

# Quantum Computing

*for “classical” developers*

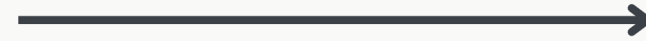


## Intro: Key takeaway

---



"Classical"  
Computer



is being  
replaced by



Quantum  
Computer



"Classical"  
Computer



is being  
**enhanced** by



Quantum  
Computer

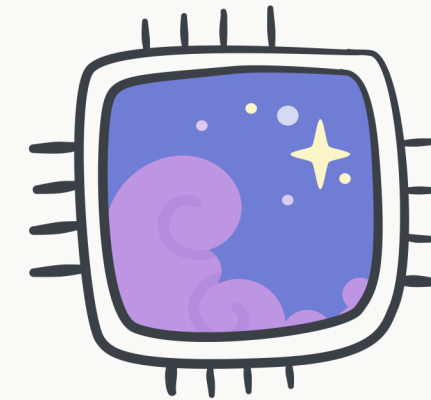




"Classical"  
Computer



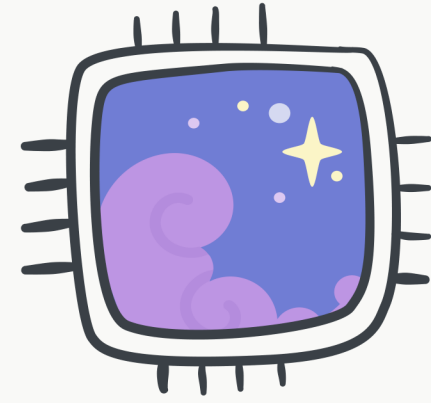
is being  
**enhanced** by



Quantum  
Processing Unit  
(QPU)



"Classical"  
Computer



Quantum Processing  
Unit (QPU)

## Part I: Quantum computing fundamentals

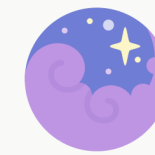
---

**“Classical” bit**



vs.

**“Quantum” bit**



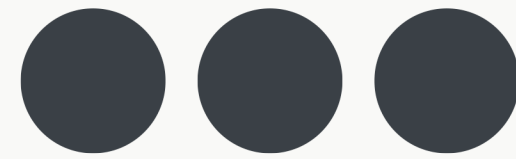


## “Classical” bit



Can be in one of two possible  
states, 0 or 1

**n “classical” bits**



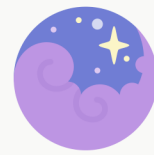
Can be in 1 of  $2^n$  states

# “Quantum” bit



can be in two possible states,  
0 or 1, simultaneously

# “Quantum” bit

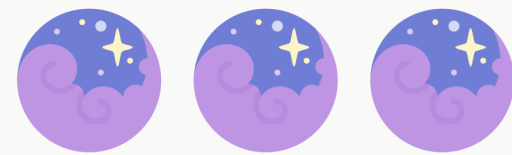


can be in two possible states,  
0 or 1, simultaneously

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



n “quantum” bits



can be in  $2^n$  states  
simultaneously

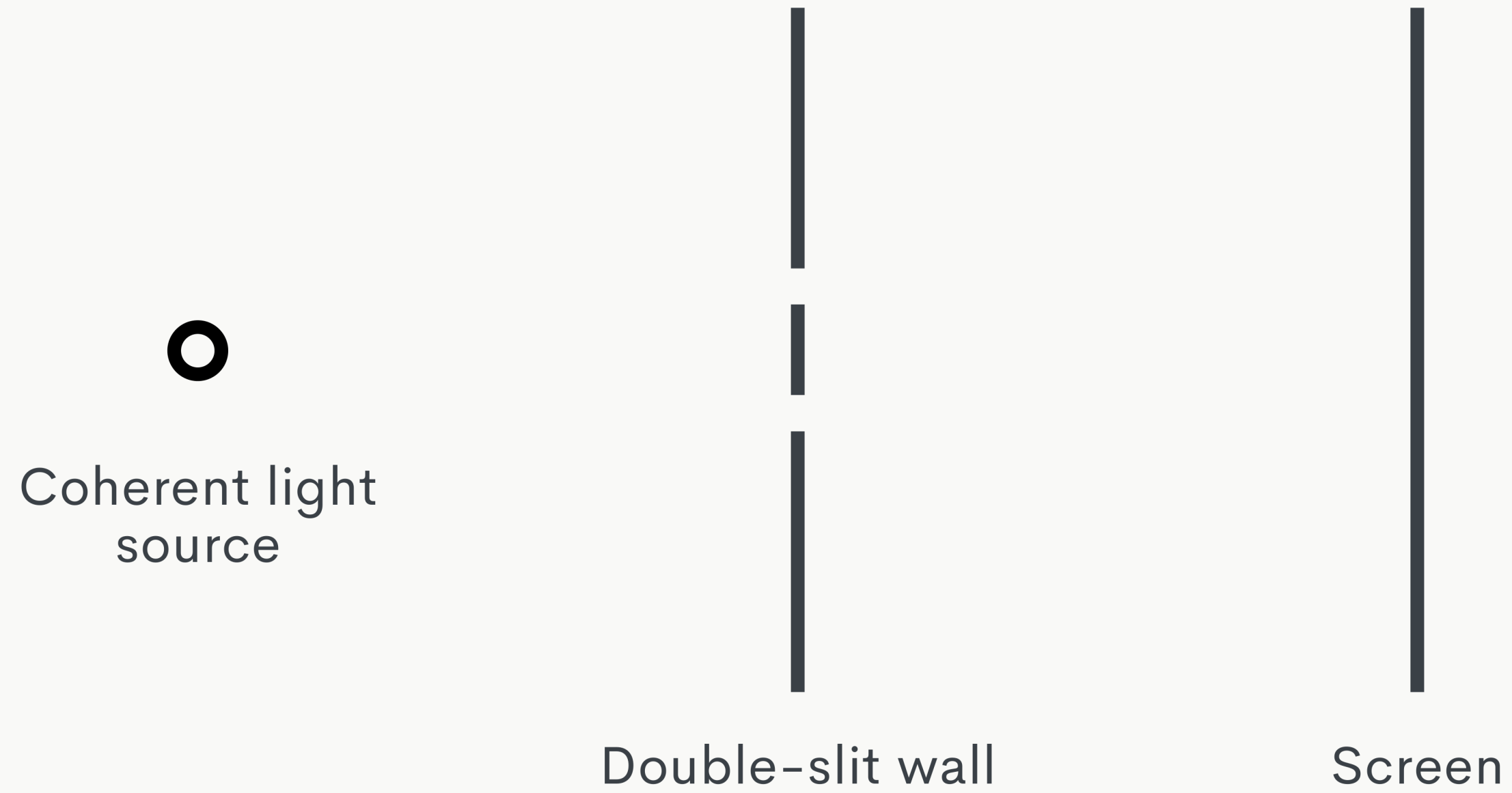
$$|0\rangle + |1\rangle + |2\rangle + \dots + |2^n\rangle$$

## Part I: Quantum computing fundamentals — Superposition & interference

---

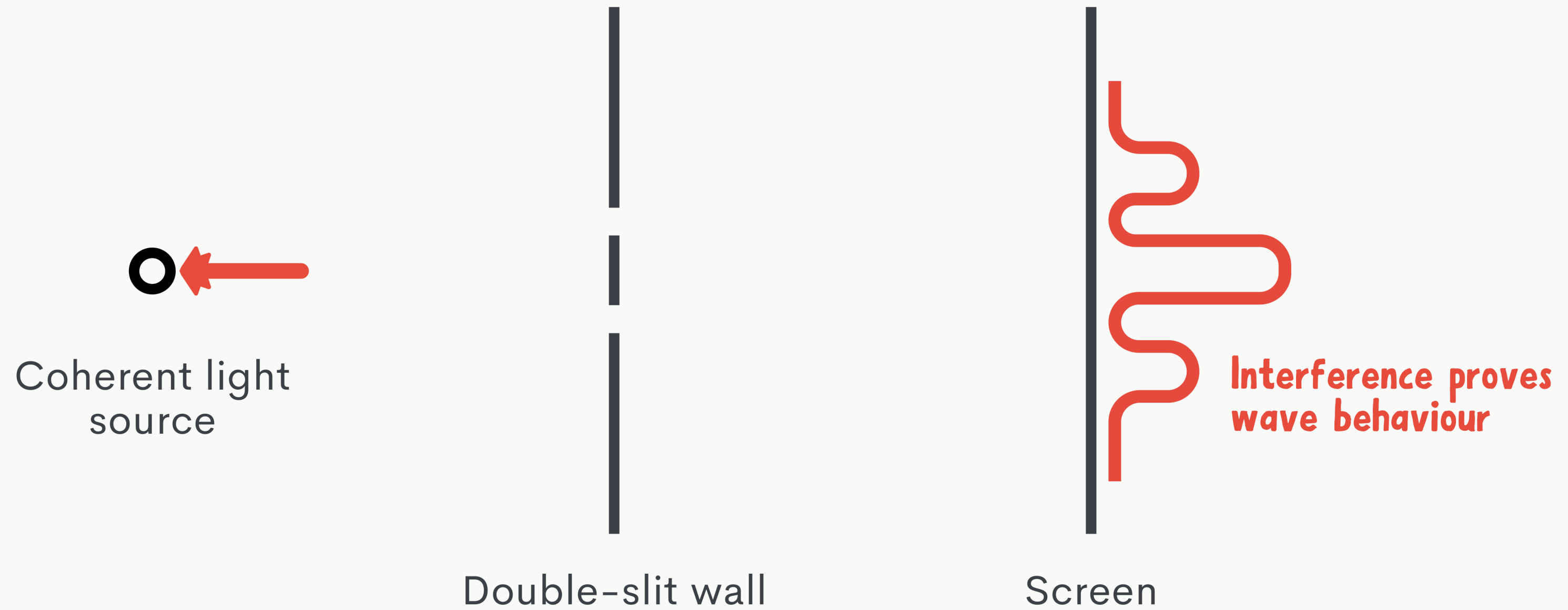
# Part I: Quantum computing fundamentals — Superposition & interference

---



# Part I: Quantum computing fundamentals — Superposition & interference

---





**When we try to observe which slit the light goes through**



Coherent light source



Double-slit wall



Screen

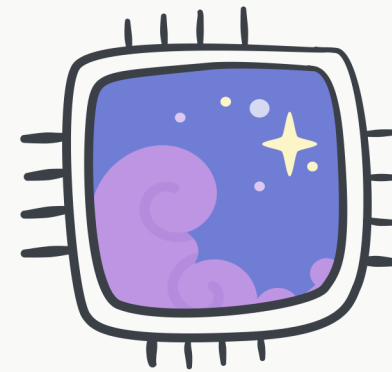


**No interference proves particle behaviour**

## Part I: Quantum computing fundamentals — Qubit superposition & interference

---

$$x + 2 = 3$$



Quantum  
circuit

$$x = |0\rangle + |1\rangle + |2\rangle + |3\rangle + \dots \longrightarrow |0, false\rangle + |1, true\rangle + |2, false\rangle + \dots$$

$$|0, \textit{false}\rangle + |1, \textit{true}\rangle + |2, \textit{false}\rangle + \dots$$





State collapses when being measured to a single value

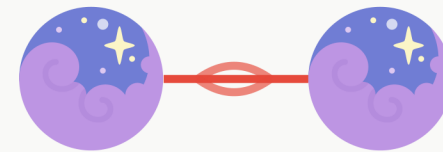
$$\langle 0, \text{true} \rangle + \langle 1, \text{true} \rangle + |2, \text{false}\rangle + \langle 3, \text{false} \rangle$$

## Part I: Quantum computing fundamentals — Entanglement

---

## Part I: Quantum computing fundamentals — Entanglement

---



Entangled  
qubits





**When we observe one, both collapse  
so we know the value of both qubits**



1



0



“Spooky action  
at a distance”

— Albert Einstein

## Part II: Shor's algorithm breaking RSA encryption

---

## Part II: Shor's algorithm breaking RSA encryption

---



Modern  
RSA encryption





Modern  
RSA encryption

$$N = a \times b$$

Really large  
number

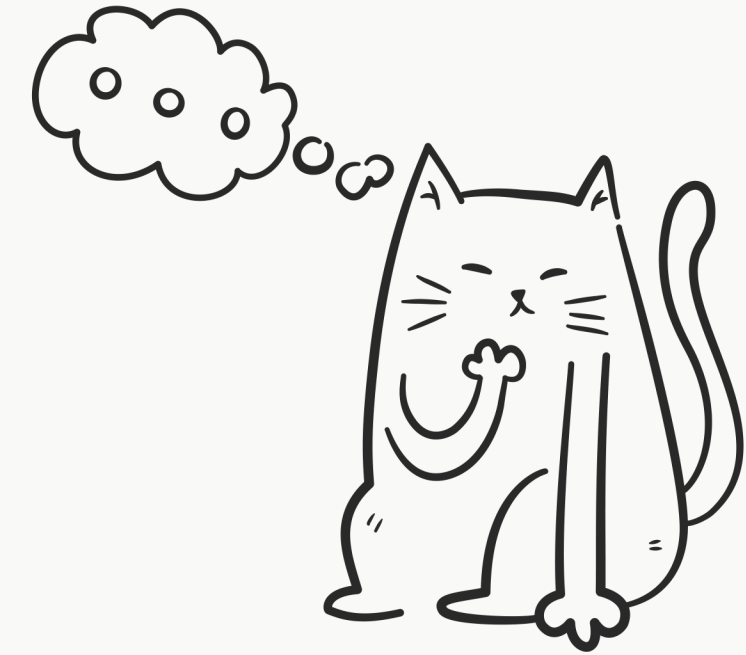
You need its factors to  
decrypt the message



Modern  
RSA encryption

$$N = a \times b$$

Really large  
number



You need its factors to  
decrypt the message

**Even with supercomputer,  
guessing by brute force would  
take over 300 trillion years**



Modern  
RSA encryption

$$N = a \times b$$

Really large  
number

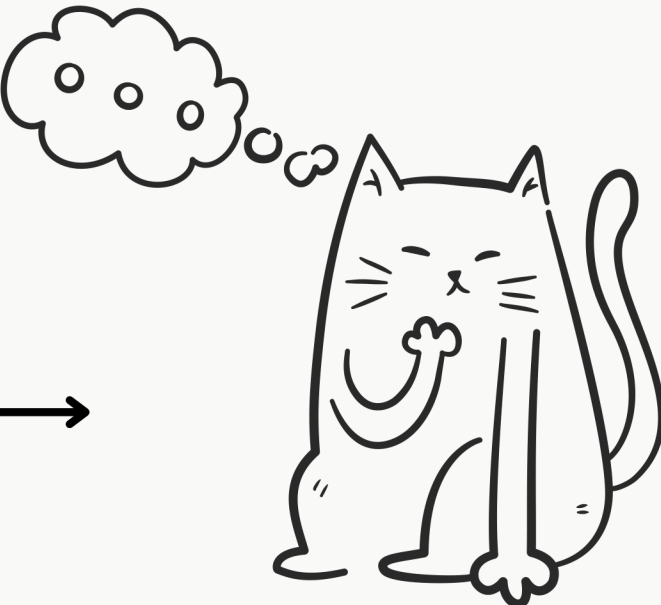
You need its factors to  
decrypt the message

Part II: Shor's algorithm breaking RSA encryption

---



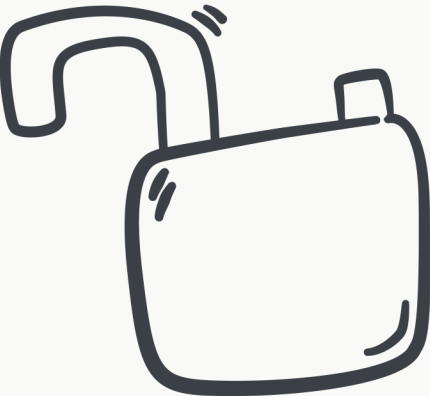
Modern  
RSA encryption



Making a  
crappy guess



300  
trillion  
years



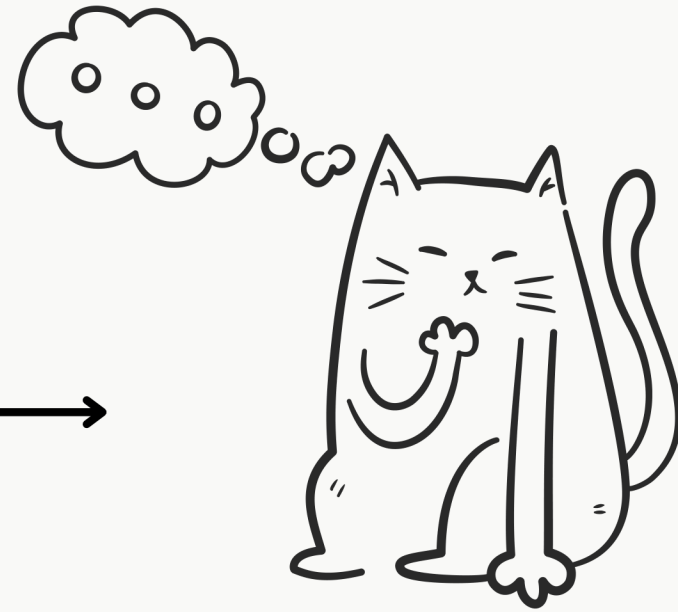
Break  
encryption

## Part II: Shor's algorithm breaking RSA encryption

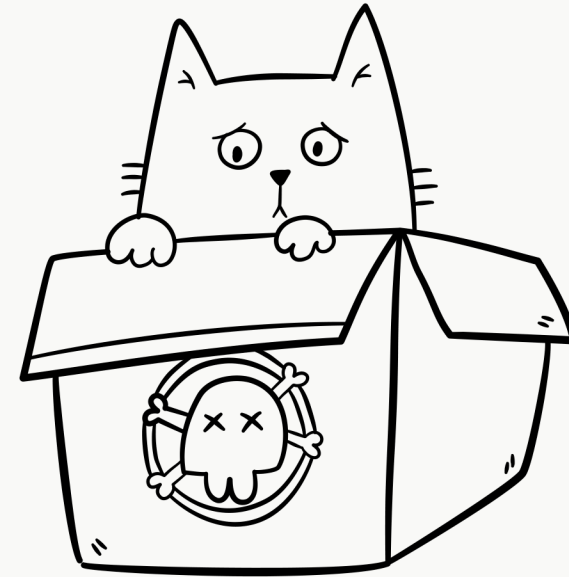
---



Modern  
RSA encryption



Making a  
crappy guess



"magic box"  
turns bad  
guess into  
good one

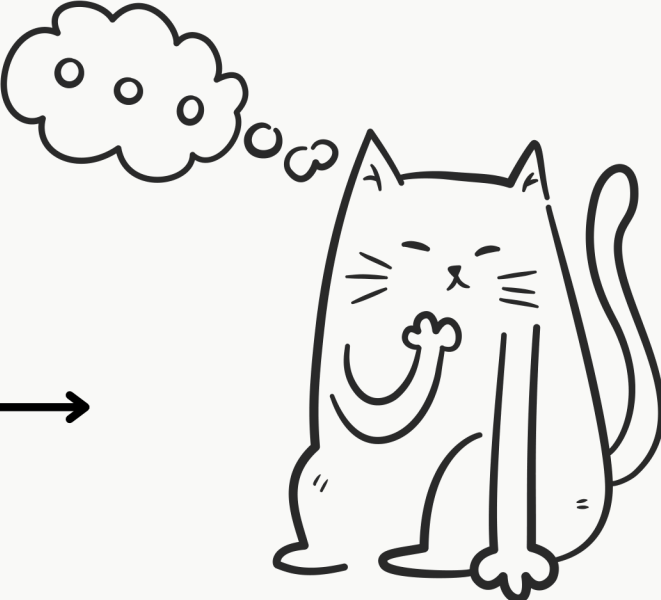


Break  
encryption

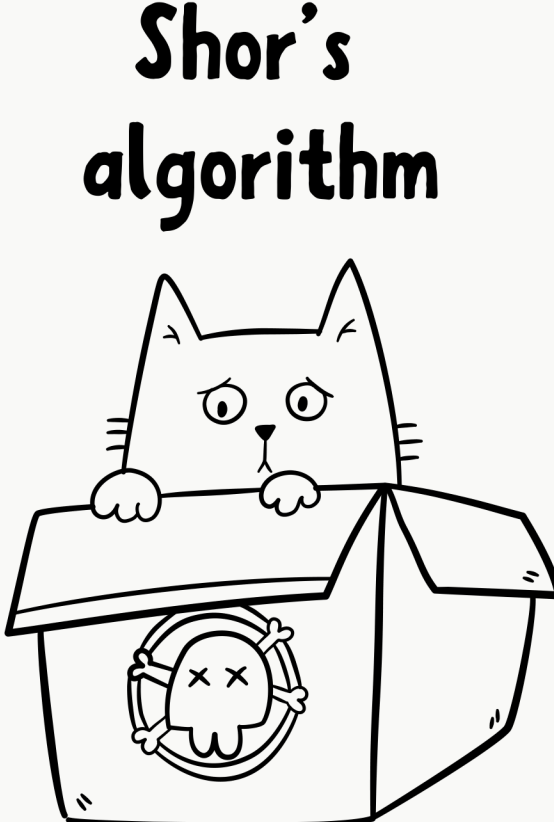
Part II: Shor's algorithm breaking RSA encryption



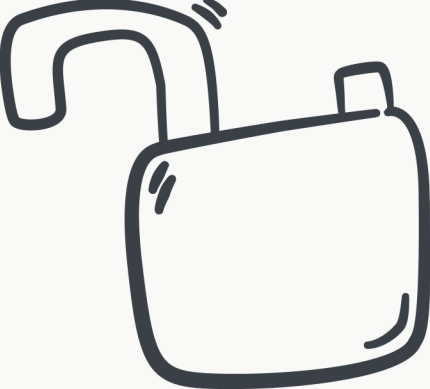
Modern  
RSA encryption



Making a  
crappy guess



"magic box"  
turns bad  
guess into  
good one



Break  
encryption

$$A \times A \times A \times \dots \times A = m \times B + 1$$

$$A^p = m \times B + 1$$

$$g^p = m \times N + 1$$

$$(g^{p/2} + 1) \times (g^{p/2} - 1) = m \times N$$



$$g^p = m \times N + 1$$

$$(g^{p/2} + 1) \times (g^{p/2} - 1) = m \times N$$

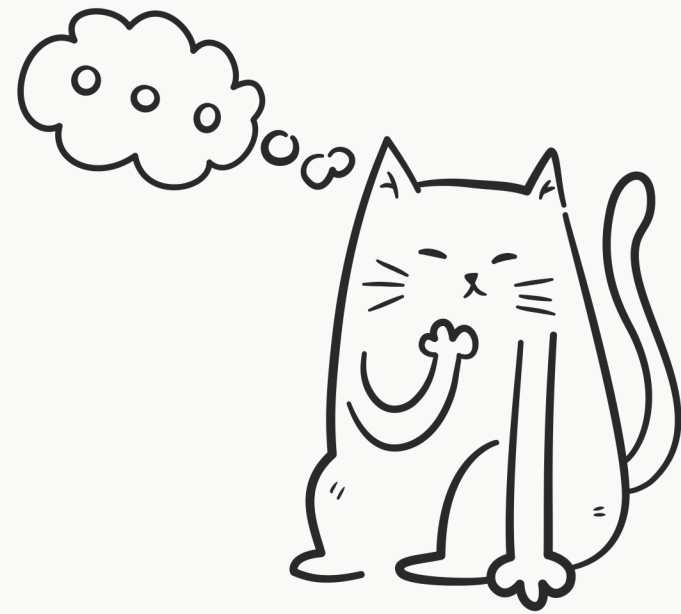
**Something**

**Something else**

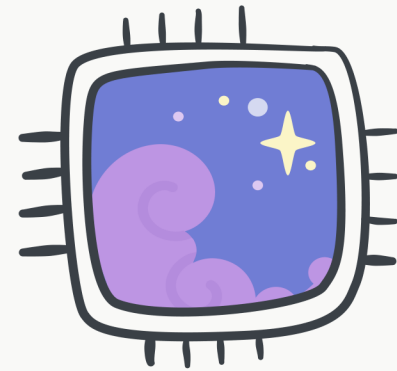
**Shares factors  
with N**

## Part II: Shor's algorithm breaking RSA encryption

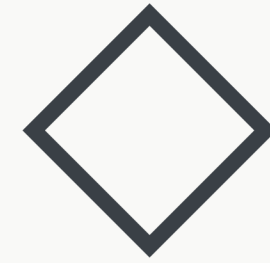
---



Making a  
crappy guess



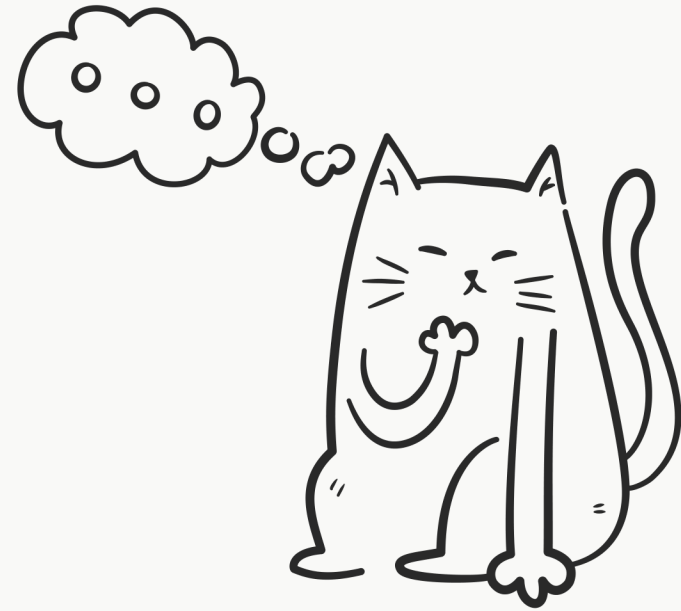
Shor's  
algorithm  
turns bad  
guess into  
good one



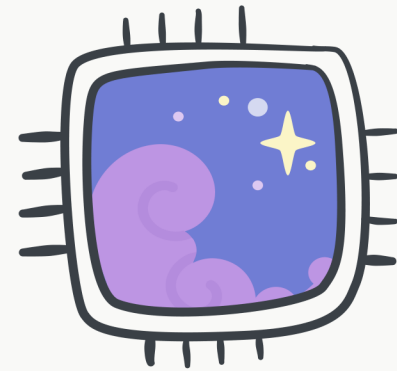
Check for  
problems

## Part II: Shor's algorithm breaking RSA encryption

---



Making a  
crappy guess



Shor's  
algorithm  
turns bad  
guess into  
good one

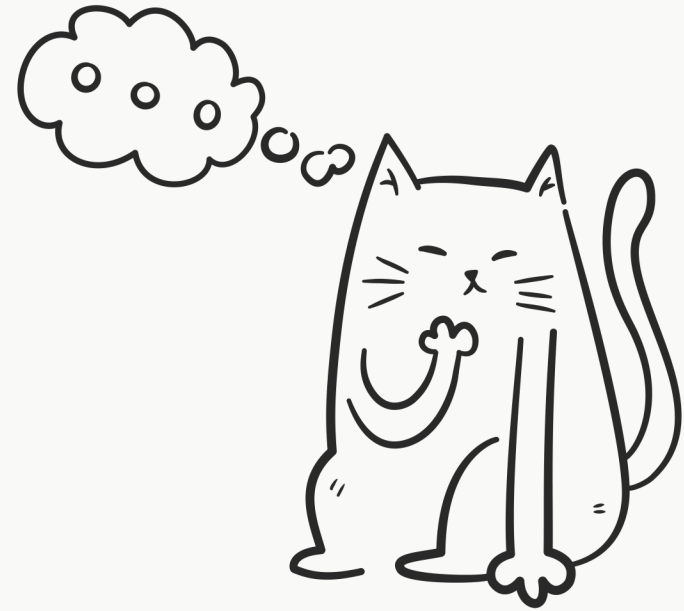


Check for  
problems

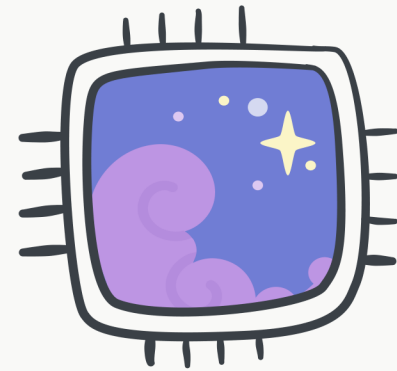
**Multiple of N?**

## Part II: Shor's algorithm breaking RSA encryption

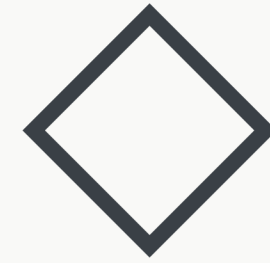
---



Making a  
crappy guess



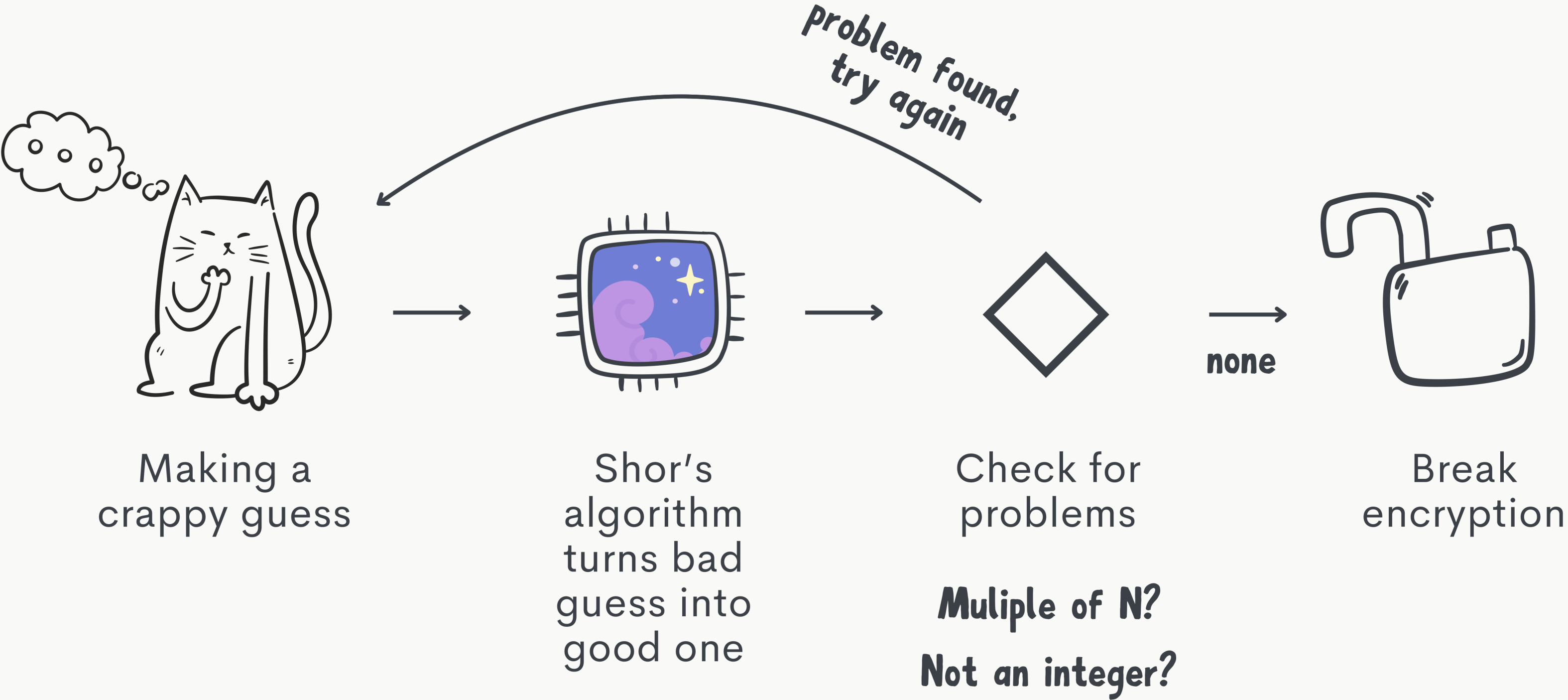
Shor's  
algorithm  
turns bad  
guess into  
good one

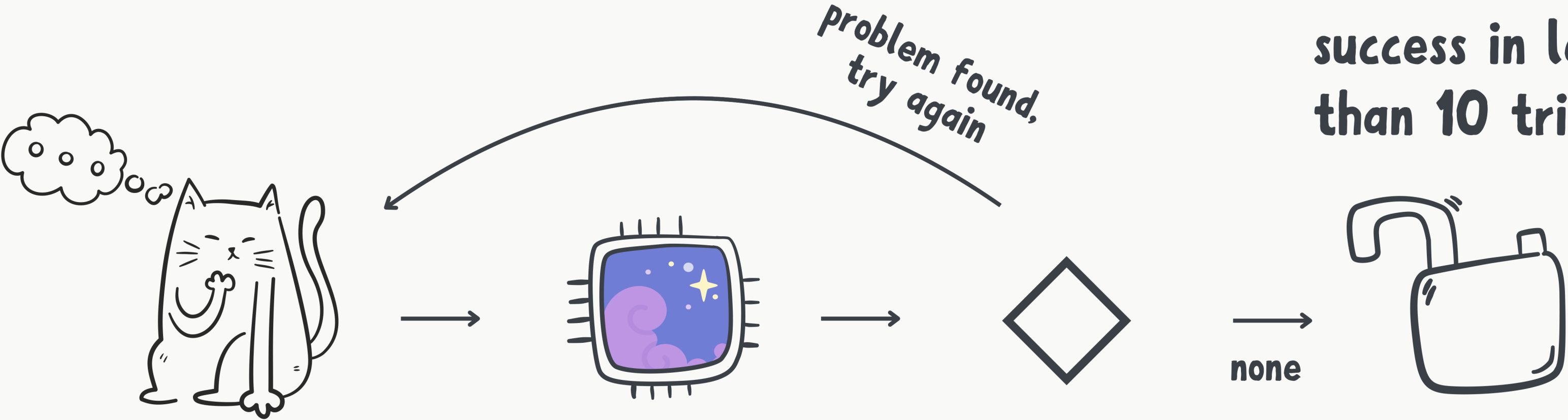


Check for  
problems

**Multiple of  $N$ ?**  
**Not an integer?**

Part II: Shor's algorithm breaking RSA encryption





Making a crappy guess

Shor's algorithm turns bad guess into good one

Check for problems  
**Multiple of N?**  
**Not an integer?**

Break encryption

**99% chance of success in less than 10 tries!**

## Part II: Shor's algorithm breaking RSA encryption

---

$$g^p = m \times N + 1$$



$$g^p = 1 \pmod{N}$$

$$g^p = 1 \pmod{N}$$

$$g^x = r \pmod{N}$$

**e.g. for  $x=21$ , we might  
get a remainder  $r=3$**

$$g^p = 1 \pmod{N}$$

$$g^x = r \pmod{N}$$

$$g^{x+p} = r \pmod{N}$$

**e.g. for  $x=21$ , we might  
get a remainder  $r=3$**

$$g^p = 1 \pmod{N}$$

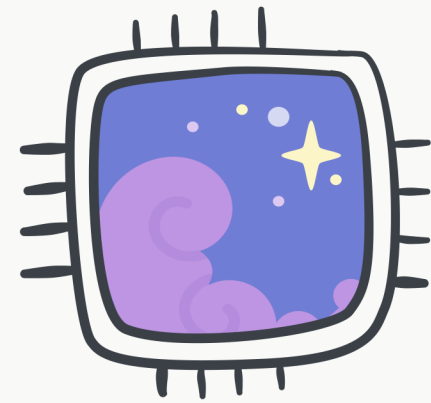
$$g^x = r \pmod{N}$$

$$g^{x+p} = r \pmod{N}$$

$$g^{x+2p} = r \pmod{N}$$

**e.g. for  $x=21$ , we might  
get a remainder  $r=3$**

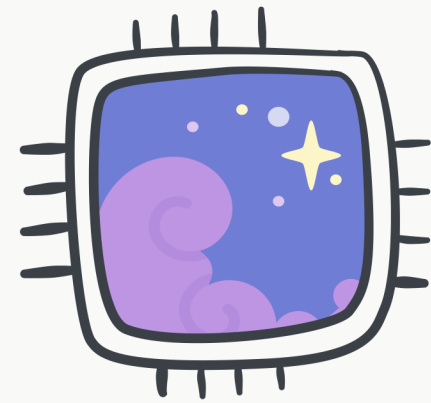
$$g^x = r \pmod{N}$$



Quantum  
circuit



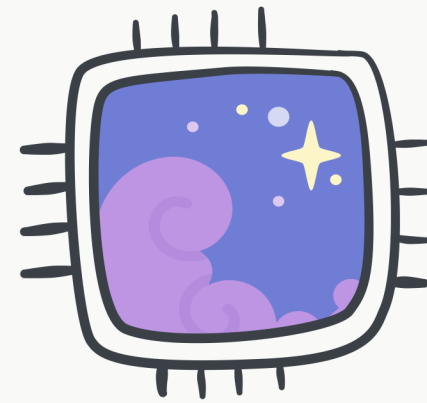
$$g^x = r \pmod{N}$$



Quantum  
circuit

$$x = |0\rangle + |1\rangle + |2\rangle + \dots \longrightarrow$$

$$g^x = r \pmod{N}$$



Quantum  
circuit

$$x = |0\rangle + |1\rangle + |2\rangle + \dots \quad \longrightarrow \quad |0, +17\rangle + |1, +3\rangle + |2, +92\rangle + \dots$$

## Part II: Shor's algorithm breaking RSA encryption

---



$$|0, +17\rangle + |1, +3\rangle + |2, +92\rangle + \dots$$





**only measure the  
remainder value**

$$|1, +3\rangle + |11, +3\rangle + |21, +3\rangle + \dots$$



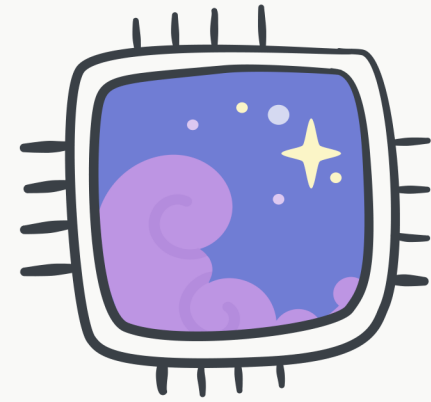
**only measure the  
remainder value**

$$|1\rangle + |11\rangle + |21\rangle + |31\rangle + |41\rangle + \dots$$

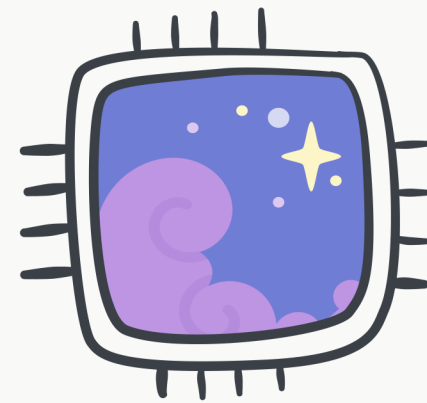
**the resulting superposition will  
have a frequency of 1 over p**

## Part II: Shor's algorithm breaking RSA encryption

---



Quantum  
Fourier  
Transform



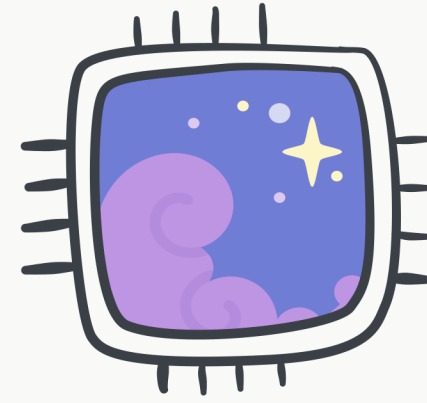
Quantum  
Fourier  
Transform

$$|1\rangle + |11\rangle + |21\rangle + |31\rangle + \dots \longrightarrow 1 \div 10$$

**the result is the frequency  
of the superposition**



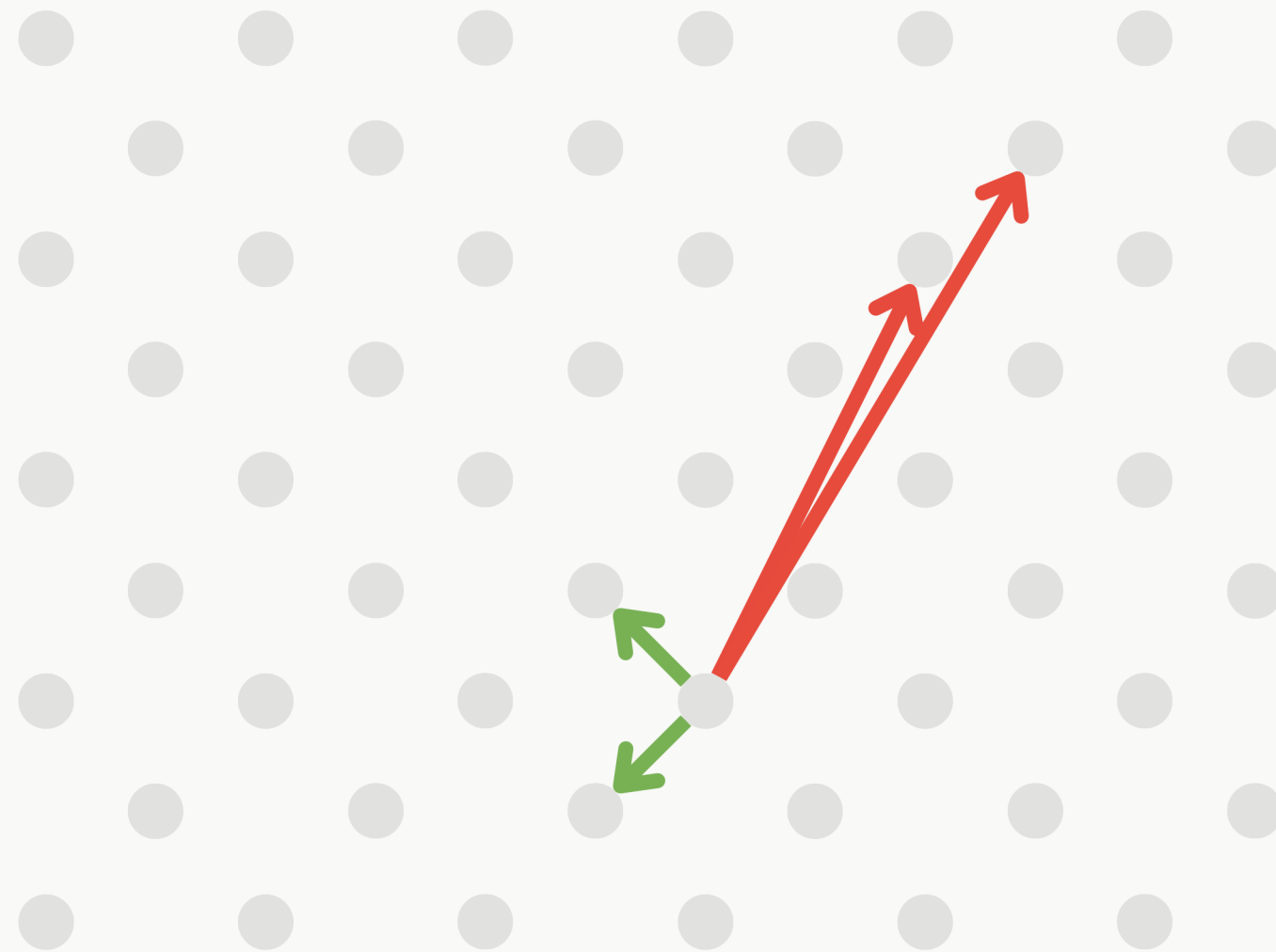
"Classical"  
Computer



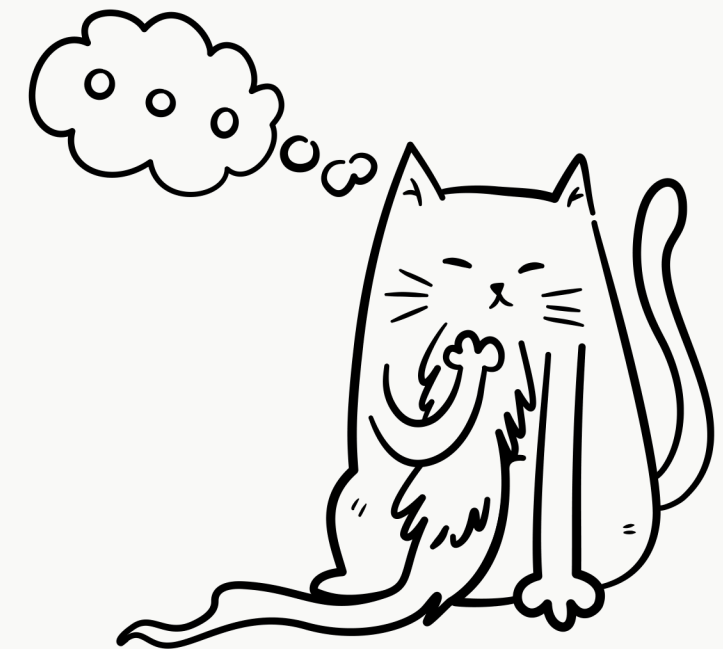
QPU: modulo  
calculation + QFT

## Part II: Shor's algorithm breaking RSA encryption — Post quantum cryptography

---



— good basis  
— bad basis



**finding the closest point  
with vectors of the bad  
basis in higher dimensions  
is really hard**

## Part II: Shor's algorithm breaking RSA encryption — Current challenges with QPUs

---



## Problem 1: Number of qubits

E.g. to run Shor's algorithm on RSA-2048, you would need around 2 million "noisy" qubits to have sufficient memory

<https://quantum-journal.org/papers/q-2021-04-15-433/>

[https://en.wikipedia.org/wiki/List\\_of\\_quantum\\_processors](https://en.wikipedia.org/wiki/List_of_quantum_processors)

## **Problem 1: Number of qubits**

E.g. to run Shor's algorithm on RSA-2048, you would need around 2 million "noisy" qubits to have sufficient memory

## **Problem 2: Error correction**

It is really hard to prevent any unintentional outside influence that would cause the quantum state irrevertably to collapse

## Part III: Practical applications of quantum computing

---

# Quantum Machine Learning (QML)

# Quantum Machine Learning (QML)

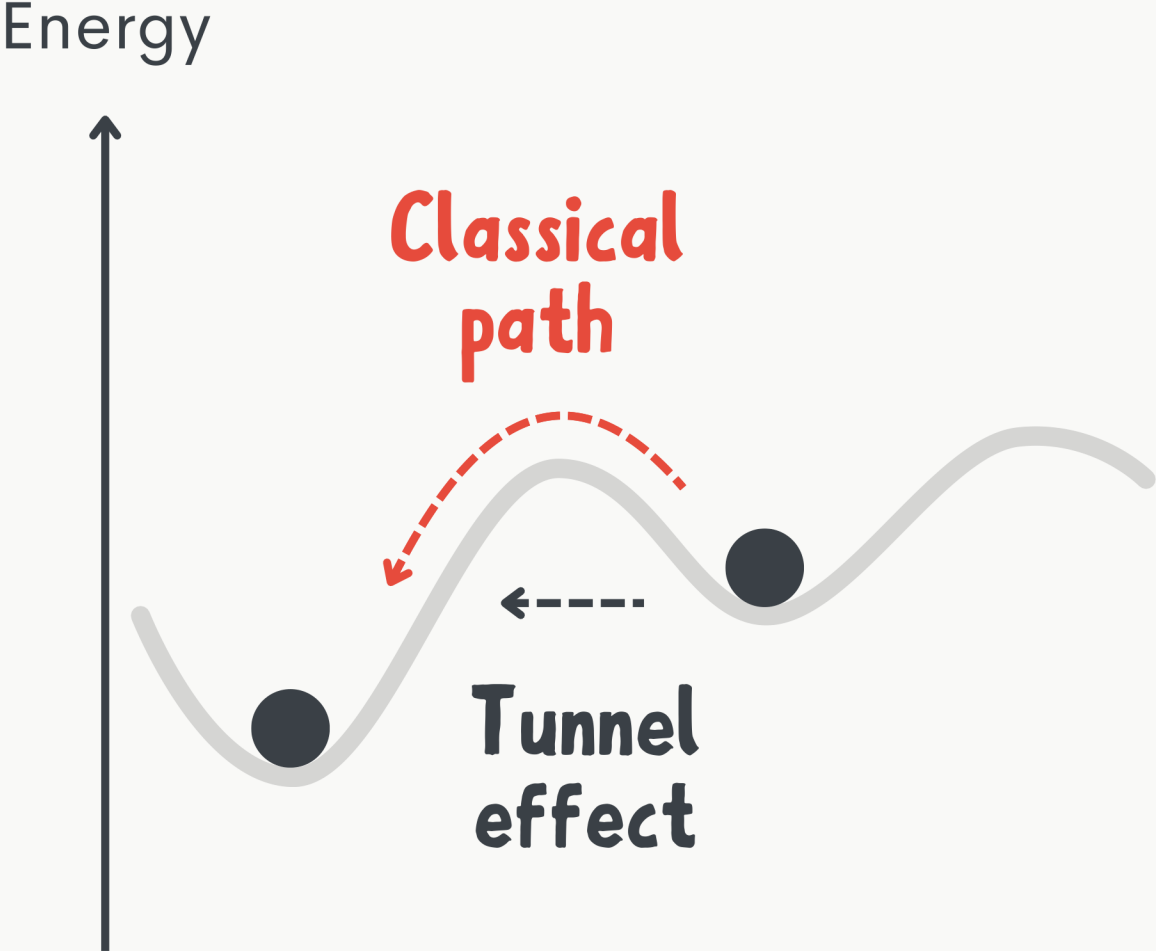
- Quantum Fourier Transforms

# Quantum Machine Learning (QML)

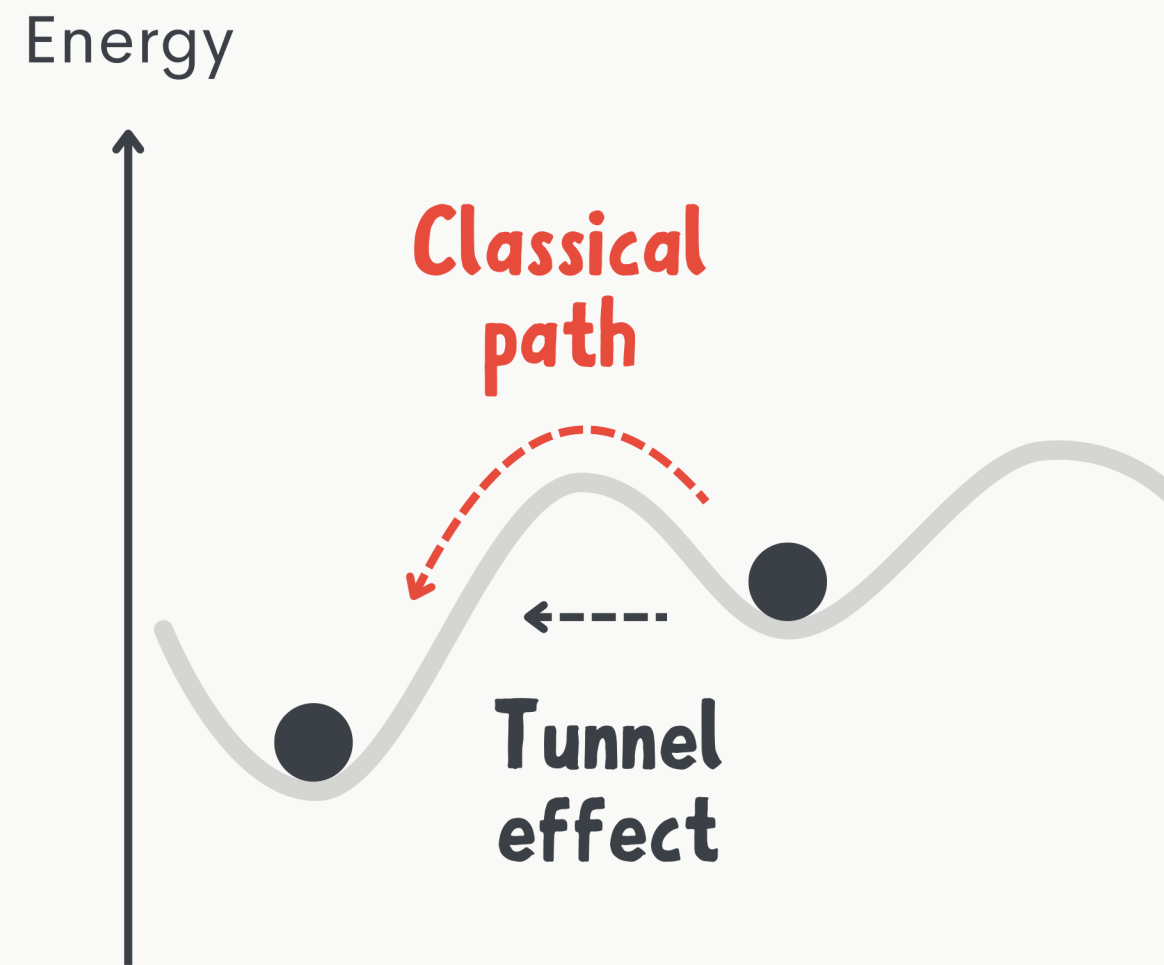
- Quantum Fourier Transforms
- Grover's algorithm

# Quantum Machine Learning (QML)

- Quantum Fourier Transforms
- Grover's algorithm
- Quantum annealing







**This can cause “vacuum decay”  
and destroy the whole universe**



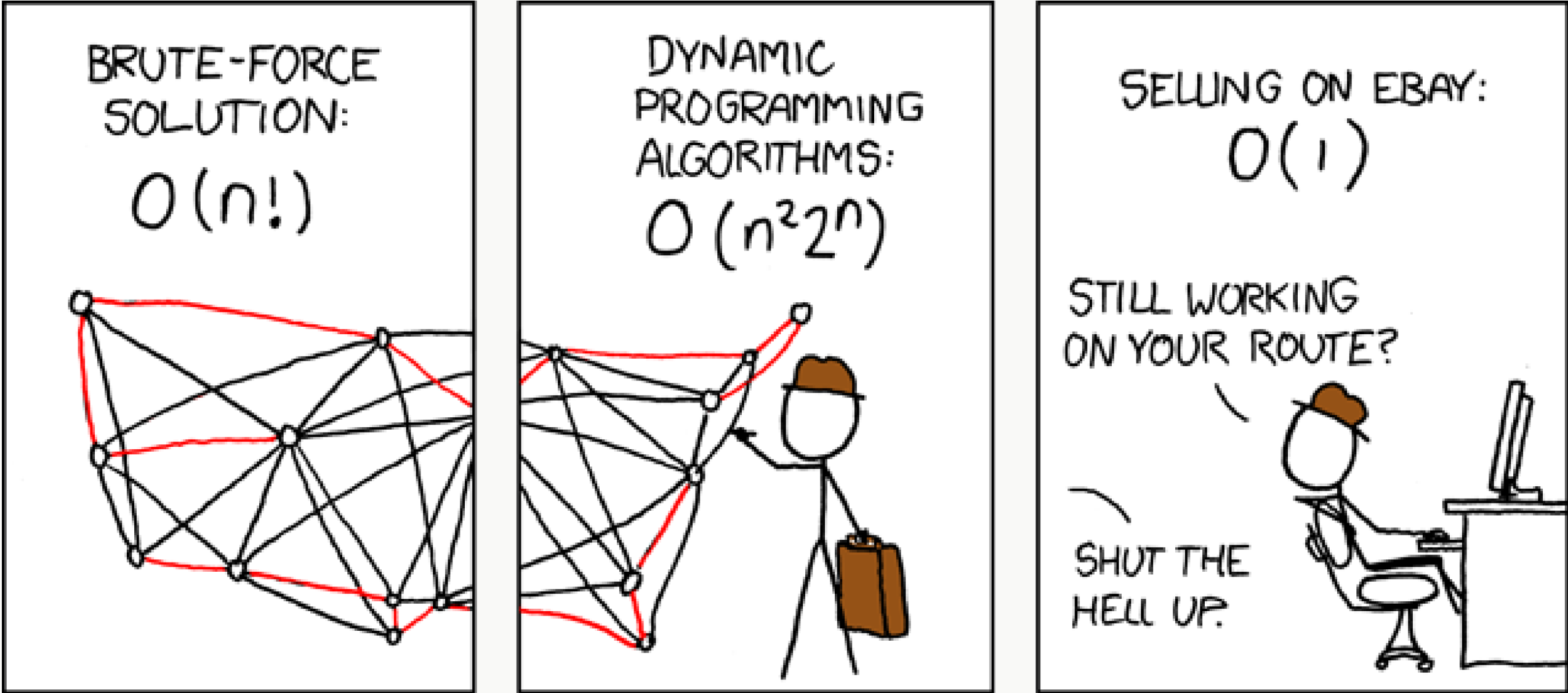
# Quantum Machine Learning (QML)

- Quantum Fourier Transforms
- Grover's algorithm
- Quantum annealing

# Quantum Simulations

# Quantum Simulations

- Optimisation problems



# Quantum Simulations

- Optimisation problems

# Quantum Simulations

- Optimisation problems
  - Supply chain management

# Quantum Simulations

- Optimisation problems
  - Supply chain management
  - Financial modelling and portfolio optimisations



# Quantum Simulations

- Optimisation problems
  - Supply chain management
  - Financial modelling and portfolio optimisations
  - Traffic and fleet management optimisations

# Quantum Simulations

- Optimisation problems
- Simulating quantum systems

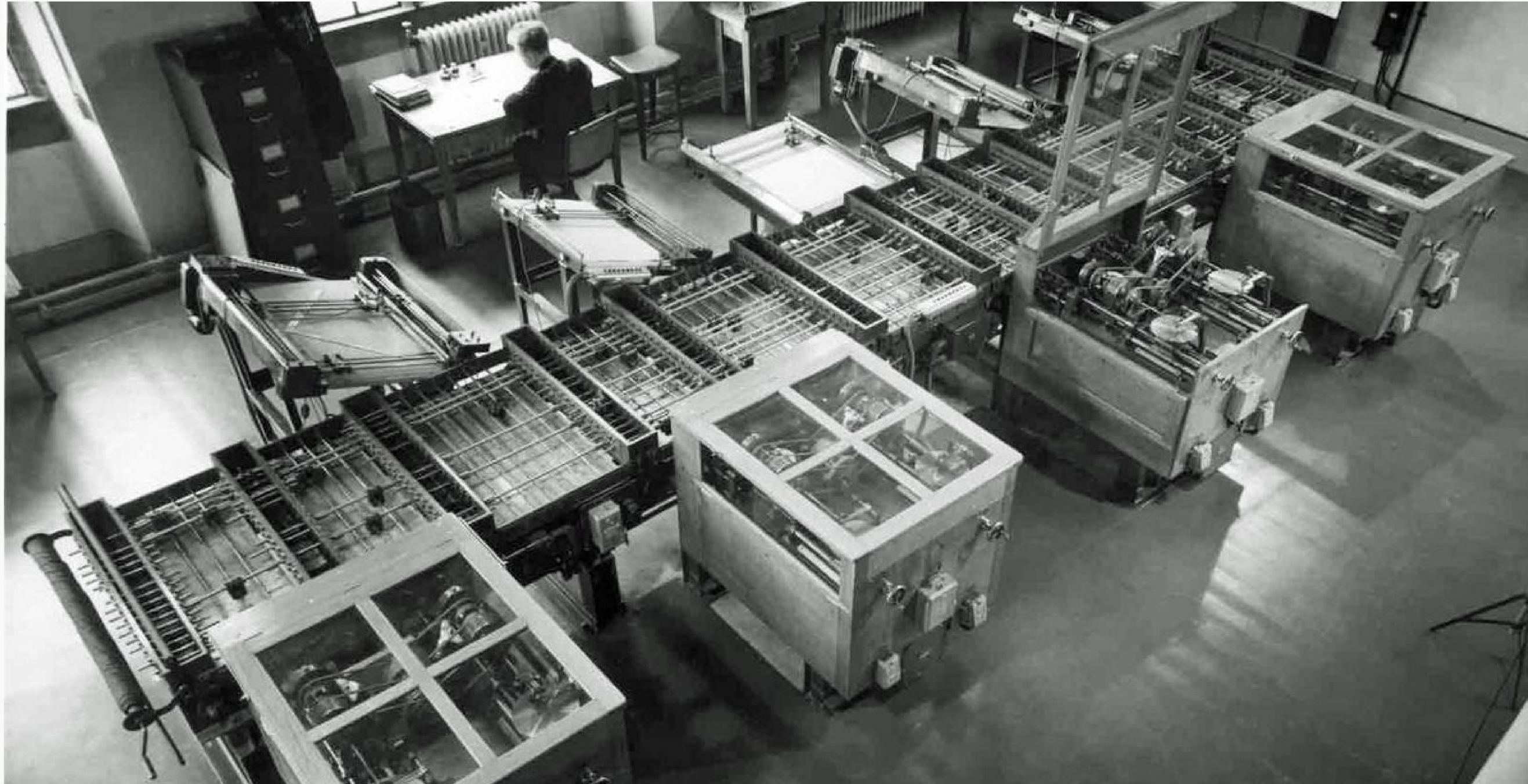
“Nature isn't classical, dammit, and if you want to make a simulation of nature, you'd better make it quantum mechanical.”



— Richard Feynman

## Part III: Practical applications of quantum computing

---



<https://pbs.twimg.com/media/EfCFix5UcAASXVz.jpg>

[http://amg.nzfmm.co.nz/differential\\_analyser\\_explained.html](http://amg.nzfmm.co.nz/differential_analyser_explained.html)

# Quantum Simulations

- Optimisation problems
- Simulating quantum systems

# Quantum Simulations

- Optimisation problems
- Simulating quantum systems
  - Weather simulations and forecasting

# Quantum Simulations

- Optimisation problems
- Simulating quantum systems
  - Weather simulations and forecasting
  - Drug manufacturing & protein folding

## Conclusion

---



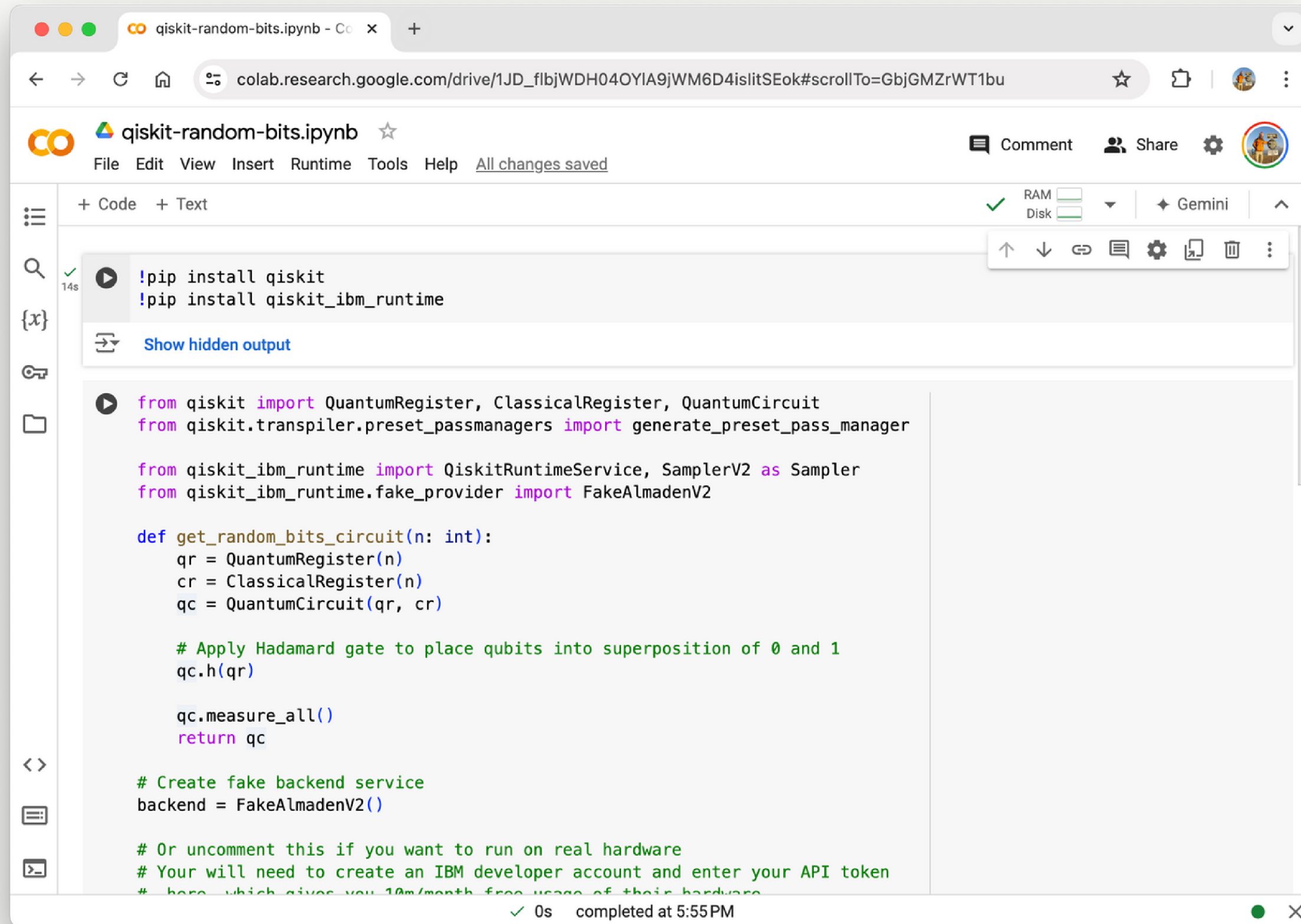
# Qiskit – IBM quantum computing development kit

<https://docs.quantum.ibm.com/start>

<https://github.com/Qiskit>

<https://www.youtube.com/@qiskit>

## Conclusion — Using quantum computing today



```
!pip install qiskit
!pip install qiskit_ibm_runtime

from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
from qiskit.transpiler.preset_passmanagers import generate_preset_pass_manager

from qiskit_ibm_runtime import QiskitRuntimeService, SamplerV2 as Sampler
from qiskit_ibm_runtime.fake_provider import FakeAlmadenV2

def get_random_bits_circuit(n: int):
    qr = QuantumRegister(n)
    cr = ClassicalRegister(n)
    qc = QuantumCircuit(qr, cr)

    # Apply Hadamard gate to place qubits into superposition of 0 and 1
    qc.h(qr)

    qc.measure_all()
    return qc

# Create fake backend service
backend = FakeAlmadenV2()

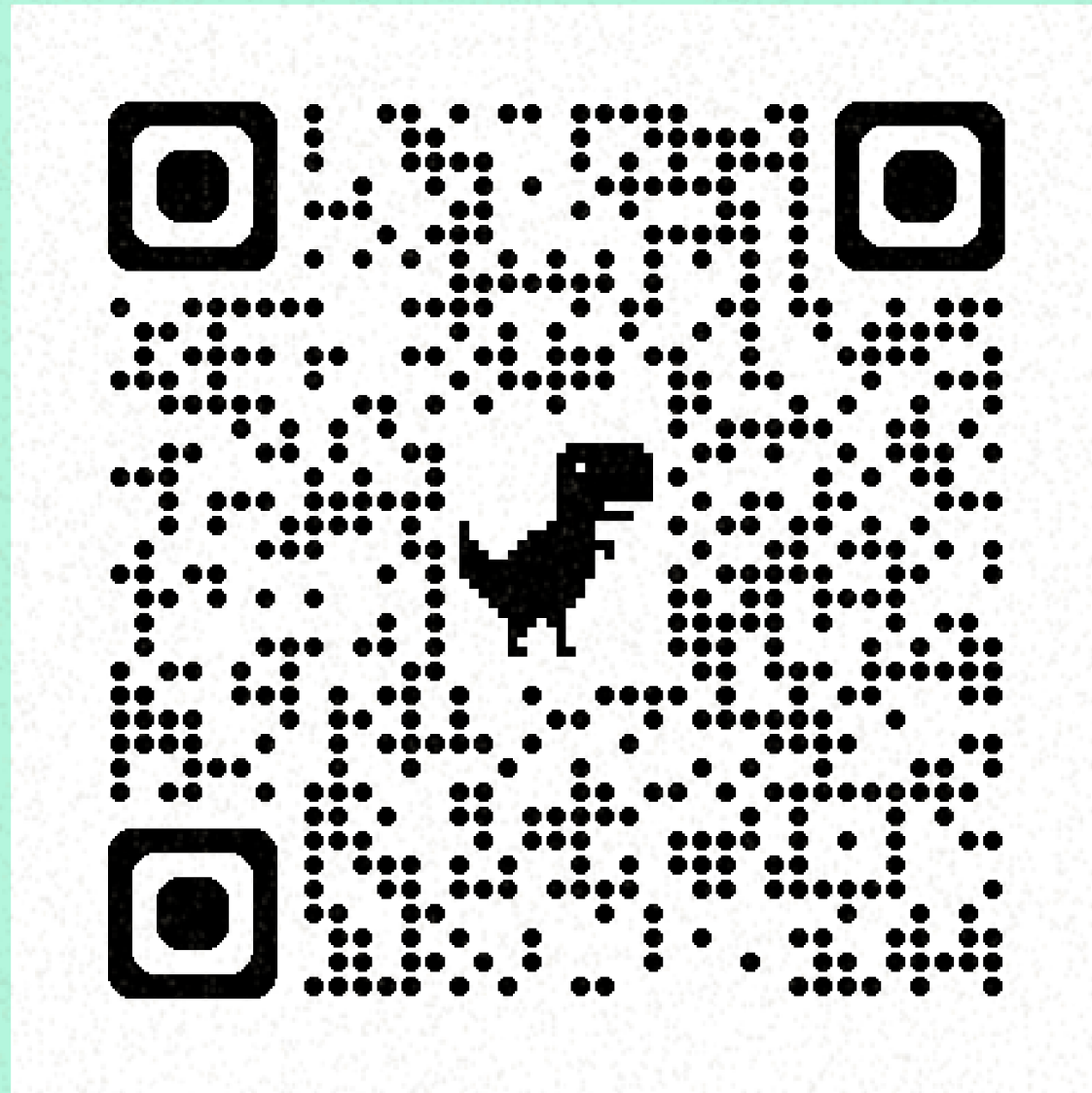
# Or uncomment this if you want to run on real hardware
# You will need to create an IBM developer account and enter your API token
# here, which gives you 10m/month free usage of their hardware
```

0s completed at 5:55 PM

Let's you run quantum circuits in simulators and against real hardware







Link to the slides:

[https://www.julianburr.de/  
wearedevelopers-world-congress-  
2024-slides.pdf](https://www.julianburr.de/wearedevelopers-world-congress-2024-slides.pdf)

<https://www.linkedin.com/in/julianburr/>

<https://twitter.com/jburr90>