

CS251

Great Ideas
in

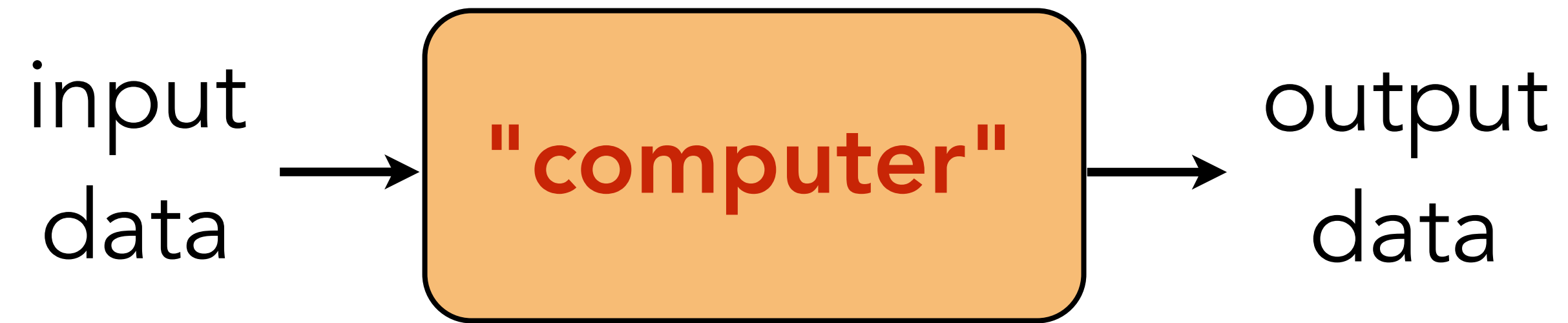
Theoretical

Computer Science



Universality of Computation

This Chapter



What is **computation**?

What is an **algorithm**?

How can we mathematically define them?

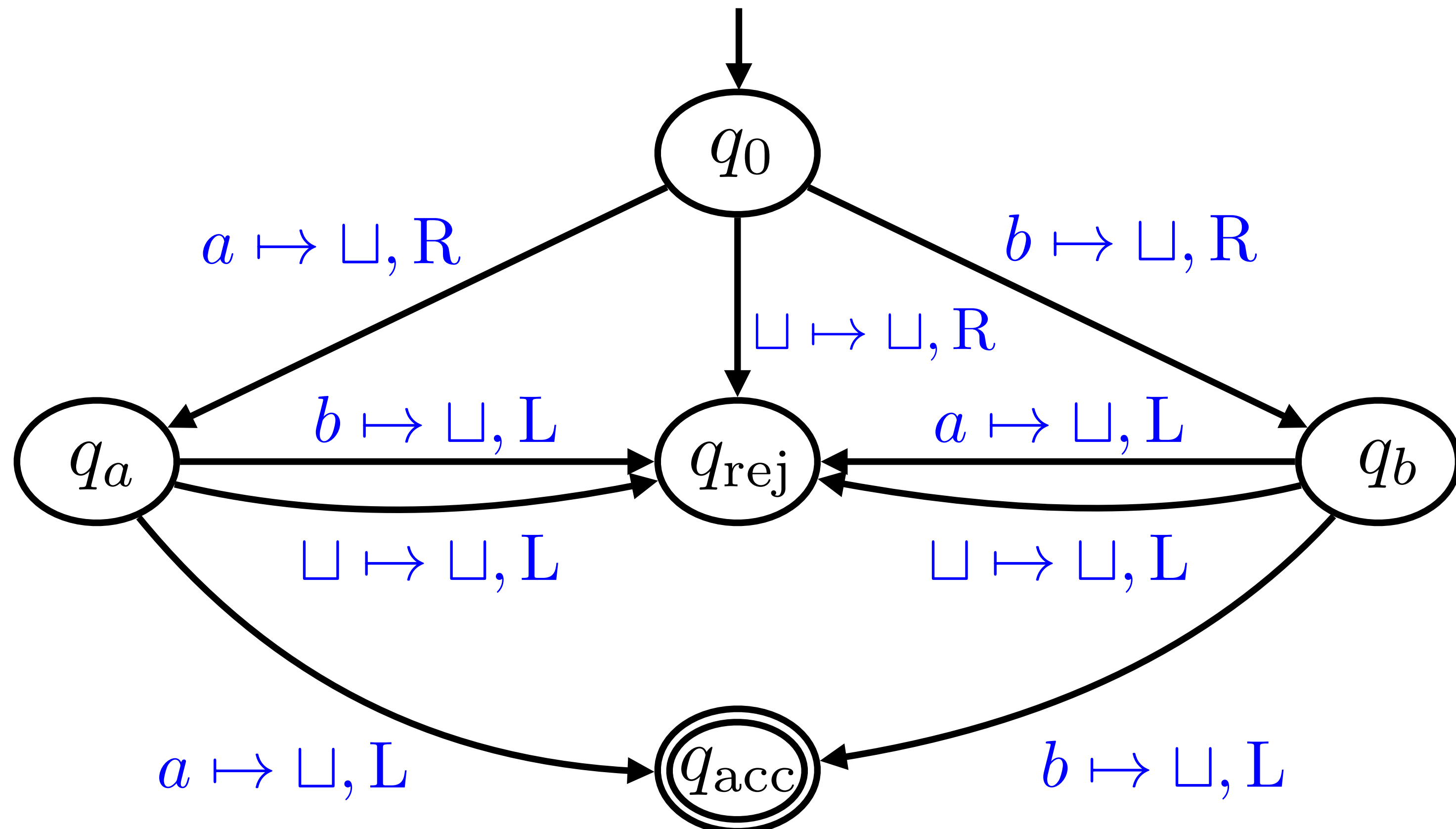
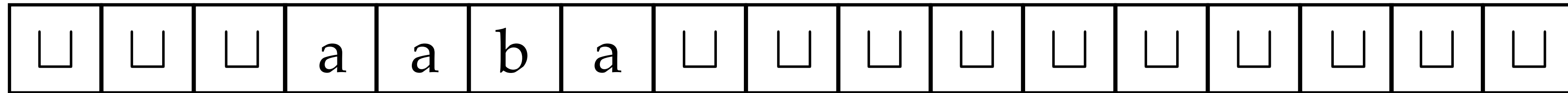
Last Time

A **totally minimal (TM)** programming language such that:

- it can simulate simple bytecode;
(and therefore Python, C, Java, SML, etc...)
- it is simple to define and reason about mathematically rigorously.

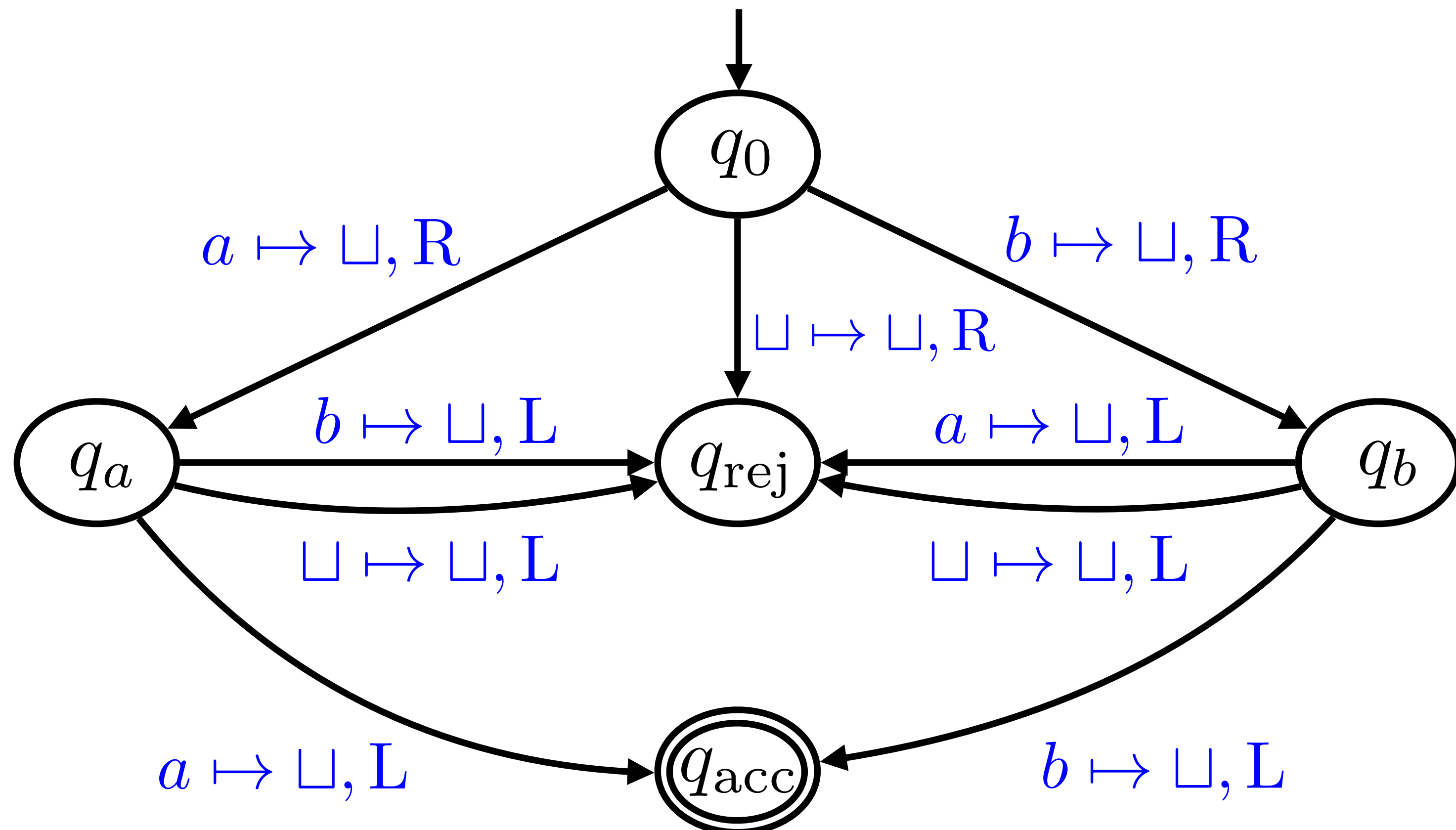
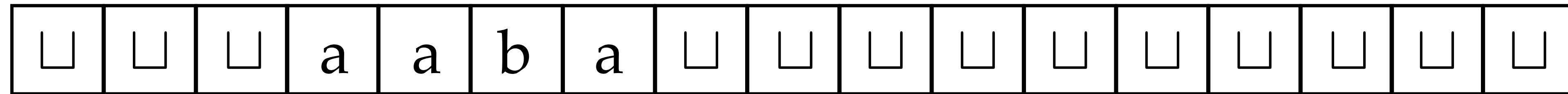
Last Time

TM \approx DFA + memory



Last Time

TM \approx finite set of very simple instructions + tape (memory)



Last Time

Definition: A language $L \subseteq \Sigma^*$ is ***decidable*** if there is a TM M such that for all $w \in \Sigma^*$,

- if $w \in L$, M accepts w ;
- if $w \notin L$, M rejects w .

Some TM subroutines and tricks

- Move right (or left) until first \sqcup encountered
- Shift entire input string one cell to the right
- Convert input from
 $x_1x_2x_3\dots x_n$ to $\sqcup x_1 \sqcup x_2 \sqcup x_3 \dots \sqcup x_n$
- Simulate big Γ with $\{0,1,\sqcup\}$.
- "Mark" cells. If $\Gamma = \{0,1,\sqcup\}$, extend it to
 $\Gamma = \{0,1,0^\bullet,1^\bullet, \sqcup\}$.
- Copy a stretch of tape between two marked cells into another marked section of the tape

Some TM subroutines and tricks

- Simulate a TM with 2 tapes and heads
- Implement basic data structures
- Simulate "random access memory"
- ⋮
- Simulate assembly language



You can prove this rigorously if you want to!

Theorem: Any language that can be computed in Python, Java, C, etc. can be decided by a TM.



You can describe a TM in 3 ways:

Low-level description

State diagram

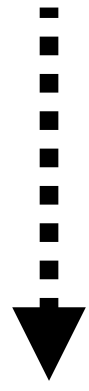
Medium-level description

Describe the movement/behavior of the tape head

High-level description

Code/pseudocode , algorithm(?)

Algorithm written in English



C, Python, SML, etc.



Machine Language



TM

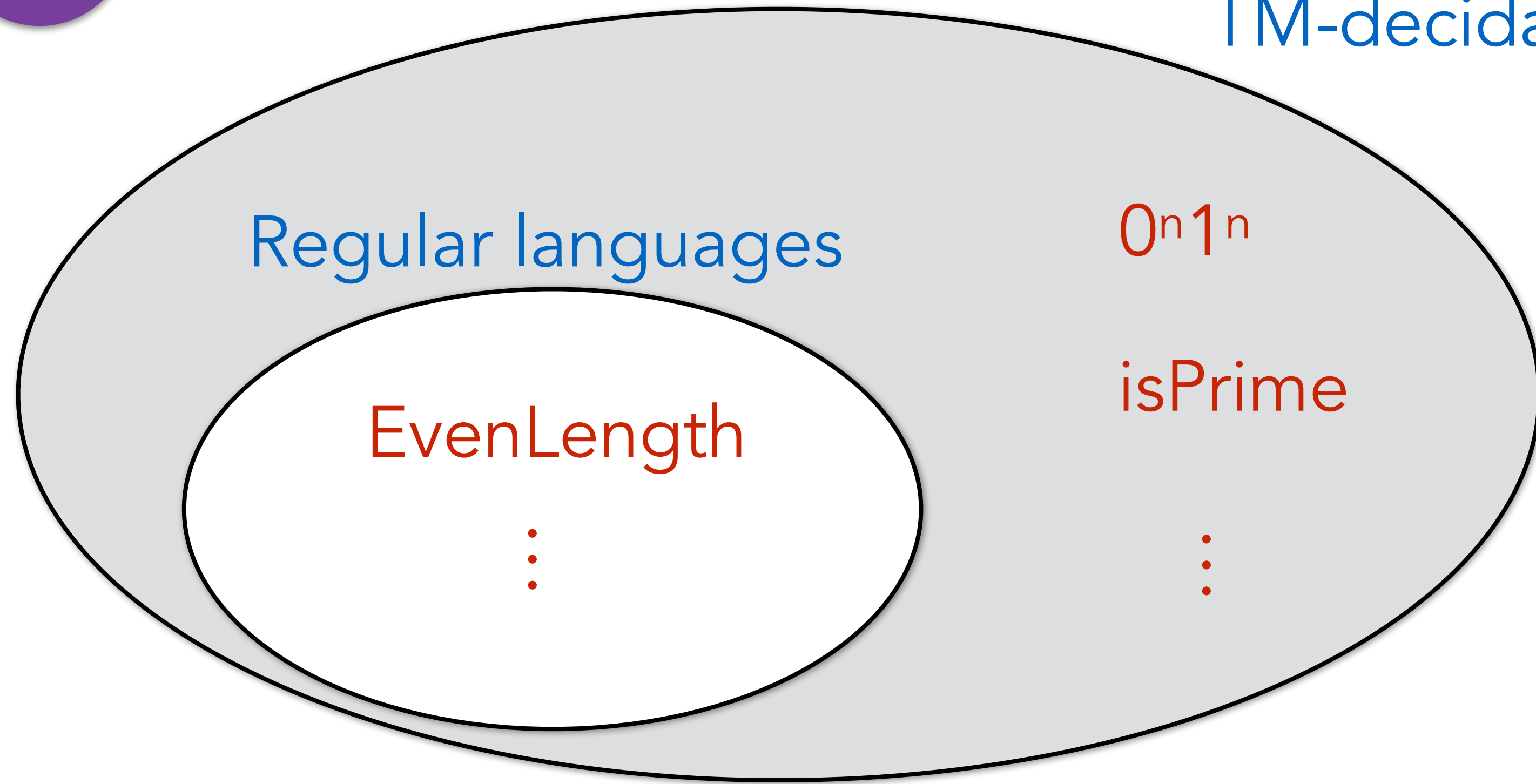


1. Is TM the right definition?

TM-decidable



Solvable with any computing device



2. Are there **undecidable** languages?



3. What was Turing's motivation?

Entscheidungsproblem (1928)

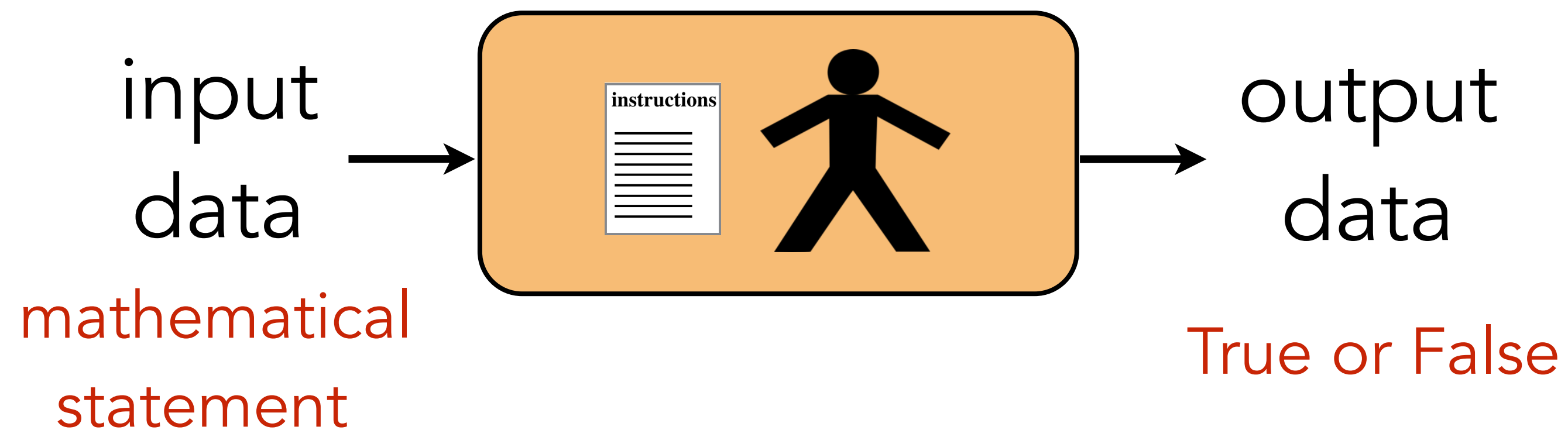


Is there a **finitary procedure** to determine the validity of a given mathematical statement?

e.g. $\neg \exists x, y, z, n \in \mathbb{N} : (n \geq 3) \wedge (x^n + y^n = z^n)$

(Mechanization of mathematics)

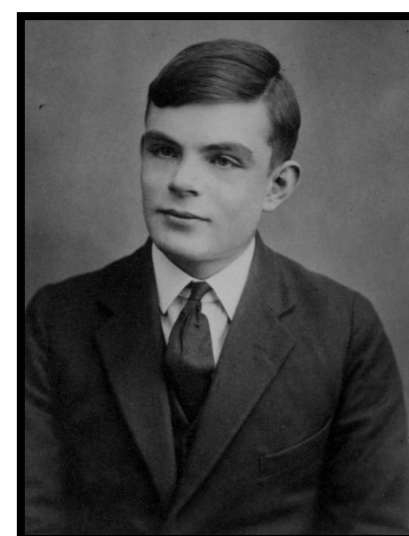
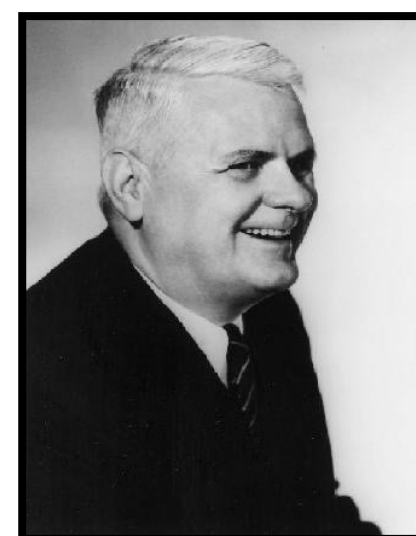
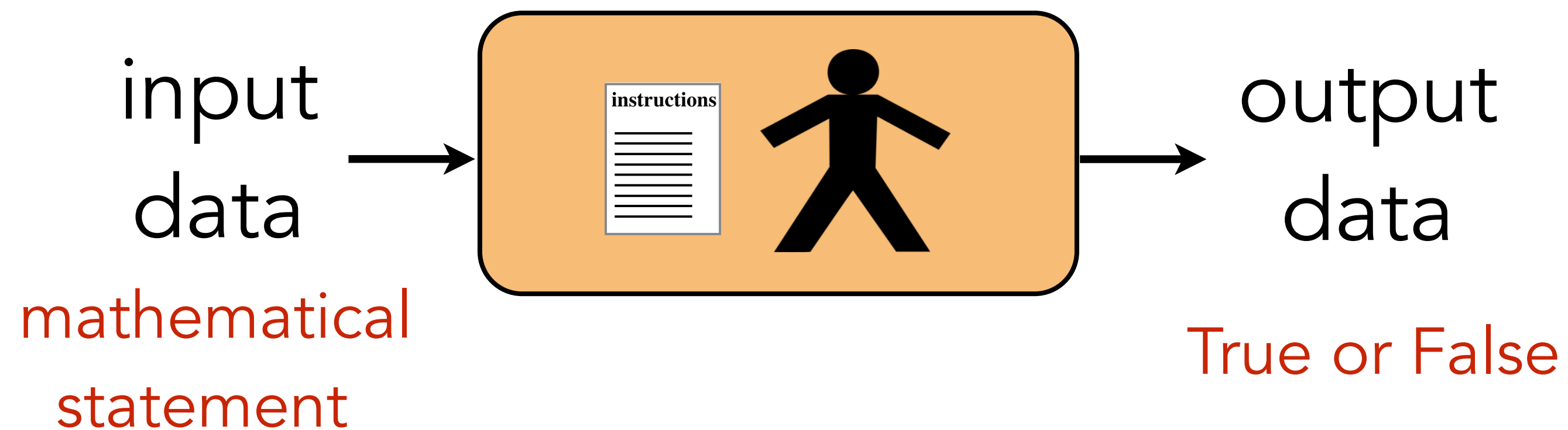
What was Hilbert really asking for?



"computers" in early 20th century

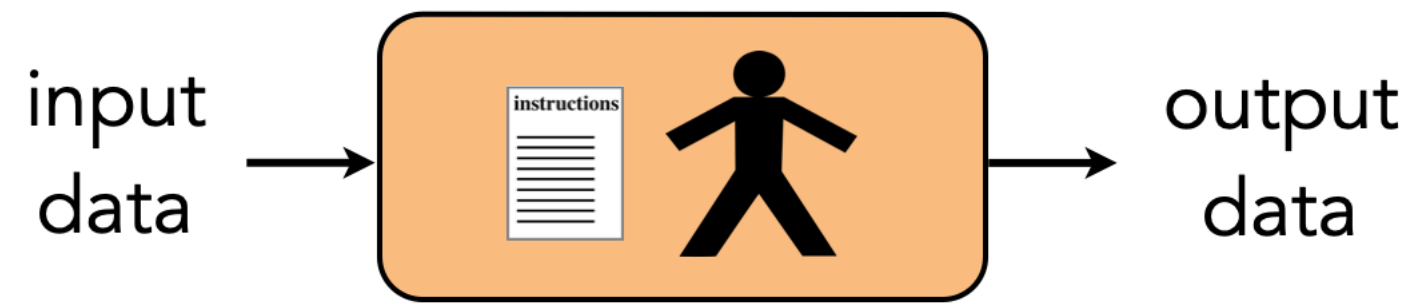


What was Hilbert really asking for?



"Alright, let's define this thing mathematically."

Turing's thinking



- A (human) computer writes symbols on paper.
(can view the paper as a sequence of squares)
- No upper bound on the number of squares.
- Human can reliably distinguish finitely many shapes.
- Human observes one square at a time.
- Human has finitely many mental states.
- Human can change symbol,
can change focus to neighboring square,
based on its state and the symbol it observes.
- Human acts deterministically.
- ...

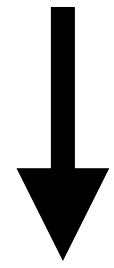
The beauty of the definition:

1. Simplicity

2. Generality

1. Simplicity

any reasonable definition of computation



strong enough to capture computation the way TMs do.

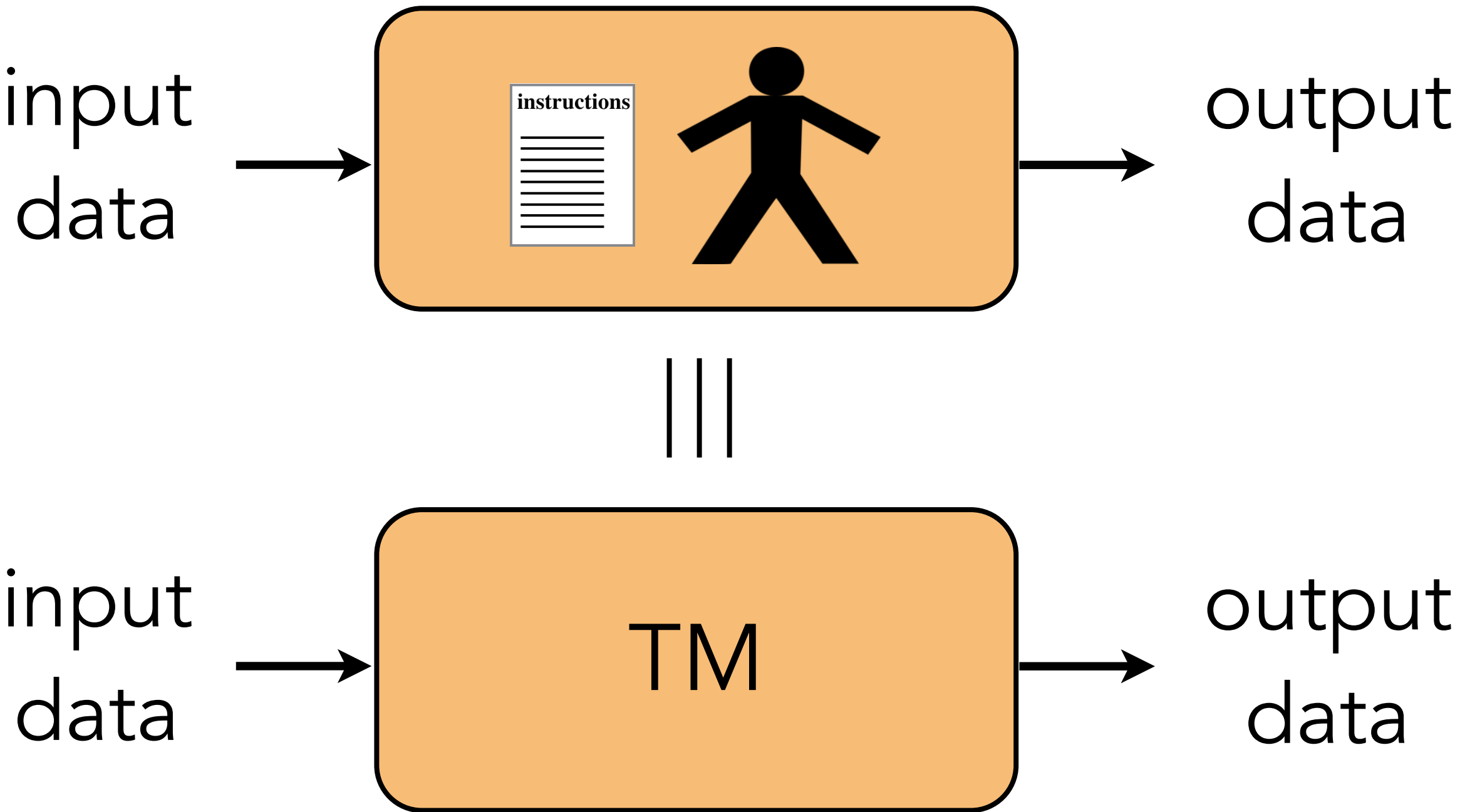
(anyone who attempted to define computation
could accidentally hit a correct definition)

2. Generality

"Clearly" captures what a human does given a set of instructions.

Church-Turing Thesis

The intuitive notion of "computable" is captured by functions computable by a Turing machine.



What else did Turing do in his paper?

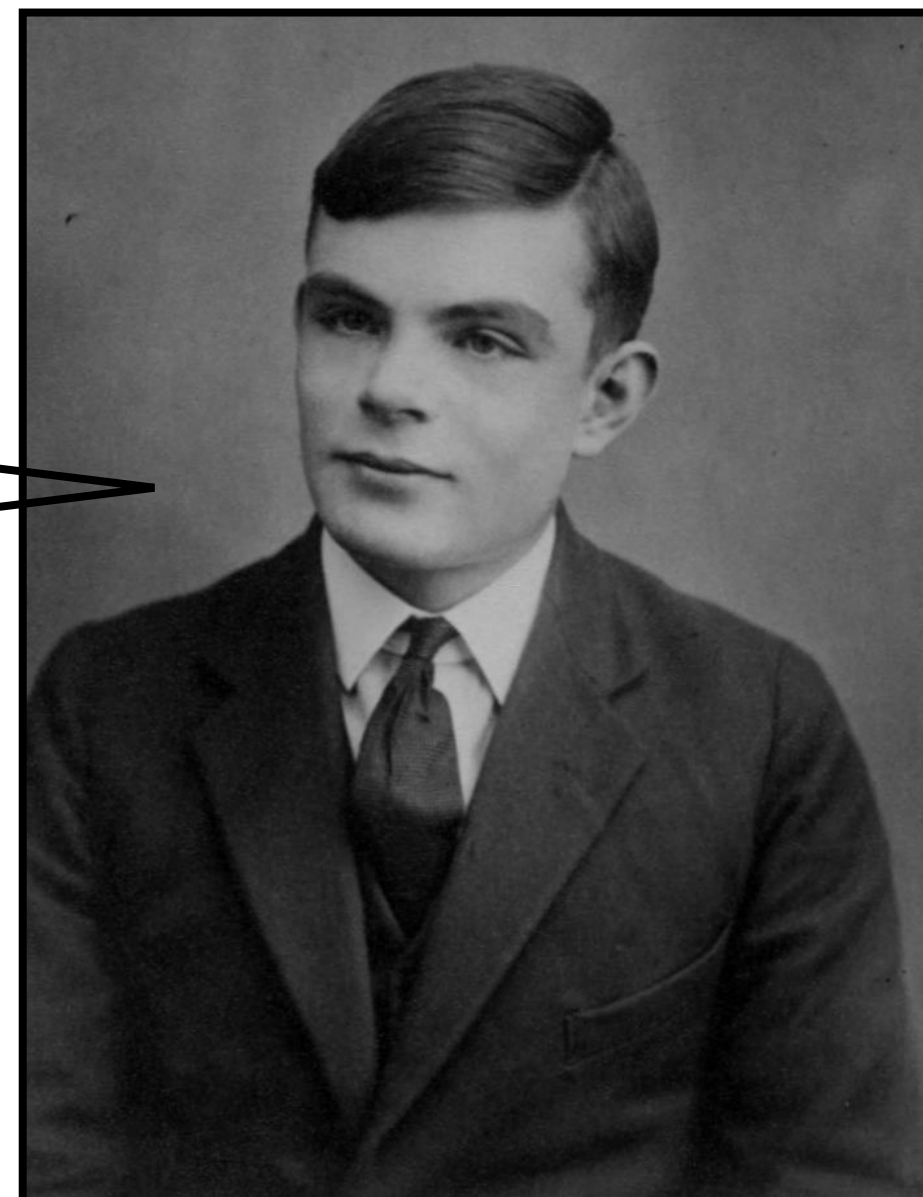
Entscheidungsproblem (1928)

Is there a **finitary procedure** to determine the validity of a given mathematical statement?

e.g. $\neg \exists x, y, z, n \in \mathbb{N} : (n \geq 3) \wedge (x^n + y^n = z^n)$

(Mechanization of mathematics)

No!

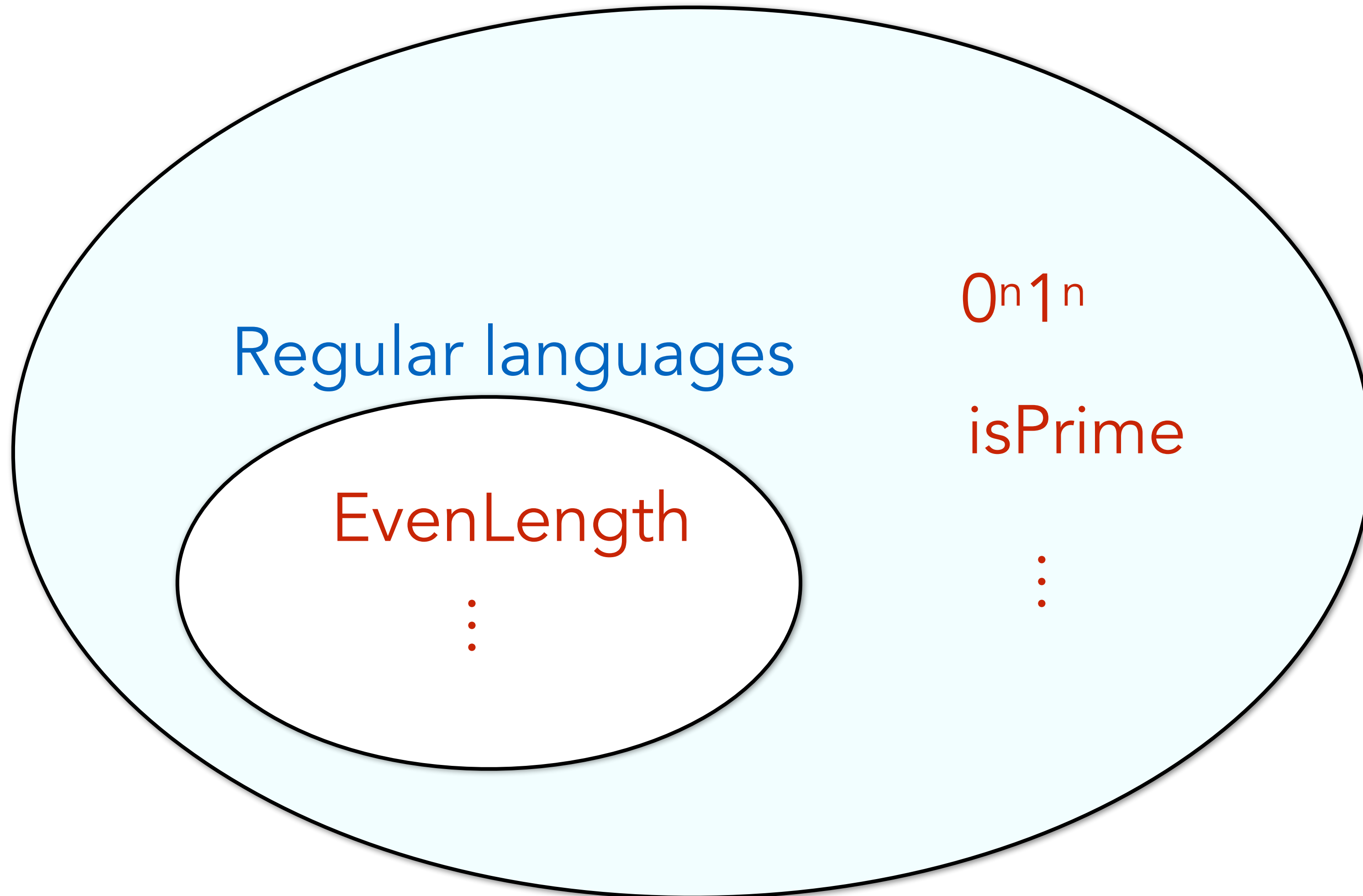


Entscheidungsproblem



Are there others?

TM-decidable



Regular languages

EvenLength

⋮

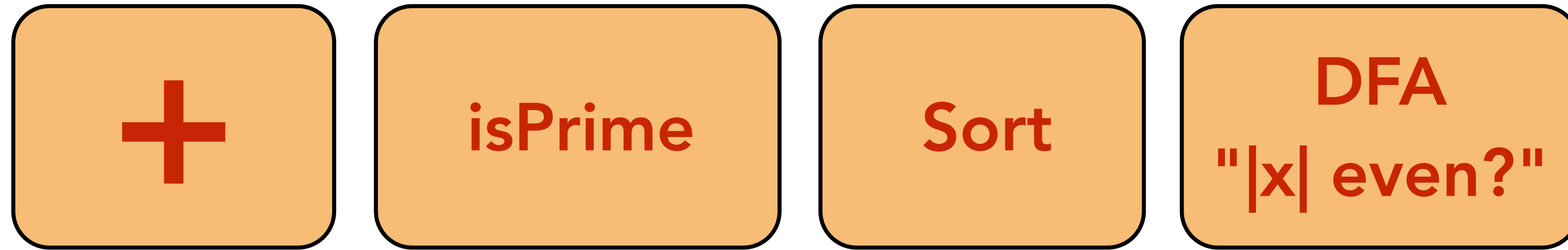
$0^n 1^n$

isPrime

⋮

What else did Turing do in his paper?

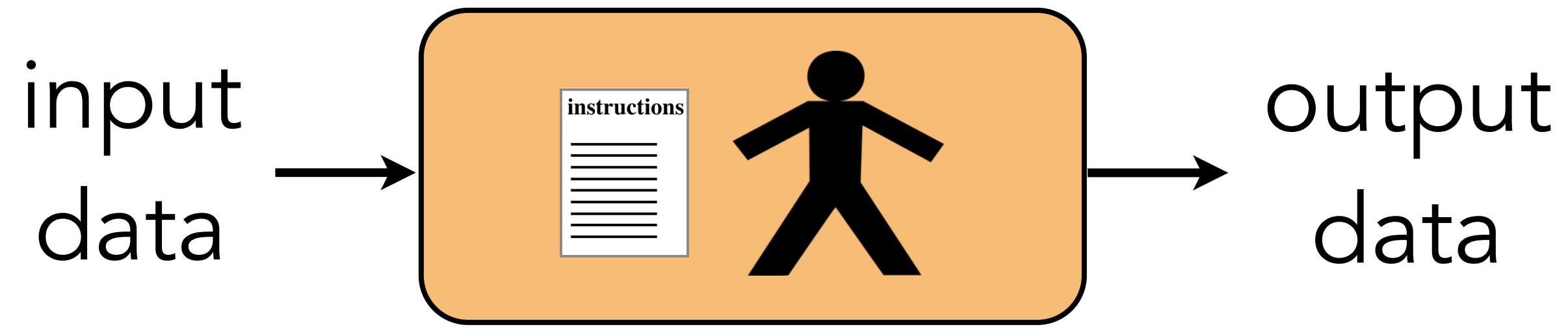
Universal Machine (one machine to rule them all)



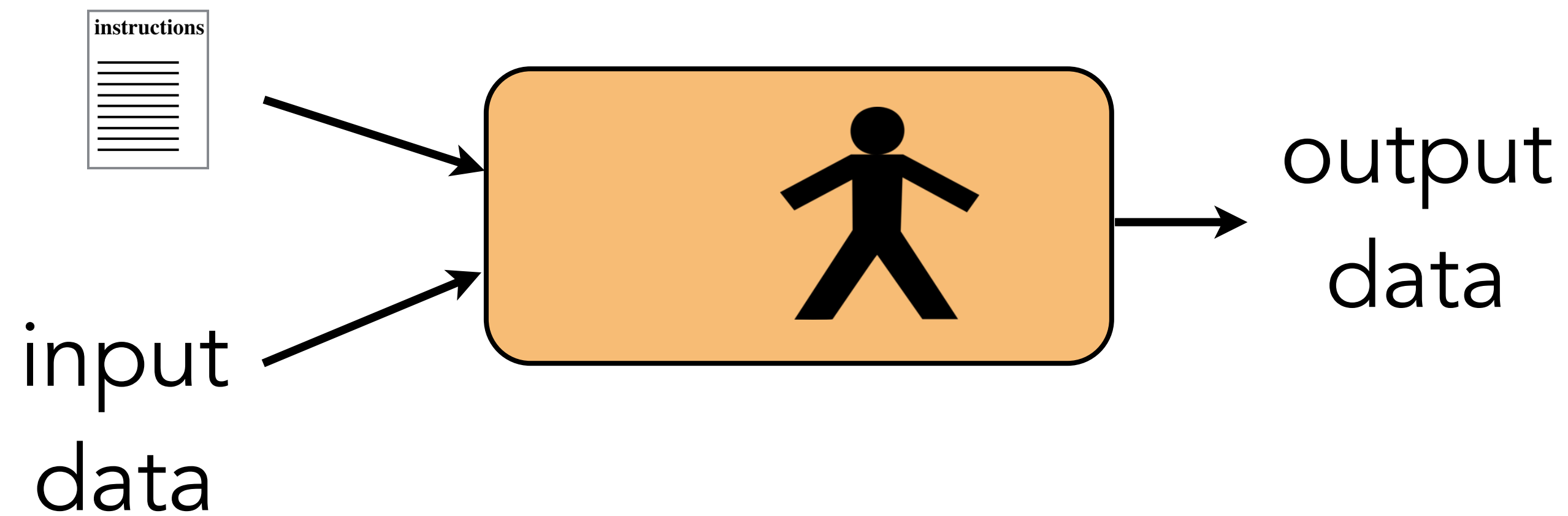
Do we really need a separate
(physical) machine for each task?

What else did Turing do in his paper?

Universal Machine (one machine to rule them all)

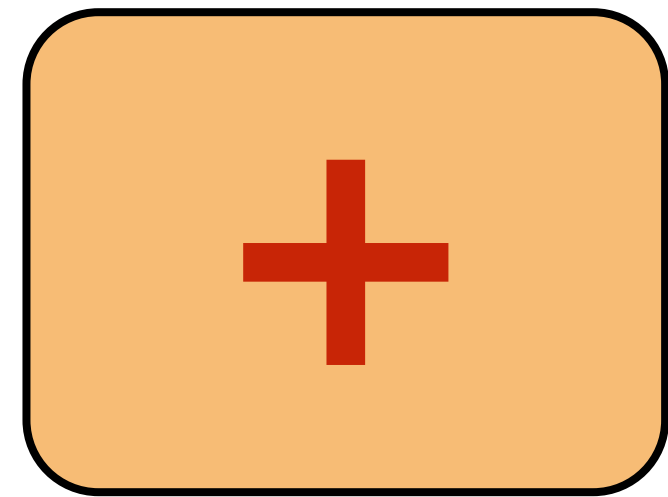


A **human** is a universal machine



What else did Turing do in his paper?

Universal Machine (one machine to rule them all)



All can be encoded!
(e.g. think source code)

What else did Turing do in his paper?

Universal Machine (one machine to rule them all)

```
def foo(input):
```

```
    i = 0
```

```
    STATE 0:
```

```
        letter = input[i];
```

```
        switch(letter):
```

```
            case 'a': input[i] = ' '; i++; go to STATE a;
```

```
            case 'b': input[i] = ' '; i++; go to STATE b;
```

```
            case ' ': input[i] = ' '; i++; go to STATE rej;
```

```
    STATE a:
```

```
        letter = input[i];
```

```
        switch(letter):
```

```
            case 'a': input[i] = ' '; i--; go to STATE acc;
```

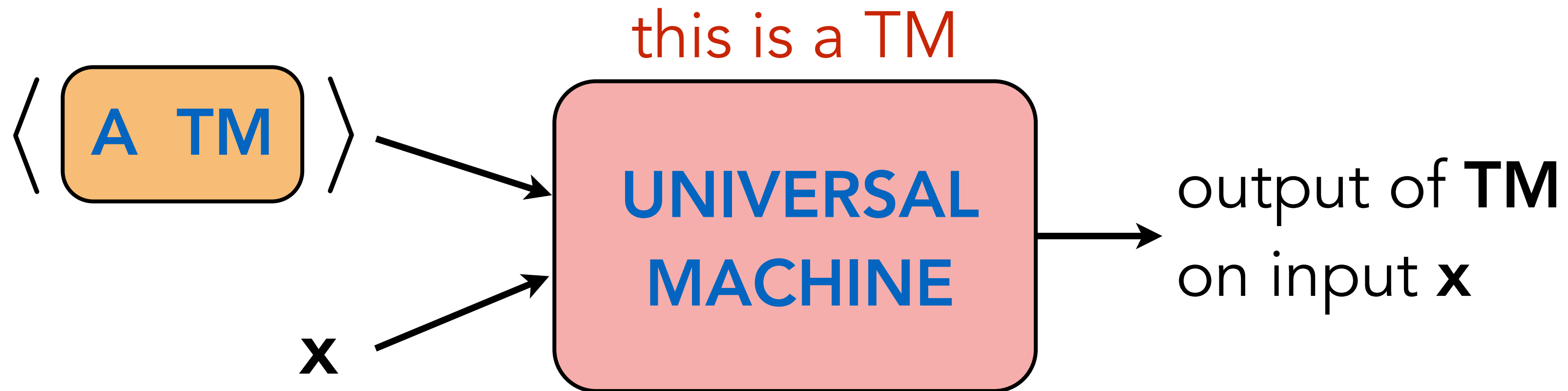
```
            case 'b': input[i] = ' '; i--; go to STATE rej;
```

```
            case ' ': input[i] = ' '; i--; go to STATE rej;
```

$\langle M \rangle =$

What else did Turing do in his paper?

Universal Machine (one machine to rule them all)



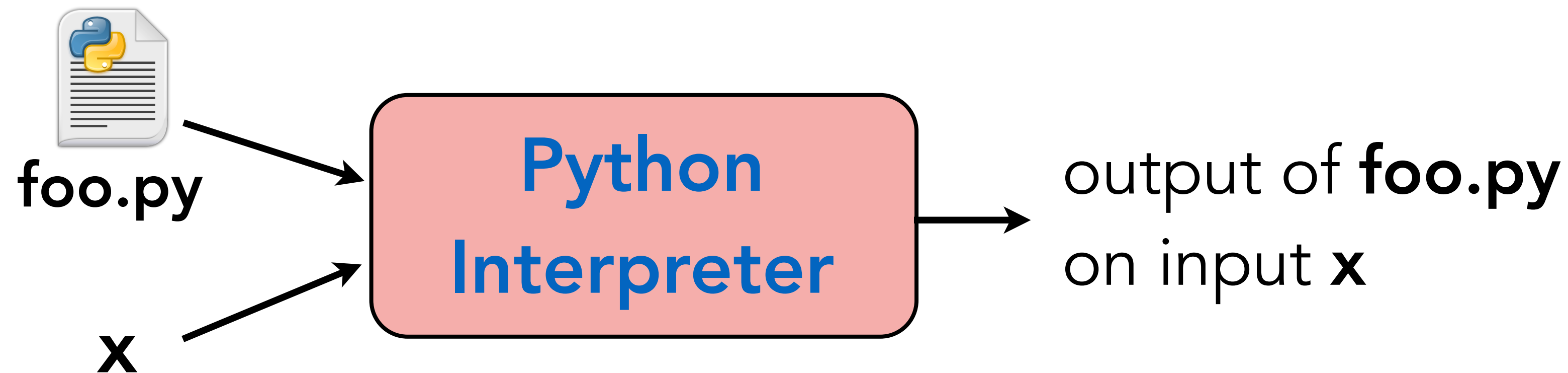
Could you write a Python function that does this?

Yes!

What else did Turing do in his paper?

Universal Machine (one machine to rule them all)

This is exactly what an **interpreter** does.





Code is data!



Code is data!

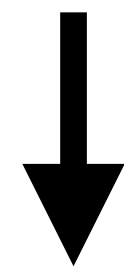
The **positive** side

Universal TM

The **negative** side

Self-referencing

(can feed a machine
its own code as input)



Undecidability

ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO
THE ENTSCHIEDUNGSPROBLEM

By A. M. TURING.

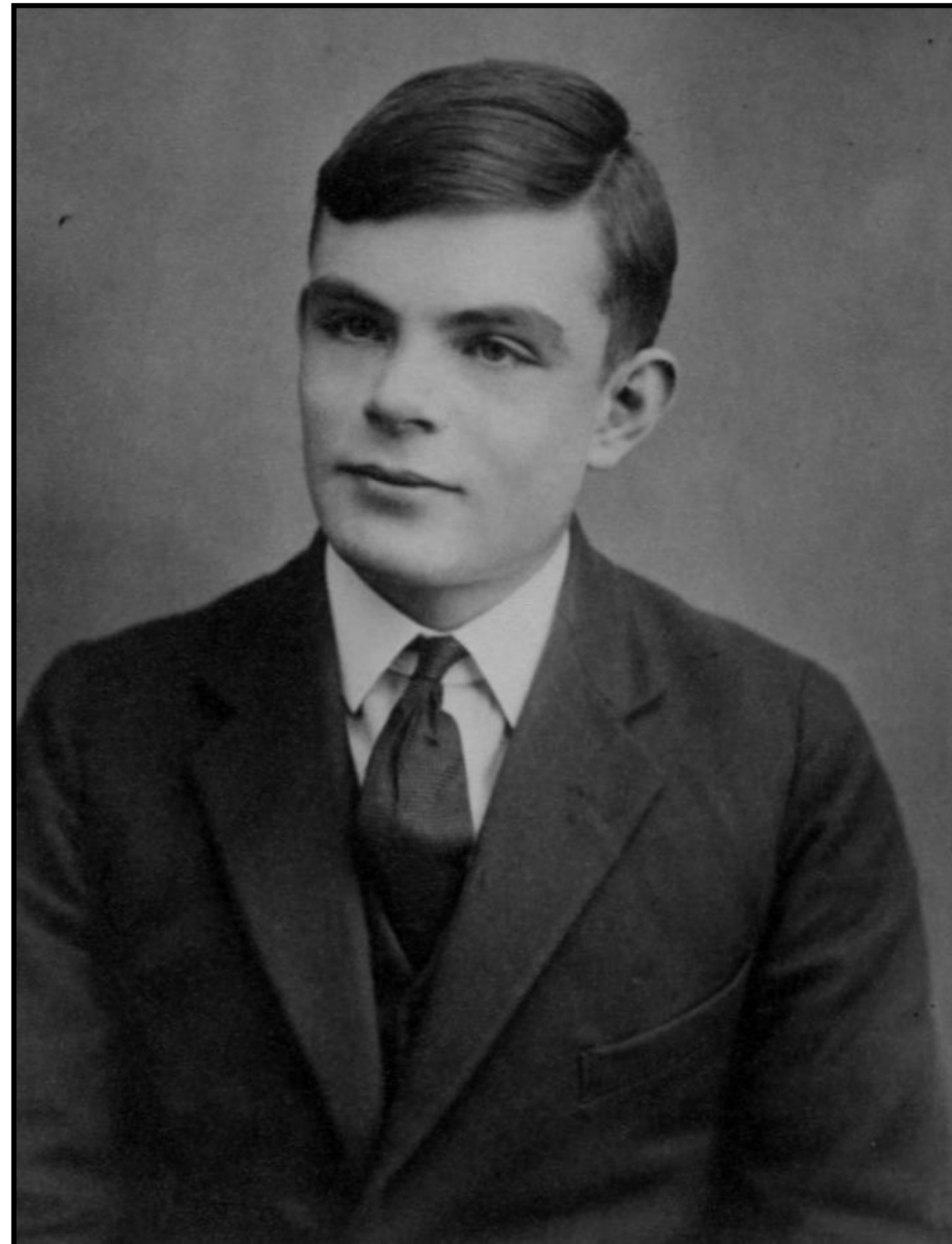
[Received 28 May, 1936.—Read 12 November, 1936.]

The “computable” numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means. Although the subject of this paper is ostensibly the computable *numbers*, it is almost equally easy to define and investigate computable functions of an integral variable or a real or computable variable, computable predicates, and so forth. The fundamental problems involved are, however, the same in each case, and I have chosen the computable numbers for explicit treatment as involving the least cumbersome technique. I hope shortly to give an account of the relations of the computable numbers, functions, and so forth to one another. This will include a development of the theory of functions of a real variable expressed in terms of computable numbers. According to my definition, a number is computable if its decimal can be written down by a machine.

In §§ 9, 10 I give some arguments with the intention of showing that the computable numbers include all numbers which could naturally be regarded as computable. In particular, I show that certain large classes of numbers are computable. They include, for instance, the real parts of all algebraic numbers, the real parts of the zeros of the Bessel functions, the numbers π , e , etc. The computable numbers do not, however, include all definable numbers, and an example is given of a definable number which is not computable.

Although the class of computable numbers is so great, and in many ways similar to the class of real numbers, it is nevertheless enumerable. In § 8 I examine certain arguments which would seem to prove the contrary. By the correct application of one of these arguments, conclusions are reached which are superficially similar to those of Gödel†. These results

† Gödel, “Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme, I”, *Monatshefte Math. Phys.*, 38 (1931), 173–198.

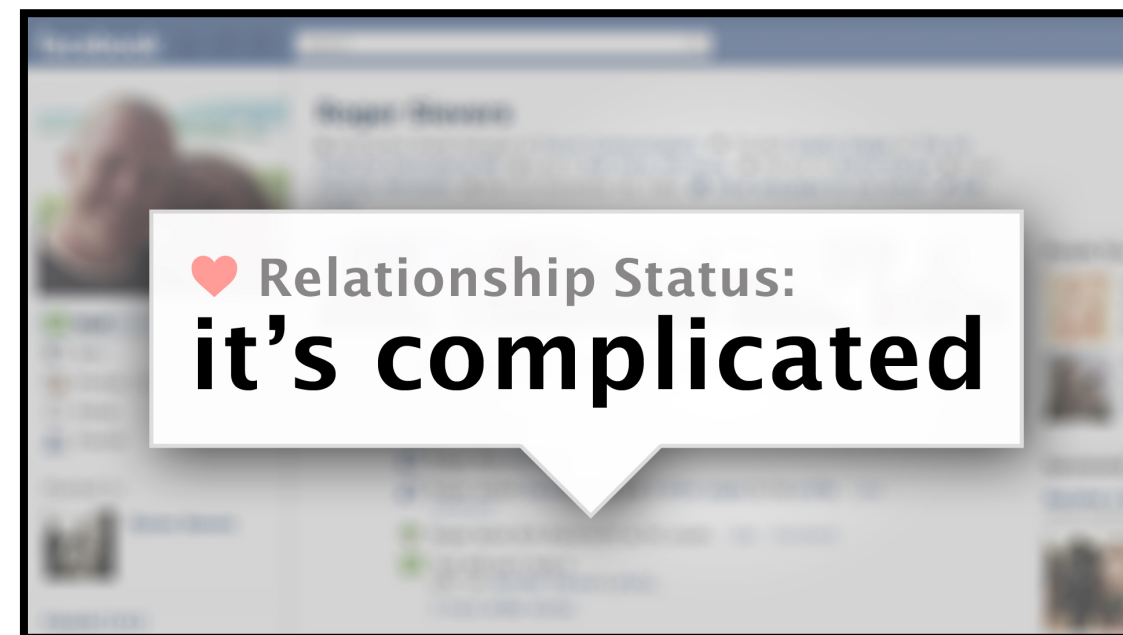


1936

Computation and Physics

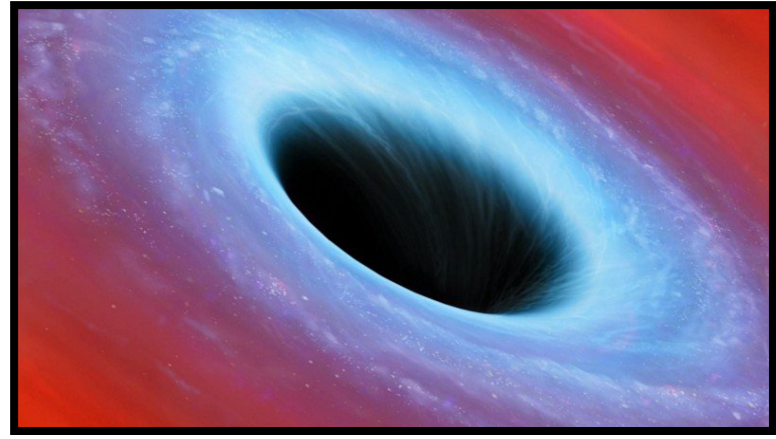
Perhaps Turing and others weren't ambitious enough!

computation

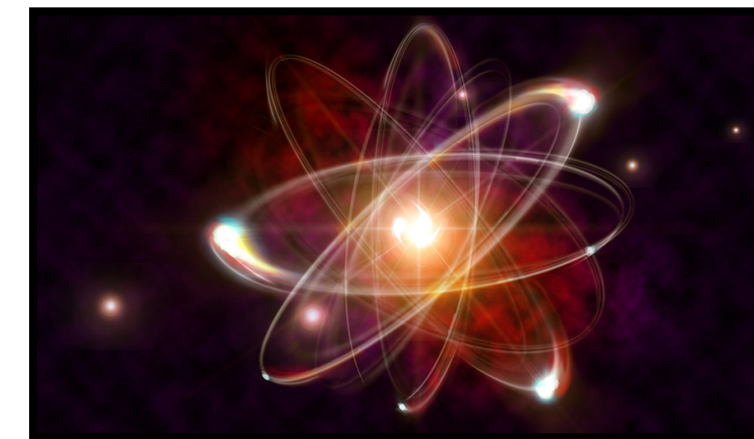


universe and the
laws of physics

Physics → **Computation**



Solvable by any physical process



||| ???

Solvable by a TM

Church-Turing Thesis 2.0

(Physical) Church-Turing Thesis:

Any computational problem that can be solved by any physical process, can be solved by a (prob.) TM.



This is not a theorem!

Extended Physical Church-Turing Thesis

What can be computed **efficiently** in this universe, by any physical process or device, can be computed **efficiently** by a TM.

Church-Turing Thesis 2.0

(Physical) Church-Turing Thesis:

Any computational problem that can be solved by any physical process, can be solved by a (prob.) TM.

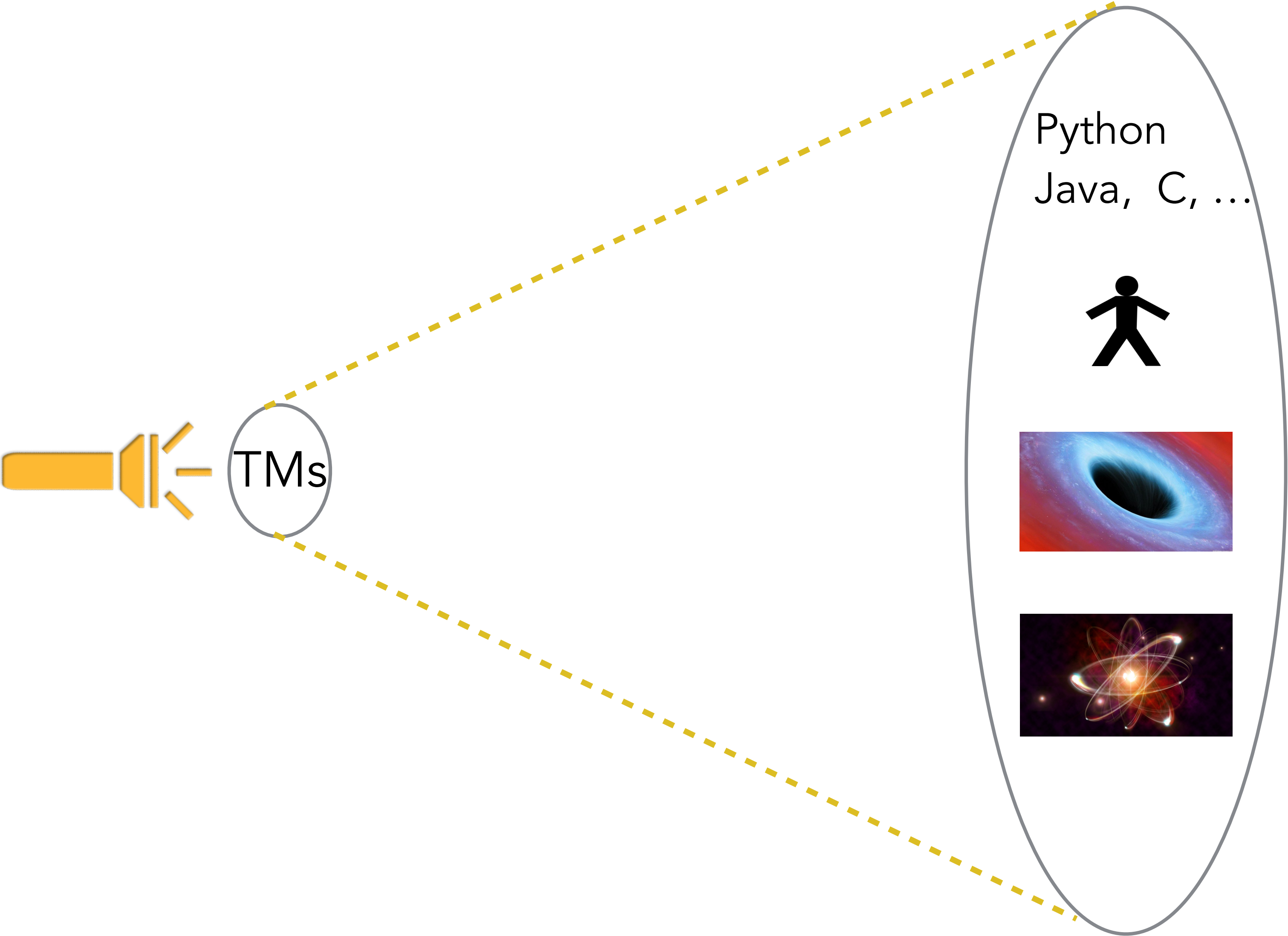


This is not a theorem!

Extended Physical Church-Turing Thesis

What can be computed **efficiently** in this universe, by any physical process or device, can be computed **efficiently** by a **QM**.

Grand unification/simplification of computation!!



All types of computation

Implications

1. Studying the power and limits of TMs

→ Studying the power and limits of our universe

(Can you come up with sensible laws of physics that would allow it to compute any problem?)

2. Computation in its full generality is everywhere.

Even in extremely simple systems!

(What is the simplest universe you can create that has the same computational capacity of our universe?)

3. The universe may be a simulation.

(a philosophical musing)

Computation → **Physics**

Black Holes at Exp-time

Authors: [Leonard Susskind](#)

Abstract: Classical GR governs the evolution of black holes for a long time, but at some exponentially large time it must break down. The breakdown, and what comes after it, is not well understood. In this paper I'll discuss the problem using concepts drawn from complexity geometry. In particular the geometric concept of cut locus plays a key role.

Submitted 1 June, 2020; **originally announced** June 2020.

Comments: 14 figures

Complexity and Momentum

Authors: [Leonard Susskind](#), [Ying Zhao](#)

Abstract: Previous work has explored the connections between three concepts -- operator size, complexity, and the bulk radial momentum of an infalling object -- in the context of JT gravity and the SYK model. In this paper we investigate the higher dimensional generalizations of these connections. We use a toy model to study the growth of an operator when perturbing the vacuum of a CFT. From circuit analysi... [▽ More](#)

Submitted 4 June, 2020; **originally announced** June 2020.

Comments: 14 pages, 3 figures

The Complexity Geometry of a Single Qubit

Authors: [Adam R. Brown](#), [Leonard Susskind](#)

Abstract: The computational complexity of a quantum state quantifies how hard it is to make. 'Complexity geometry', first proposed by Nielsen, is an approach to defining computational complexity using the tools of differential geometry. Here we demonstrate many of the attractive features of complexity geometry using the example of a single qubit, which turns out to be rich enough to be illustrative but simp... [▽ More](#)

Submitted 29 March, 2019; **originally announced** March 2019.

Comments: 40 pages, 7 figures, 1 qubit

Journal ref: Phys. Rev. D 100, 046020 (2019)

Black Holes and Complexity Classes

Authors: [Leonard Susskind](#)

Abstract: It is not known what the limitations are on using quantum computation to speed up classical computation. An example would be the power to speed up PSPACE-complete computations. It is also not known what the limitations are on the duration of time over which classical general relativity can describe the interior geometry of black holes. What is known is that these two questions are closely connecte... [▽ More](#)

Submitted 6 February, 2018; **originally announced** February 2018.

Comments: 13 pages, 3 figures

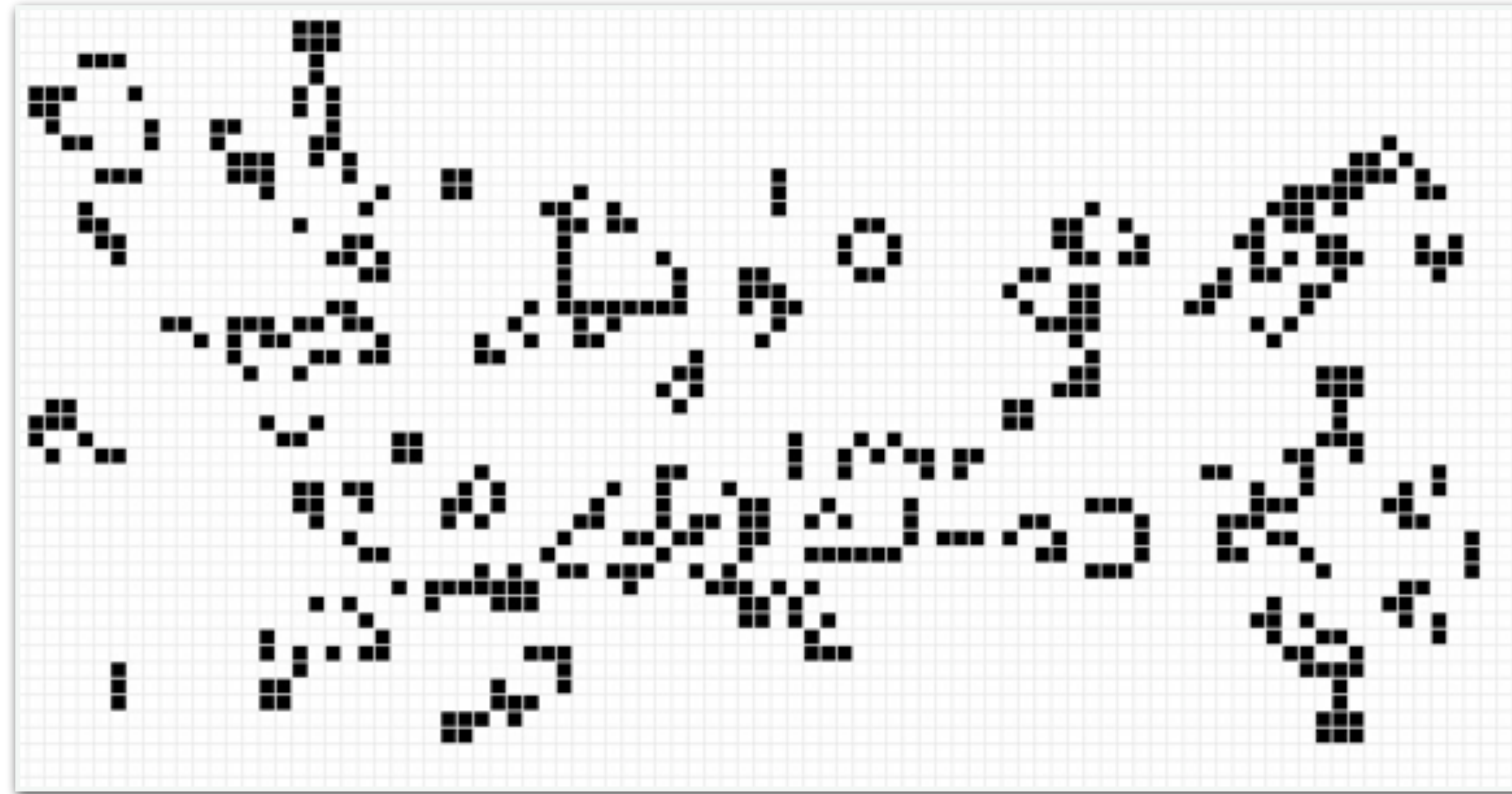
A Final Musing

What is the simplest universe you can create that has the same computational capacity of our universe?

Conway's Game of Life

Imagine an infinite
2D grid.

Each cell can be
dead or alive.



Laws of physics

Loneliness: live cell with fewer than 2 neighbors dies.

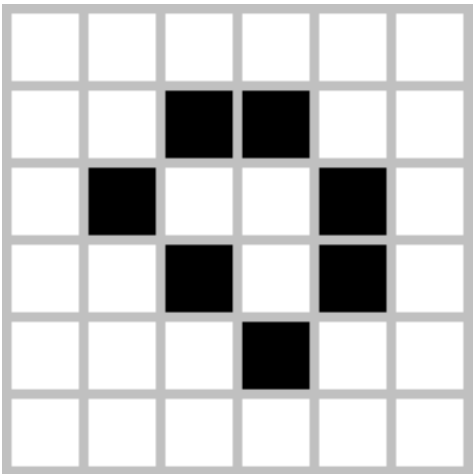
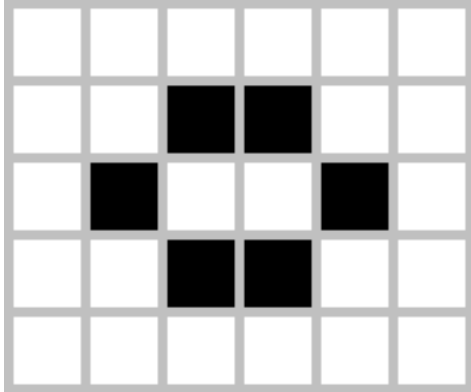
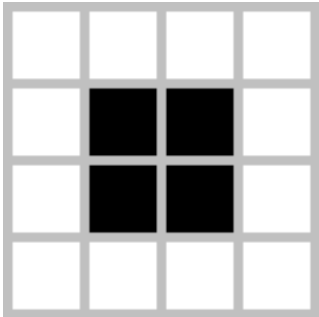
Overcrowding: live cell with more than 3 neighbors dies.

Procreation: dead cell with exactly 3 neighbors gets born.

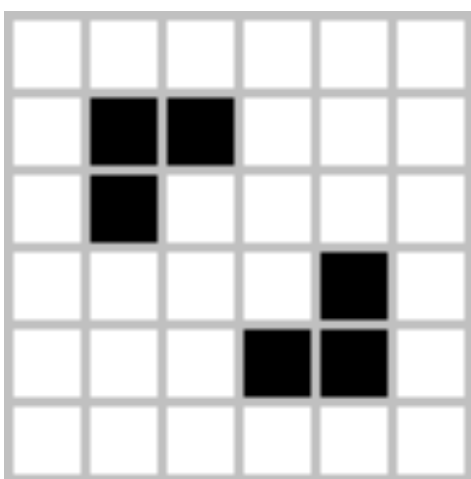
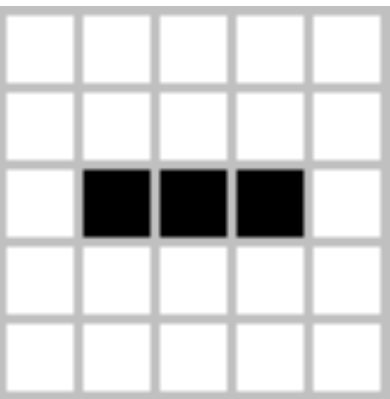
Conway's Game of Life

Some Patterns

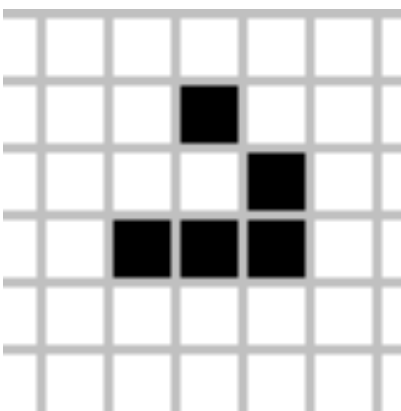
Stable



Periodic






Moving



Conway's Game of Life

Questions

-  Can a TM simulate any instance of Game of Life?
-  Can Game of Life simulate any TM?
-  Can Game of Life simulate Game of Life?

NEXT WEEK

Turing's Legacy Continues



Undecidability