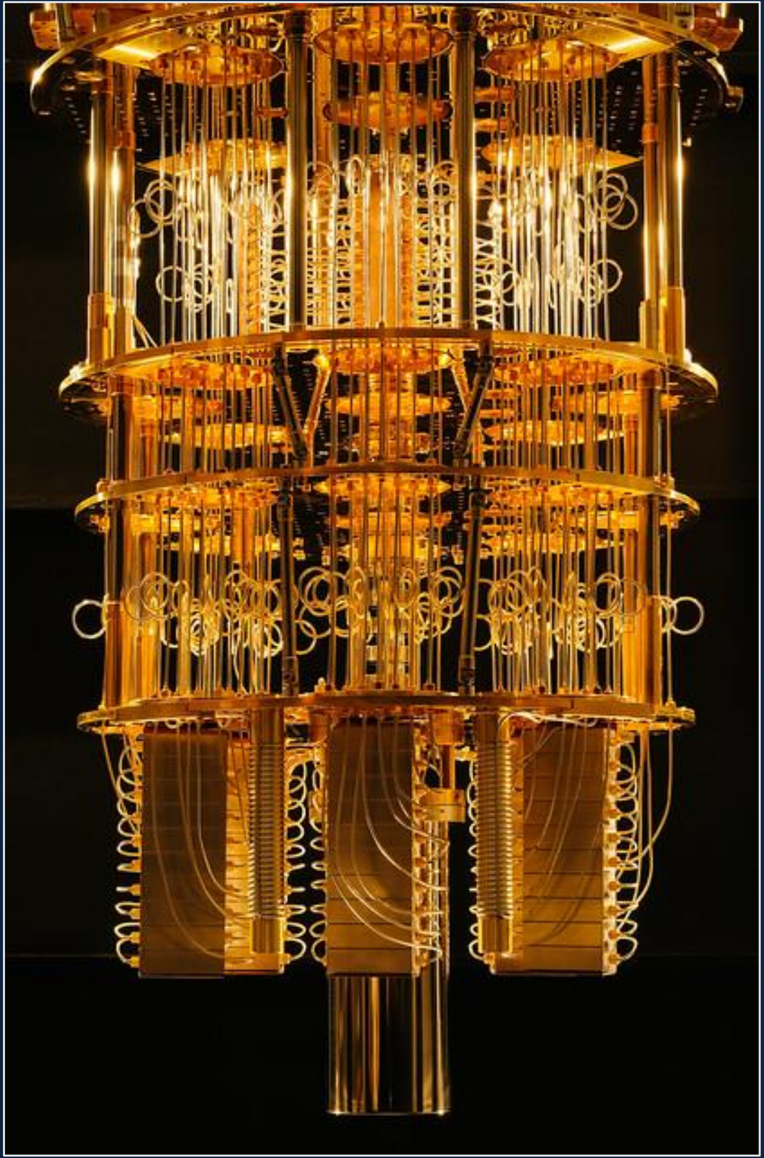
Course Intro

What is quantum computing?

Google Sycamore Processor
Photo by Erik Lucero of Google AI Quantum

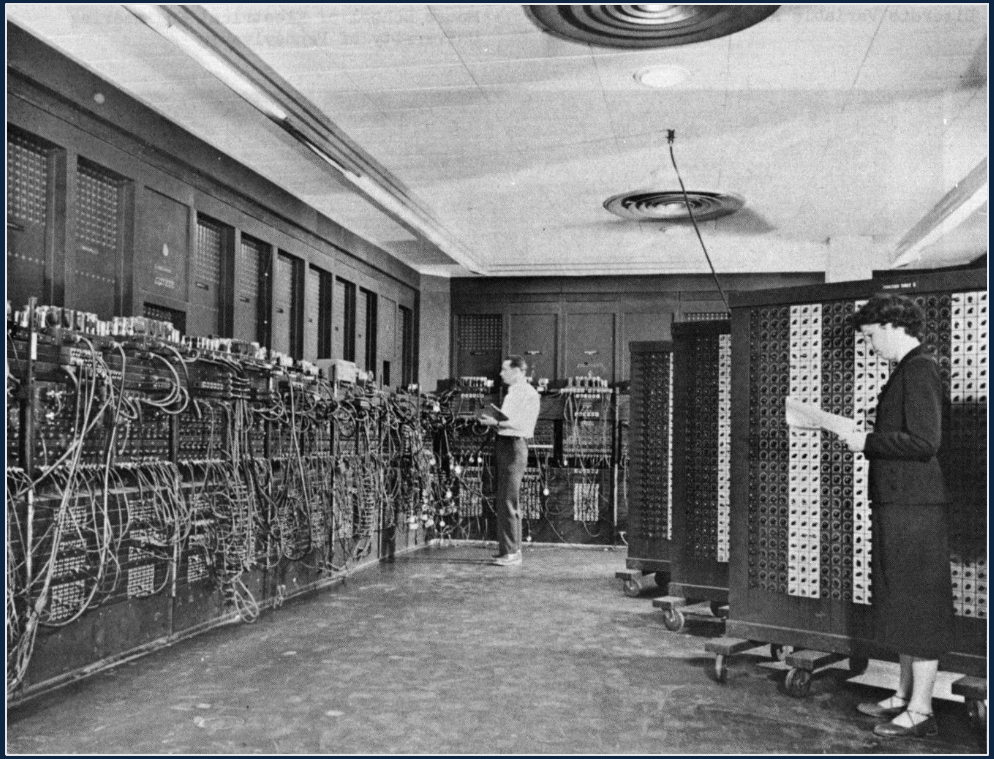


IBM Q Cryostat
Photo by IBM Research



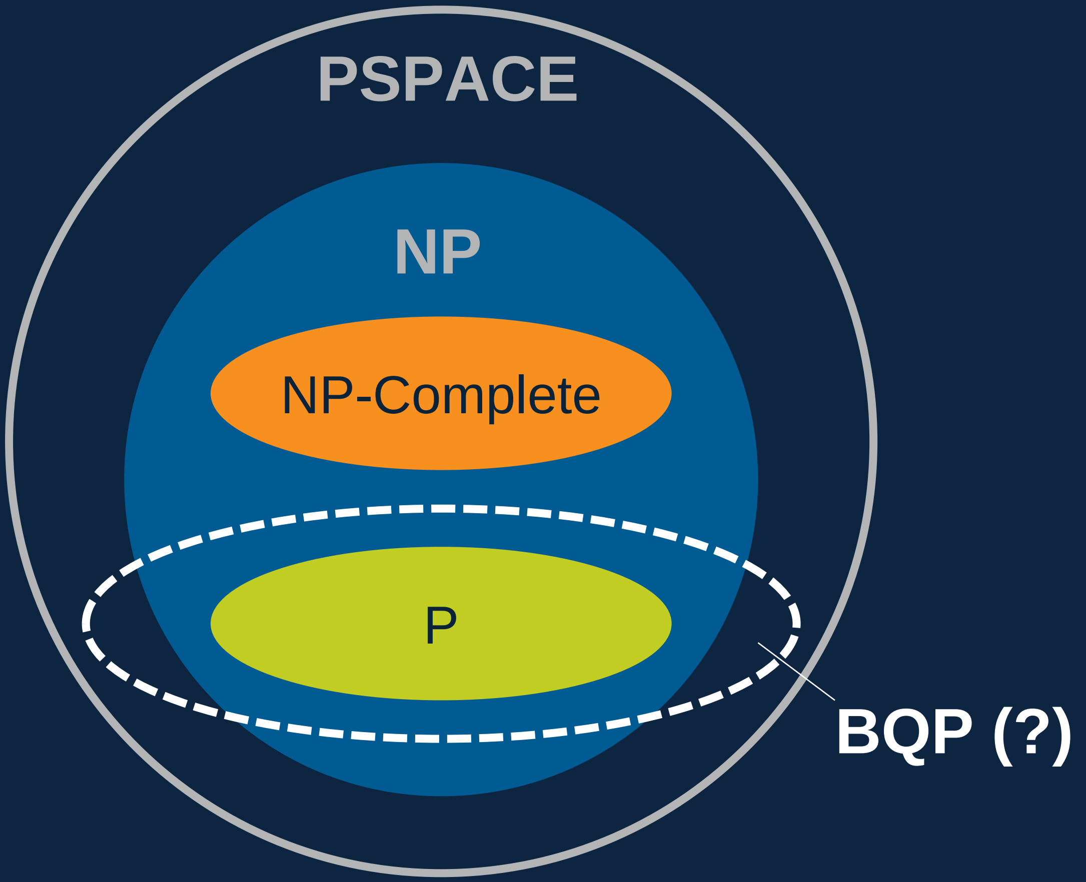
Virtually all conventional or "classical" computers process information using digital logic.

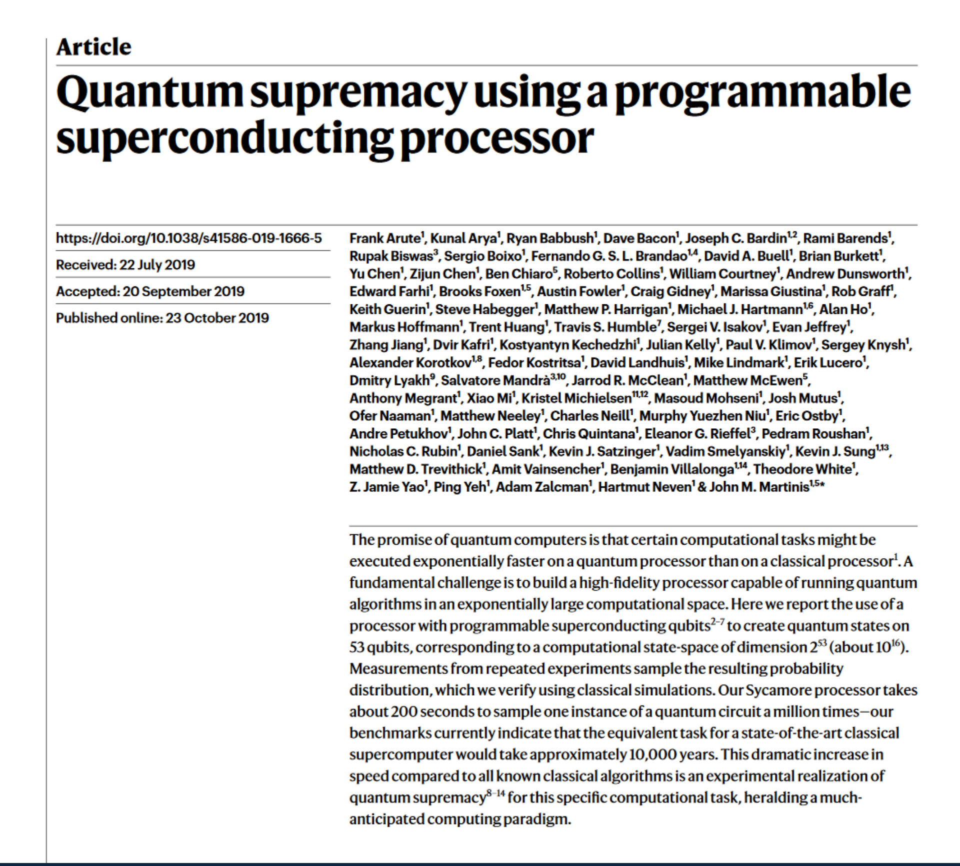
ENIAC
U.S. Army Photo



Summit Supercomputer
Photo by Carlos Jones of ORNL

Quantum computers process information using the principles of quantum mechanics.





Article

Quantum supremacy using a programmable superconducting processor

https://doi.org/10.1038/s41586-019-1666-5
Received: 22 July 2019
Accepted: 20 September 2019
Published online: 23 October 2019

Frank Arute[1], Kunal Arya[1], Ryan Babbush[1], Dave Bacon[1], Joseph C. Bardin[1,2], Rami Barends[1], Rupak Biswas[3], Sergio Boixo[1], Fernando G. S. L. Brandao[1,4], David A. Buell[1], Brian Burkett[1], Yu Chen[1], Zijun Chen[1], Ben Chiaro[5], Roberto Collins[1], William Courtney[1], Andrew Dunsworth[1], Edward Farhi[1], Brooks Foxen[1,5], Austin Fowler[1], Craig Gidney[1], Marissa Giustina[1], Rob Graff[1], Keith Guerin[1], Steve Habegger[1], Matthew P. Harrigan[1], Michael J. Hartmann[1,6], Alan Ho[1], Markus Hoffmann[1], Trent Huang[1], Travis S. Humble[7], Sergei V. Isakov[1], Evan Jeffrey[1], Zhang Jiang[1], Dvir Kafri[1], Kostyantyn Kechedzhi[1], Julian Kelly[1], Paul V. Klimov[1], Sergey Knysh[1], Alexander Korotkov[1,8], Fedor Kostritsa[1], David Landhuis[1], Mike Lindmark[1], Erik Lucero[1], Dmitry Lyakh[9], Salvatore Mandrà[3,10], Jarrod R. McClean[1], Matthew McEwen[5], Anthony Megrant[1], Xiao Mi[1], Kristel Michielsen[11,12], Masoud Mohseni[1], Josh Mutus[1], Ofer Naaman[1], Matthew Neeley[1], Charles Neill[1], Murphy Yuezhen Niu[1], Eric Ostby[1], Andre Petukhov[1], John C. Platt[1], Chris Quintana[1], Eleanor G. Rieffel[3], Pedram Roushan[1], Nicholas C. Rubin[1], Daniel Sank[1], Kevin J. Satzinger[1], Vadim Smelyanskiy[1], Kevin J. Sung[1,13], Matthew D. Trevithick[1], Amit Vainsencher[1], Benjamin Villalonga[1,14], Theodore White[1], Z. Jamie Yao[1], Ping Yeh[1], Adam Zalcman[1], Hartmut Neven[1] & John M. Martinis[1,5*]

The promise of quantum computers is that certain computational tasks might be executed exponentially faster on a quantum processor than on a classical processor[1]. A fundamental challenge is to build a high-fidelity processor capable of running quantum algorithms in an exponentially large computational space. Here we report the use of a processor with programmable superconducting qubits[2–7] to create quantum states on 53 qubits, corresponding to a computational state-space of dimension 2^{53} (about 10^{16}). Measurements from repeated experiments sample the resulting probability distribution, which we verify using classical simulations. Our Sycamore processor takes about 200 seconds to sample one instance of a quantum circuit a million times—our benchmarks currently indicate that the equivalent task for a state-of-the-art classical supercomputer would take approximately 10,000 years. This dramatic increase in speed compared to all known classical algorithms is an experimental realization of quantum supremacy[8–14] for this specific computational task, heralding a much-anticipated computing paradigm.

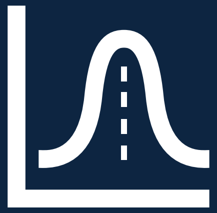
MITRE

Quantum Computing…

IS NOT simply a fast conventional computer

IS a new way of performing computation that solves specific problems better

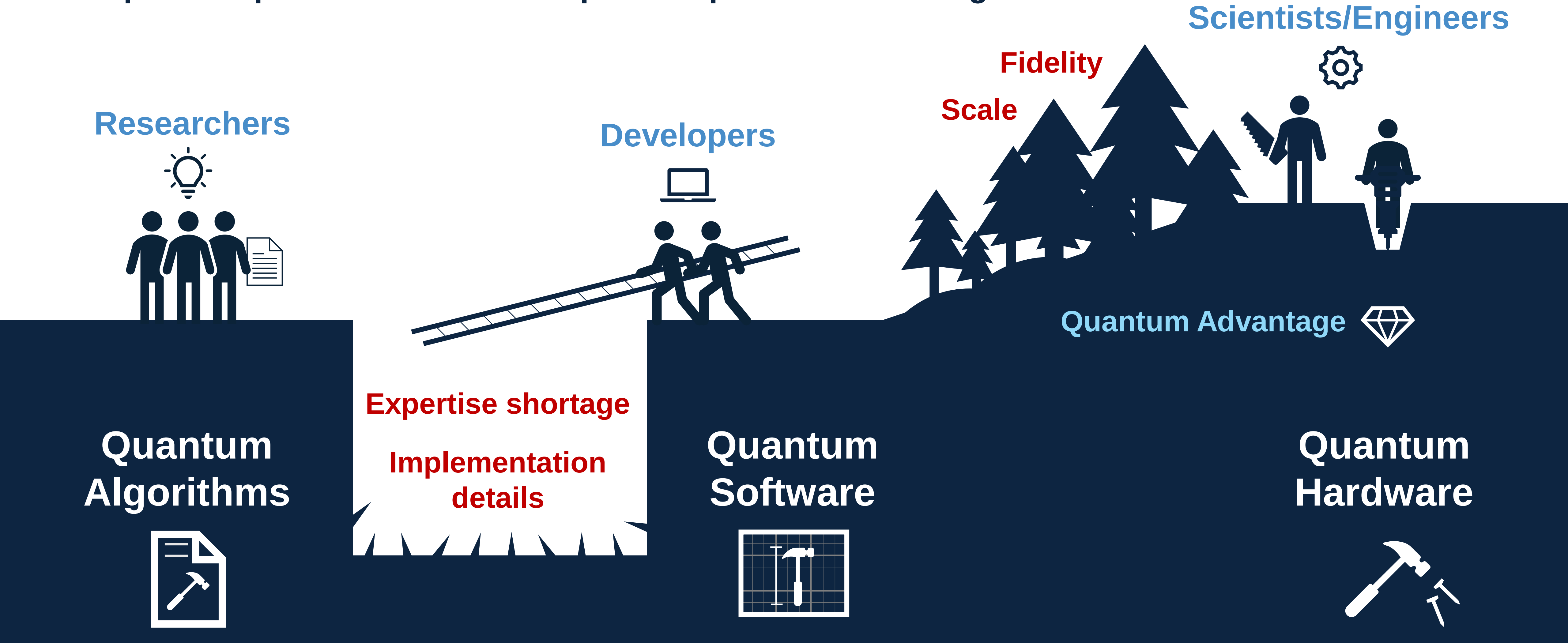
$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$



MITRE

The Simplified Quantum Computing Landscape

Multiple disciplines collaborate to pursue quantum advantage.



Syllabus

Key Information

Attend class!

Submit assignments through Canvas.

Use Discord for comms.

Instructor: Richard Preston

Instructor Email: rhpreston@mitre.org

Class Time: Wed, 4:30PM - 7:00PM, JCC Rm 265

Office Hours: Wed, 3:00PM - 4:30PM & 7:00PM - 8:30PM, 574 Boston Ave Rm 212

Exam 1: Wed Feb 21, 4:30PM, JCC Rm 265

Exam 2: Wed May 8, 12:00PM, location TBD

Canvas: canvas.tufts.edu/courses/54025

Discord Server: discord.gg/PCjUJFPksX

Graduate Student Handbook: tufts.app.box.com/v/soe-grad-handbook

MITRE

Course Objectives

1. **Understand quantum computation. Students should be able to correctly and clearly explain how quantum computers work at the level of information processing.**

2. **Attain competency in quantum software engineering. Students should be able to implement and study quantum algorithms using software.**

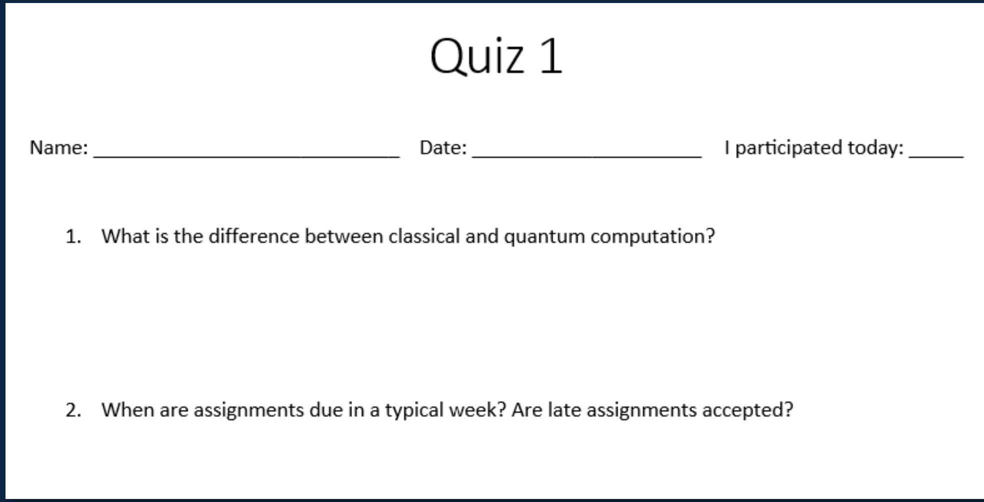
Grades

Participation	15%
Assignments	25%
Exams	30%
Final Project	30%

A+	0.98 – 1.00
A	0.94 – 0.97
A-	0.91 – 0.93
B+	0.88 – 0.90
B	0.84 – 0.87
B-	0.81 – 0.83
C	0.71 – 0.80
D	0.61 – 0.70
F	< 0.60

Participation

- **Show up!**
- **There will be a quiz at the beginning of each class after today.**
- **You get 1 point for doing the quiz and 1 point for participating.**
- **At the end of the semester, your score is the number of points out of 20.**
- **You can’t get a score higher than 1.0.**



Quiz 1

Name: ______________________ Date: ________________ I participated today: ____

1. What is the difference between classical and quantum computation?

2. When are assignments due in a typical week? Are late assignments accepted?

MITRE

Assignments

- There will be a coding assignment following each lecture starting next week.
- Assignments are due by the next class.
- Try to complete before 4:30PM but you can submit until 11:59PM with no penalty.
- You get half credit for exercises you attempted but failed to complete.
- You get half credit for submitting up to 1 week late. After that, no credit.

```
namespace Lab1 {

    open Microsoft.Quantum.Canon;
    open Microsoft.Quantum.Intrinsic;


    /// # Summary
    /// In this exercise, you are given a single qubit which is in the |0>
    /// state. Your objective is to flip the qubit. Use the single-qubit
    /// quantum gates that Q# provides to transform it into the |1> state.
    ///
    /// # Input
    /// ## target
    /// The qubit you need to flip. It will be in the |0> state initially.
    ///
    /// # Remarks
    /// This investigates how to apply quantum gates to qubits in Q#.
    operation Exercise1 (target: Qubit) : Unit {
        // TODO
        fail "Not implemented.";
    }

    /// # Summary
    /// In this exercise, you are given two qubits. Both of them are in the |0>
    /// state. Using the single-qubit gates, turn them into the |+> state and
    /// |-> state respectively. Recall the |+> state is 1/√2(|0> + |1>) and the
    /// |-> state is 1/√2(|0> - |1>).
    ///
    /// # Input
    /// ## targetA
    /// Turn this qubit from |0> to |+>.
    ///
    /// ## targetB
    /// Turn this qubit from |0> to |->.
    ///
    /// # Remarks
    /// This investigates how to prepare the |+> and |-> states.
    operation Exercise2 (targetA : Qubit, targetB : Qubit) : Unit {
        // TODO
        fail "Not implemented.";
    }
}
```

Exams

- There is a midterm and a final.
- Exam questions are designed to have unambiguous solutions.
- You will have a reference sheet and plenty of time.
- You get half credit for wrong answers.

Midterm Exam

Name: ______________________________ Date: ___________________________

Instructions: For multiple choice questions, circle the best answer. For free response questions, circle your simplified answer. All questions are intended to have unambiguous solutions. If you think a question may be ambiguous, please ask for clarification. A single notes sheet is allowed for reference. Use of electronic devices is not permitted. Use the back of the page if you need additional workspace.

Scoring: One point is awarded for each question attempted, and one additional point for each correct answer. There are 15 questions, so there are 30 possible points total.

Final Project

- Implement a quantum algorithm.
- The deliverable is a short video.
- Due May 3rd.

Assignment

The final project assignment is to study a quantum algorithm through software implementation and present your results in a video. How you approach the assignment is up to you, but a recommended breakdown is given below.

1. Select a quantum algorithm to study. Several options are provided in the last section.
   It's a good idea to check with the instructor if you are going to stray from the list.
2. Think through the high-level design of the program on paper.
   A good way to tell if you understand how your design works is by explaining it to someone else.
3. Write a Q# program that verifiably implements the algorithm.
   Write and run unit tests with the simulator to show your implementation works as expected.
4. Use the built-in resource estimator to obtain precise metrics for the computational resources required by your program, and/or run your program on a real quantum computer and capture the results.
5. Create a <10-minute video presenting your work to a technical audience. The grading rubric is provided in the next section.
   It is to your benefit if the video is accessible enough to be shared with others.

Calendar

Date	Assignment / Exam	Lecture Topic(s)
1/17		Course Intro, Syllabus, Math Refresher
1/24		Q. Information, Q. Logic Gates, Intro to Q#
1/31	**Lab 1 Due**	Visualizing Single-Qubit States, Working with Multiple Qubits
2/7	**Lab 2 Due**	Quantum Control Logic
2/14	**Lab 3 Due**	Quantum Communication, Quantum Error Correction
2/21	**Lab 4 Due**	Quantum Interference, Midterm Review
2/28	**Midterm Exam**	
3/6		Basic Quantum Algorithms
3/13	**Lab 5 Due**	Hybrid Algorithms, The Multiverse
3/20		NO CLASS
3/27	**Lab 6 Due**	Grover’s Search Algorithm, Final Project Discussion
4/3	**Lab 7 Due**	Quantum Fourier Transform
4/10	**Lab 8 Due**	Shor’s Factorization Algorithm
4/17		MAKE-UP DAY
4/24	**Lab 9 Due**	Cloud Execution, Final Review
TBD	**Final Exam**	
5/3	**Final Project Due**	

Math Refresher

A complex number is a quantity that has a real part and an imaginary part.

Definition

$$i^2 \equiv -1$$

Addition

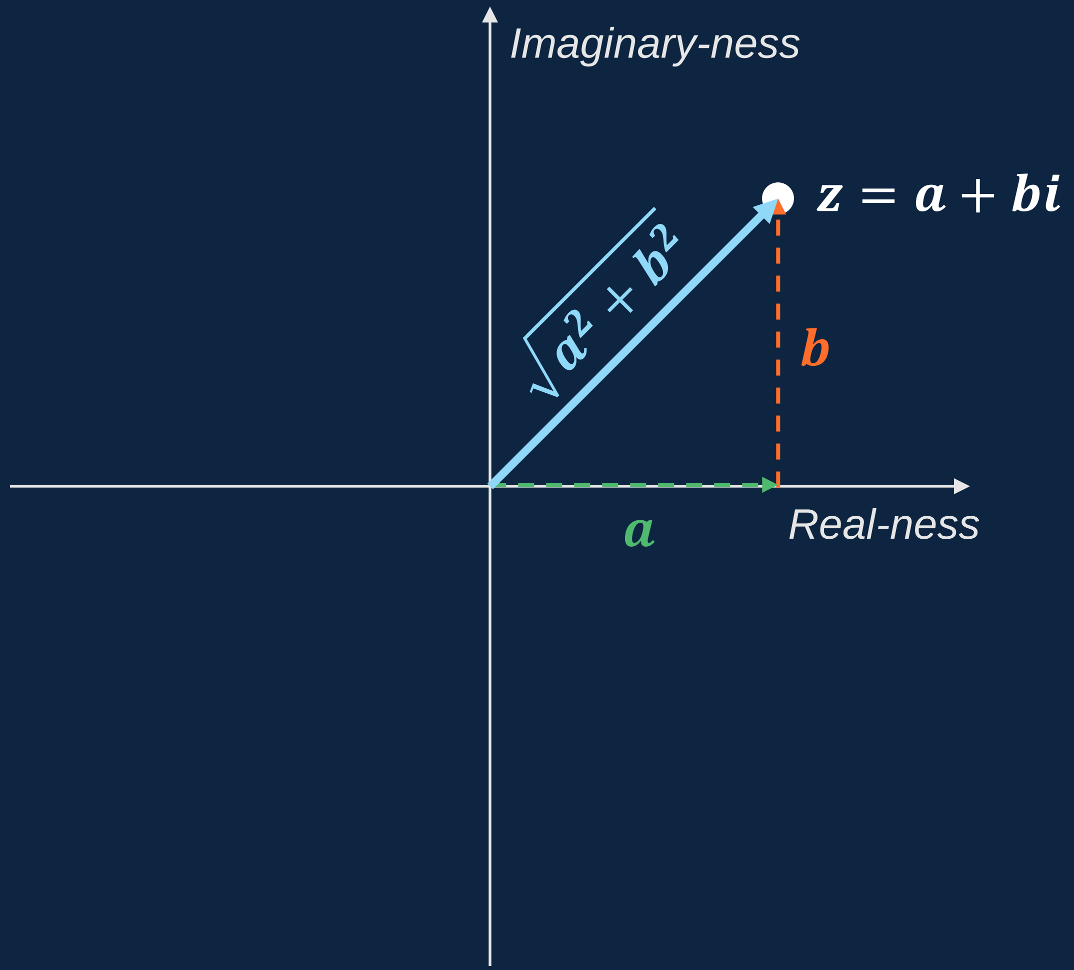
$$(a+bi)+(c+di)=(a+c)+(b+d)i$$

Multiplication

$$(a+bi)(c+di)=(ac-bd)+(ad+bc)i$$

Absolute Value / Magnitude

$$|a+bi|=\sqrt{a^2+b^2}$$



Euler's formula allows a complex number to be expressed in terms of its magnitude and phase.

Euler's formula

$$e^{i\varphi} = \cos\varphi + i\sin\varphi$$

Polar Form

$$re^{i\varphi}$$

Magnitude

$$\left|re^{i\varphi}\right| = r$$

Phase / Argument

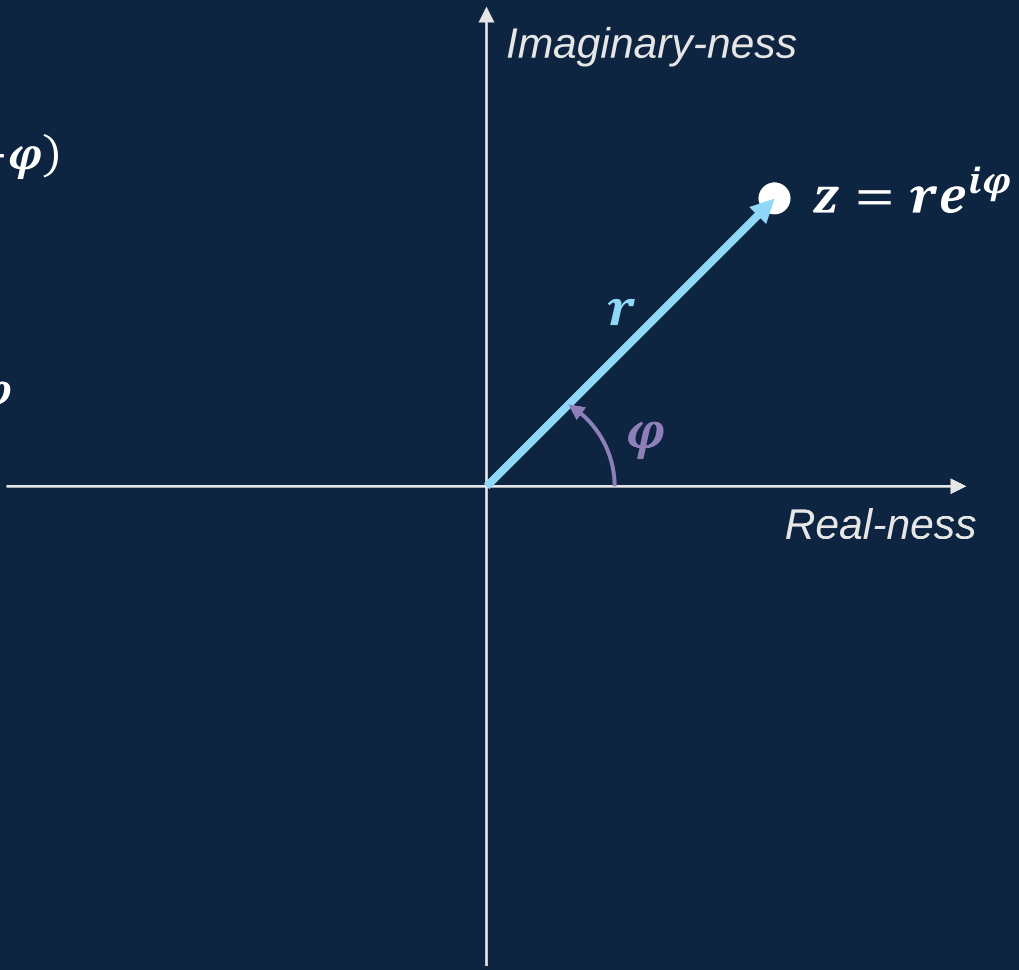
$$\arg\left(re^{i\varphi}\right) = \varphi$$

Phase Rotation

$$e^{i\theta} \cdot re^{i\varphi} = re^{i(\theta+\varphi)}$$

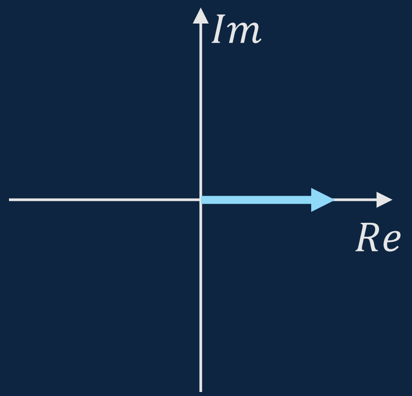
Periodicity

$$re^{i(\varphi+2\pi)} = re^{i\varphi}$$

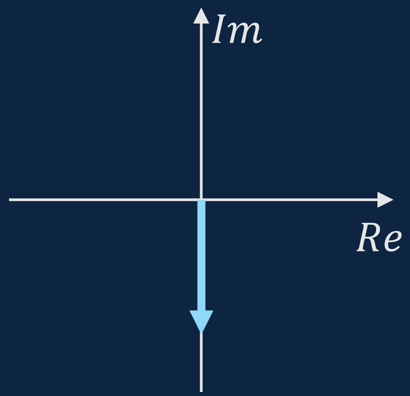


How are each of these quantities visualized on the complex plane?

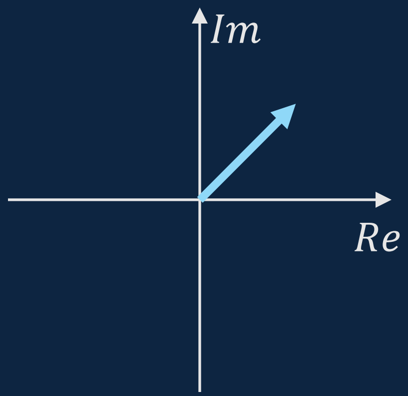
$$\mathbf{1}$$



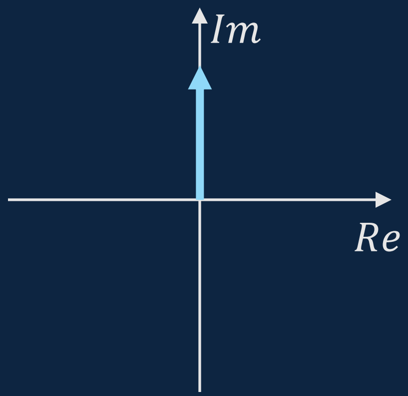
$$-i$$



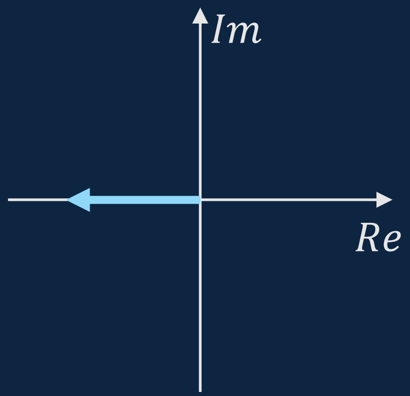
$$\frac{1}{\sqrt{2}}+\frac{i}{\sqrt{2}}$$



$$e^{\frac{\pi i}{2}}$$



$$e^{101\pi i}$$



MITRE

A vector is a quantity with magnitude and direction; a “coordinate” in n-dimensional space.

Magnitude

$$\left\| \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \right\| = \sqrt{v_1^2 + v_2^2 + \cdots + v_n^2}$$

Scalar Multiplication

$$c \cdot \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} c \cdot v_1 \\ c \cdot v_2 \\ \vdots \\ c \cdot v_n \end{bmatrix}$$

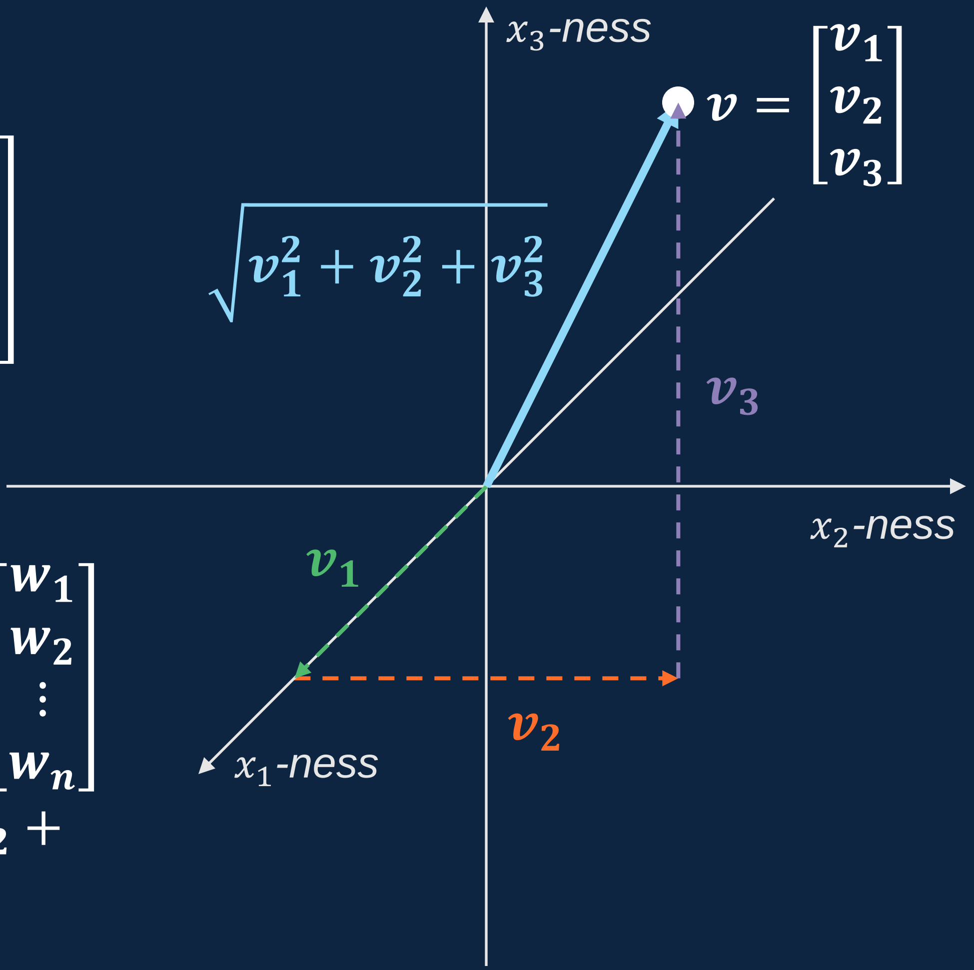
Addition

$$\begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} + \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} v_1 + w_1 \\ v_2 + w_2 \\ \vdots \\ v_n + w_n \end{bmatrix}$$

Vector Multiplication

$$\begin{bmatrix} v_1 & v_2 & \cdots & v_n \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$$

$$= v_1 \cdot w_1 + v_2 \cdot w_2 + \ldots + v_n \cdot w_n$$



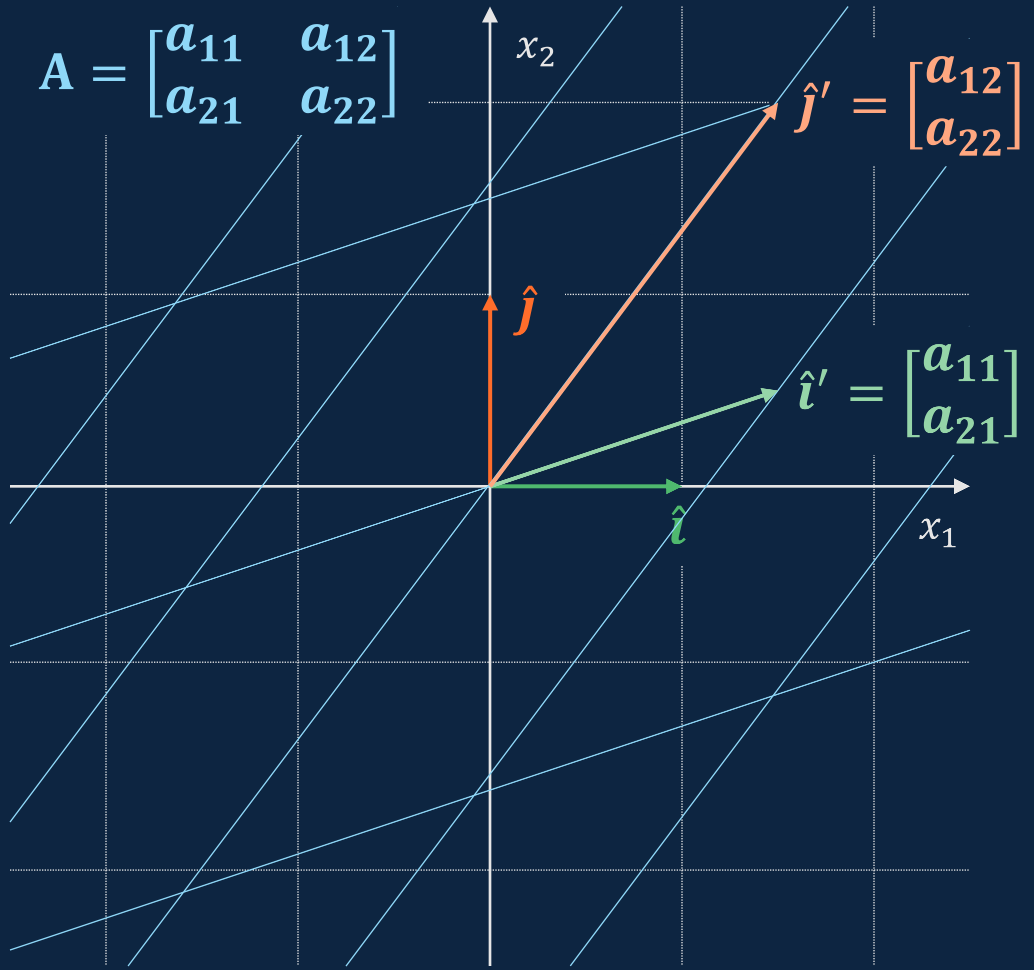
MITRE

A matrix defines a linear transformation (a.k.a. linear map) between vector spaces.

Matrix-Vector Multiplication

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

$$= \begin{bmatrix} a_{11} \cdot v_1 + \cdots + a_{1n} \cdot v_n \\ \vdots \\ a_{m1} \cdot v_1 + \cdots + a_{mn} \cdot v_n \end{bmatrix}$$



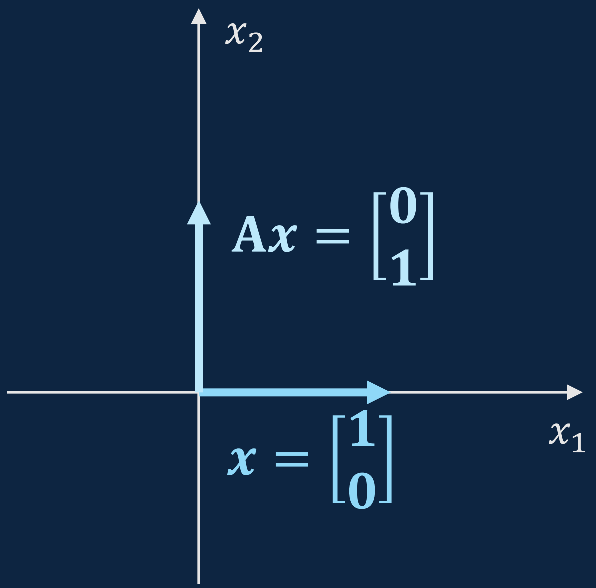
MITRE

What is the result of applying matrix A to vector x?

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad \boldsymbol{x} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$\mathbf{A}\boldsymbol{x} =$$

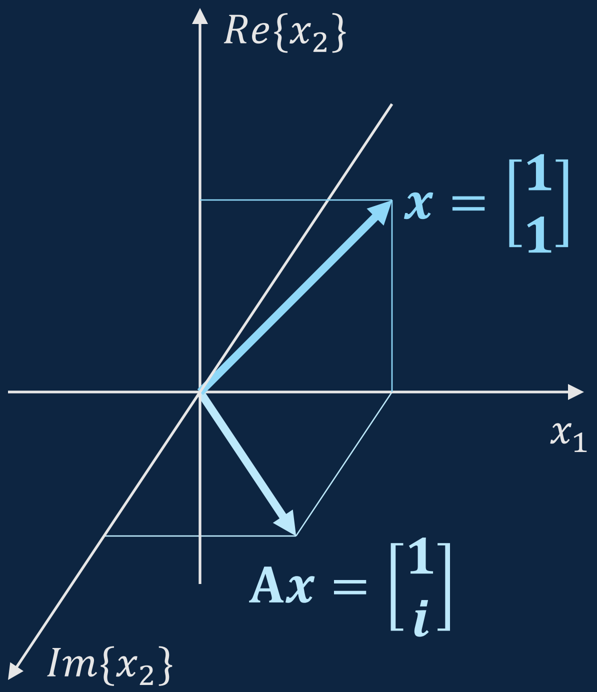
$$\begin{aligned} &\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 \cdot 1 + 1 \cdot 0 \\ 1 \cdot 1 + 0 \cdot 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 \\ 1 \end{bmatrix} \end{aligned}$$



$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \quad \boldsymbol{x} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\mathbf{A}\boldsymbol{x} =$$

$$\begin{aligned} &\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} 1 \cdot 1 + 0 \cdot 1 \\ 0 \cdot 1 + 1 \cdot i \end{bmatrix} \\ &= \begin{bmatrix} 1 \\ i \end{bmatrix} \end{aligned}$$



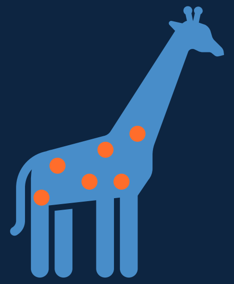
The tensor product multiplies the dimensions of a vector space.

Given vectors $v \in \mathbb{C}^m$ and $w \in \mathbb{C}^n$,
$v \otimes w$ produces a vector in $\mathbb{C}^{m \cdot n}$

$$\begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{bmatrix} \otimes \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} v_1 \cdot \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix} \\ v_2 \cdot \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix} \\ \vdots \\ v_m \cdot \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix} \end{bmatrix} = \begin{bmatrix} v_1 \cdot w_1 \\ v_1 \cdot w_2 \\ \vdots \\ v_1 \cdot w_n \\ v_2 \cdot w_1 \\ \vdots \\ v_m \cdot w_n \end{bmatrix}$$

$$\begin{bmatrix} \text{tallness} \\ \textit{wideness} \end{bmatrix} \otimes \begin{bmatrix} \textbf{redness} \\ \textbf{blueness} \end{bmatrix}$$

$$= \begin{bmatrix} \text{tall} - \text{and} - \text{redness} \\ \text{tall} - \text{and} - \text{blueness} \\ \textit{wide} - \textit{and} - \textit{redness} \\ \textit{wide} - \textit{and} - \textit{blueness} \end{bmatrix}$$



$$= \begin{bmatrix} 1 \\ 5 \\ 2 \\ 2 \end{bmatrix}$$

MITRE

Bra-ket (Dirac) notation is a concise way of expressing vectors and vector operations.

A bra $\langle x|$ denotes a row vector.

A ket $|y\rangle$ denotes a column vector.

A bra and ket together $\langle x|y\rangle$ denotes the dot product $\langle x| \cdot |y\rangle$.

Two adjacent kets $|y\rangle|z\rangle$ denotes the tensor product $|y\rangle \otimes |z\rangle$.
This can also be written as $|y, z\rangle$ or simply $|yz\rangle$.

Dirac notation is crucial for expressing quantum states.

What is the result of “tensoring” three kets together?

$$Let\ |x\rangle = \begin{bmatrix}1\\0\end{bmatrix}, |y\rangle = \begin{bmatrix}2\\3\end{bmatrix}, |z\rangle = \begin{bmatrix}0\\4\end{bmatrix}$$

$$|xyz\rangle = \begin{bmatrix}1\\0\end{bmatrix} \otimes \begin{bmatrix}2\\3\end{bmatrix} \otimes \begin{bmatrix}0\\4\end{bmatrix} = \begin{bmatrix}1\cdot 2\\1\cdot 3\\0\cdot 2\\0\cdot 3\end{bmatrix} \otimes \begin{bmatrix}0\\3\end{bmatrix} = \begin{bmatrix}2\\3\\0\\0\end{bmatrix} \otimes \begin{bmatrix}0\\4\end{bmatrix} = \begin{bmatrix}2\cdot 0\\2\cdot 4\\3\cdot 0\\3\cdot 4\\0\cdot 0\\0\cdot 4\\0\cdot 0\\0\cdot 4\end{bmatrix} = \begin{bmatrix}0\\8\\0\\12\\0\\0\\0\\0\end{bmatrix}$$

Quantum Information

MITRE

In classical computing, the fundamental unit of information is a binary state called a bit.

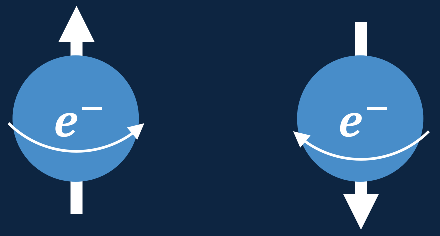
OFF ON

```
x = True
if x:
    print('x is True!')
```

Bits take on a discrete value of zero or one, off or on, false or true

In software, we refer to a single bit as a Boolean value

In quantum computing, the fundamental unit of information is a quantum state called a qubit.



$|0\rangle$ $|1\rangle$

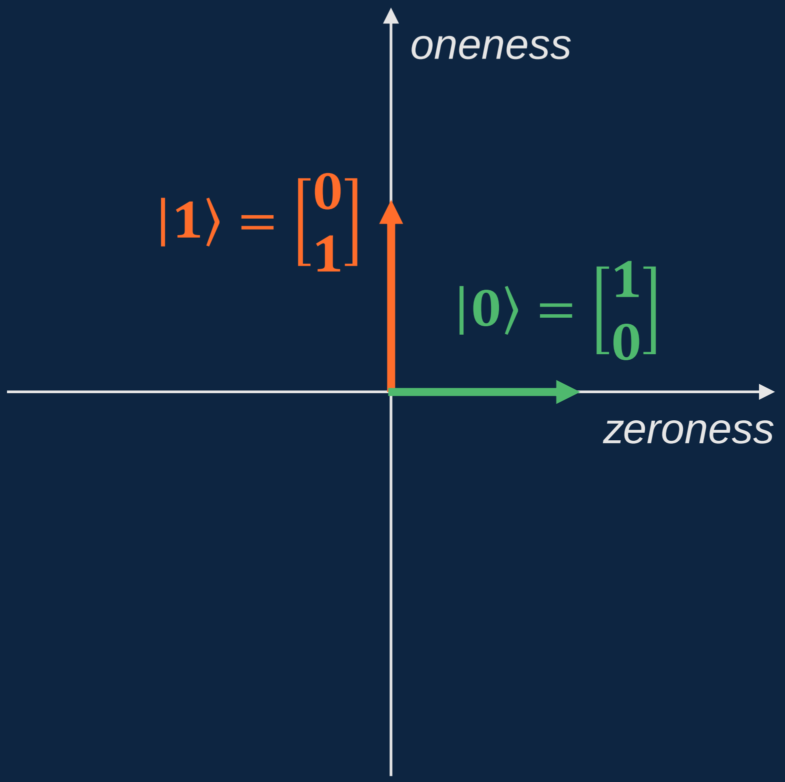
```
x = {
    'zeroness': 0.6,
    'oneness': 0.8
}
```

Qubits can be zero, one, or some combination of both (superposition)

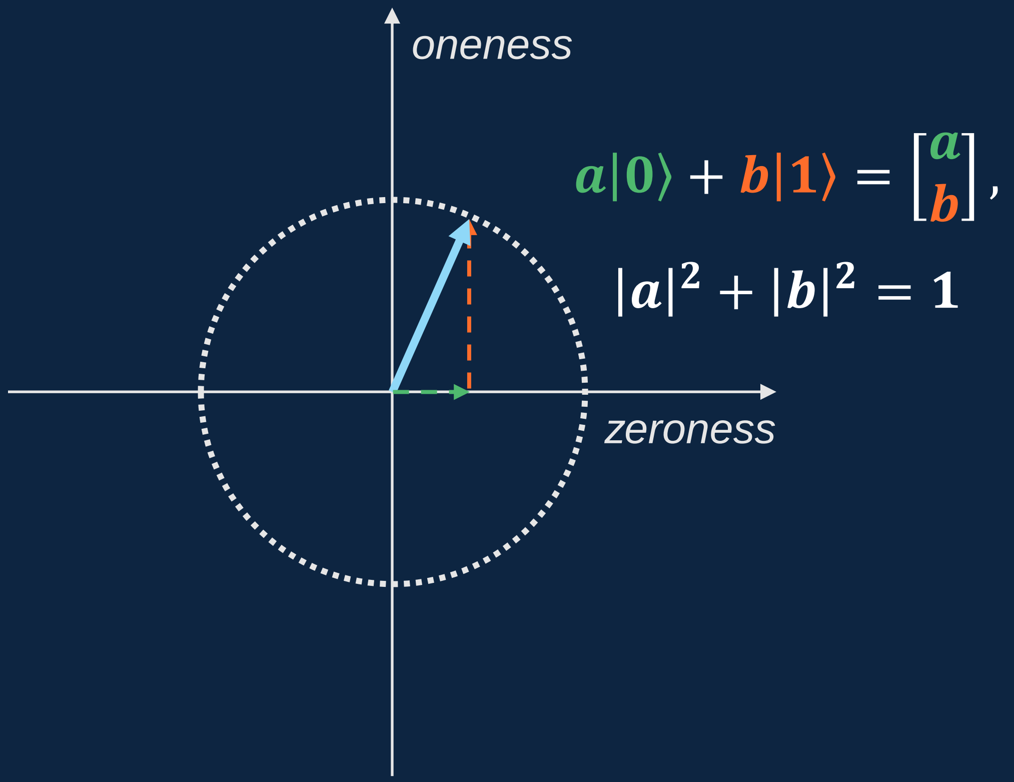
The classical software analog is a data structure with two floating point numbers

A qubit state is expressed mathematically as a vector.

Zero and One

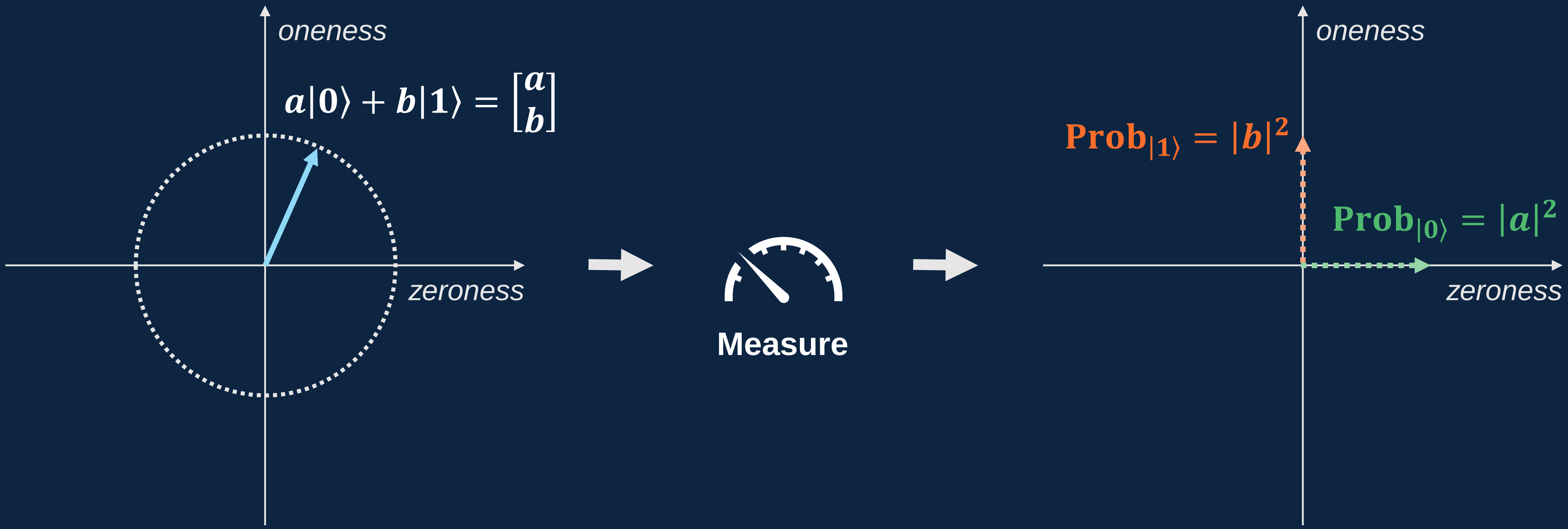


Arbitrary Qubit

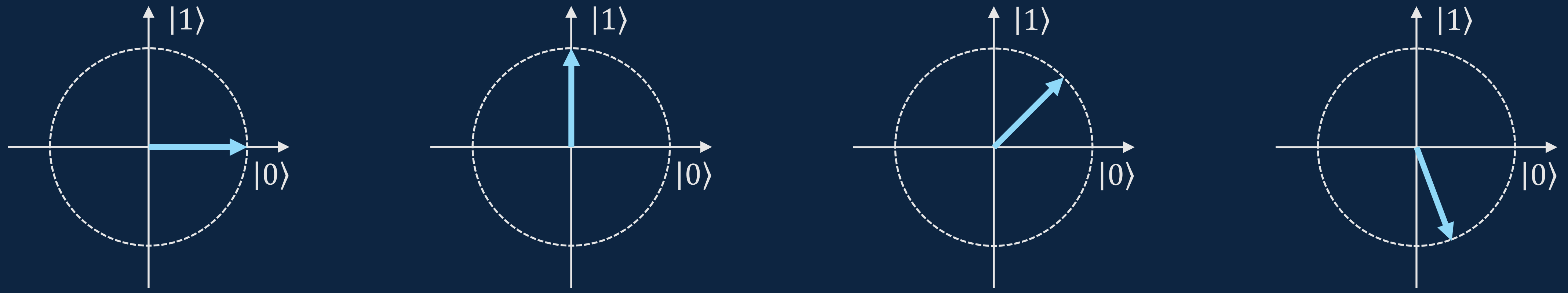


MITRE

When a qubit is measured, it becomes a |0⟩ or |1⟩ with probability equal to the amplitude squared.

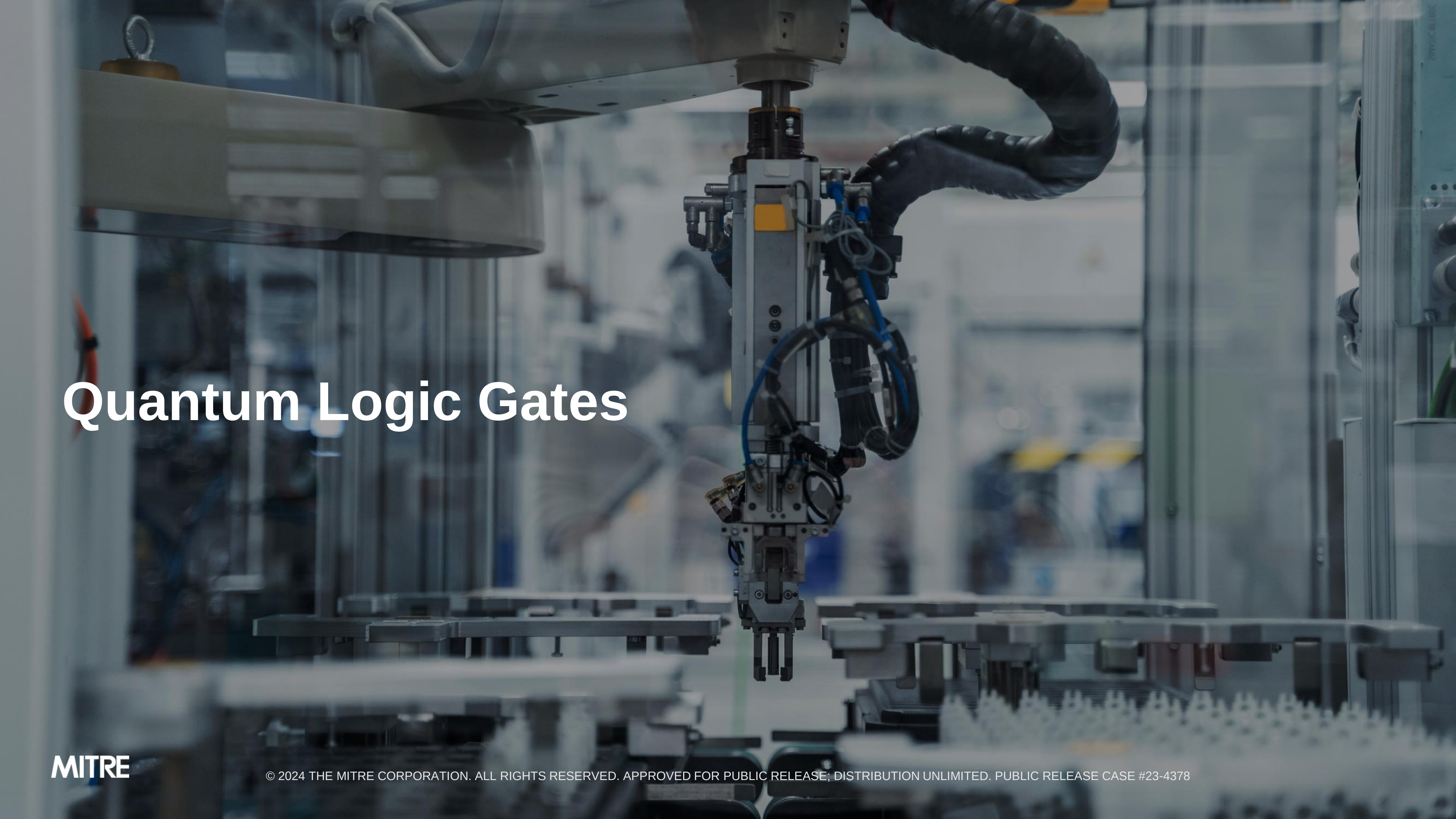


For each of these states, what is the probability of observing a $|1\rangle$ when the qubit is measured?



$\mathbf{|0\rangle}$	$\mathbf{|1\rangle}$	$\mathbf{\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)}$	$\mathbf{\frac{1}{2}|0\rangle - \frac{\sqrt{3}}{2}|1\rangle}$
0%	**100%**	**50%**	**75%**

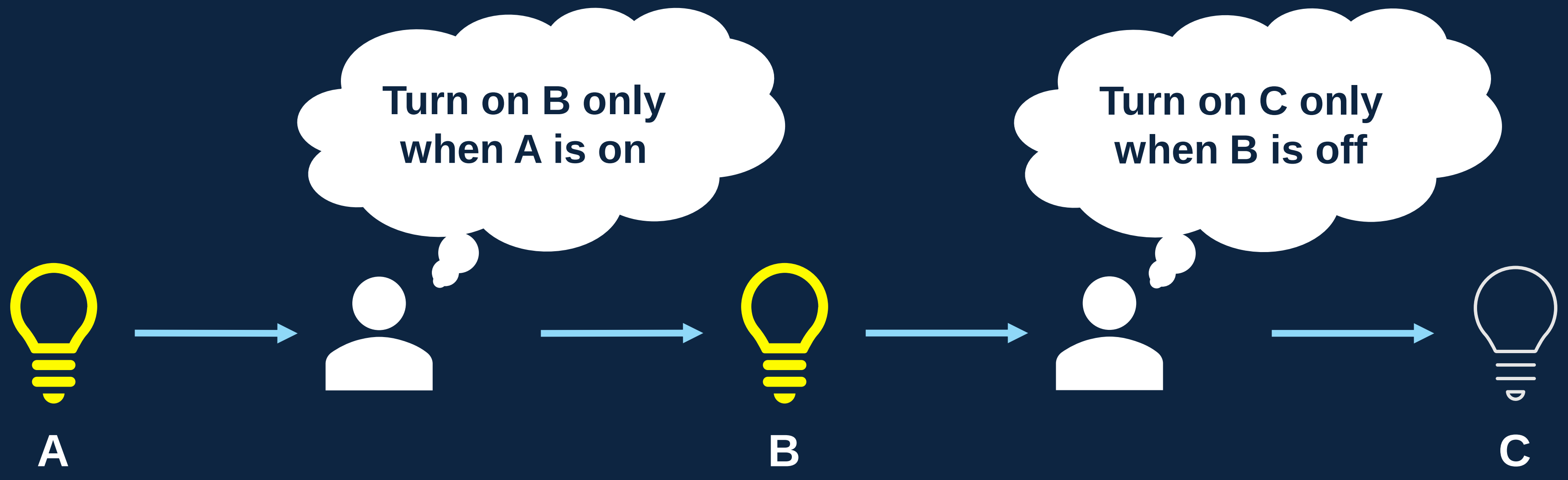
MITRE



Quantum Logic Gates

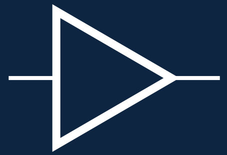
MITRE

Classical computation is a series of small decisions performed by building blocks called logic gates.



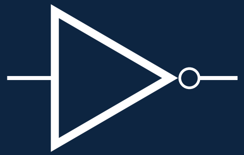
MITRE

Digital logic gates are defined by the resulting output for each possible input combination.



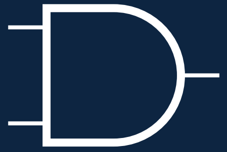
Unity

A	A
0	0
1	1



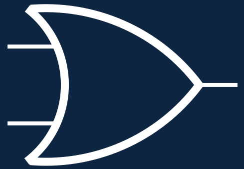
NOT

A	$\overline{A}$
0	1
1	0



AND

A	B	$\boldsymbol{A \wedge B}$
0	0	0
0	1	0
1	0	0
1	1	1



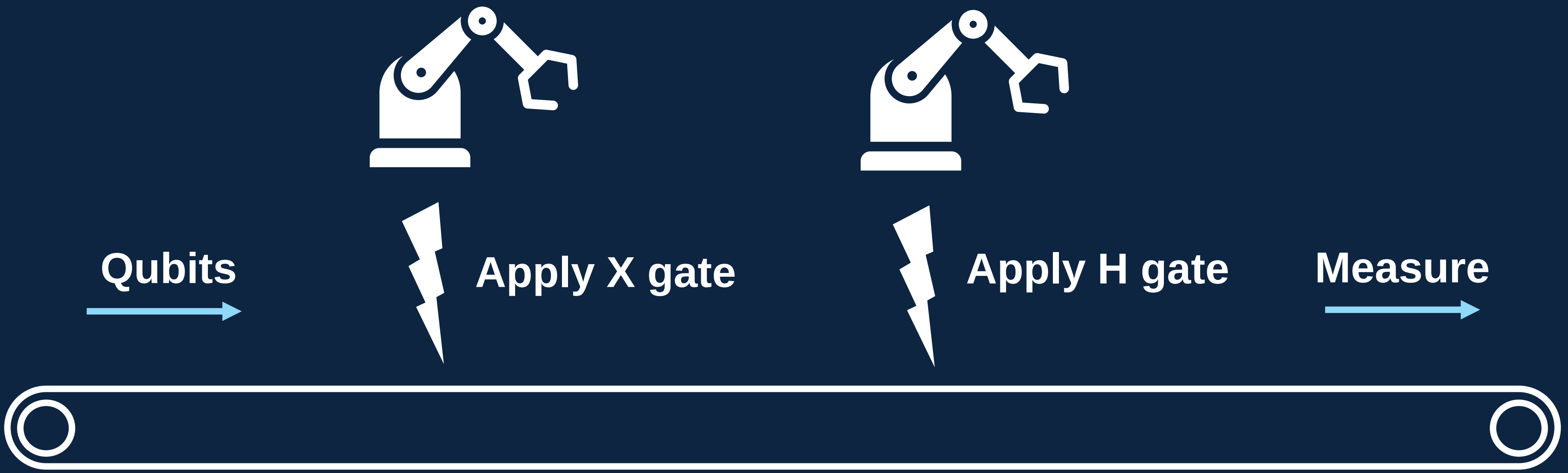
OR

A	B	$\boldsymbol{A \vee B}$
0	0	0
0	1	1
1	0	1
1	1	1

XOR

A	B	$\boldsymbol{A \veebar B}$
0	0	0
0	1	1
1	0	1
1	1	0

Quantum computation is a series of small modifications to the state of the quantum system.



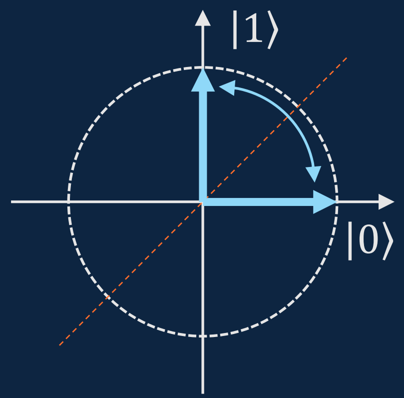
Quantum logic gates are defined as matrices that transform the state vector of one or more qubits.

X (Not)

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Definition

Symbol



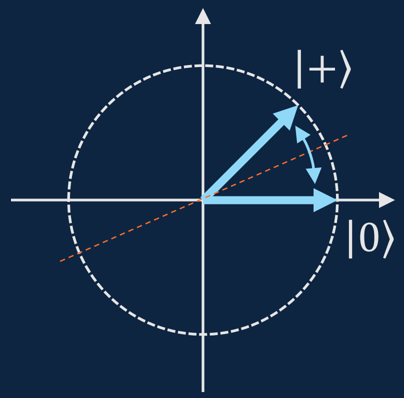
Toggles between $|0\rangle$ and $|1\rangle$

H (Hadamard)

$$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

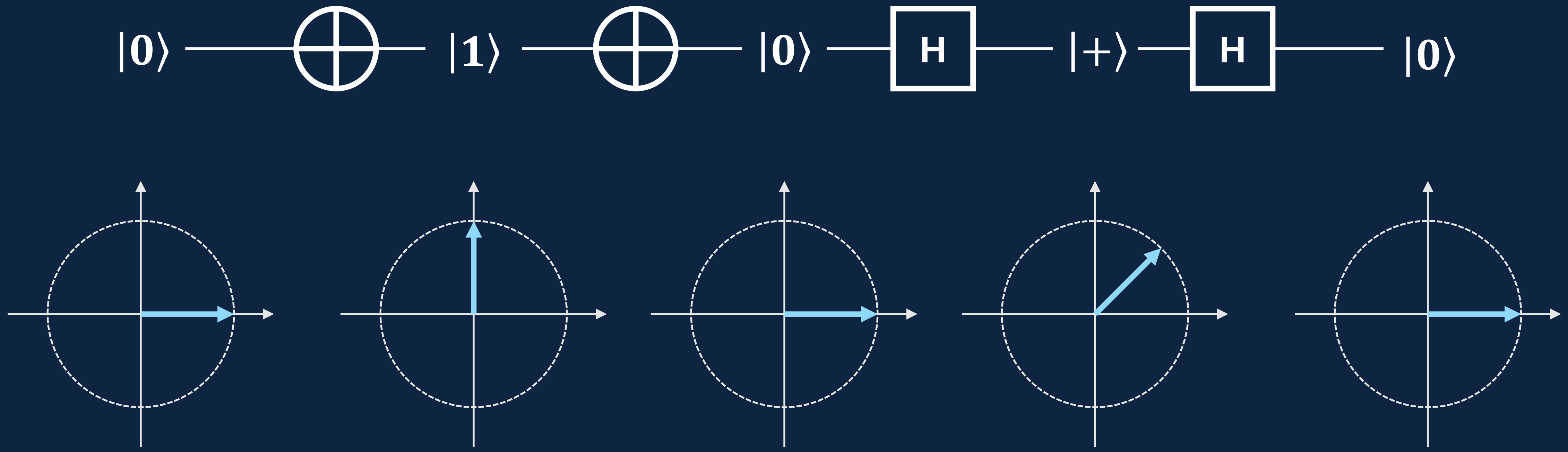
Definition

Symbol



Toggles between $|0\rangle$ and $|+\rangle$

What is the state of the qubit after each gate?



$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$