Theoretical Computer Science

Automata Theory and Models of Computation

Lecture 9 - Manuel Mazzara

Who is him?

Homer

Real character vs. mythological (850 BCE?)

Iliad and *Odyssey*

Believed to be the first and greatest of the epic poets

Author of the first known literature of Europe

Why should we mention him?

Automata Theory

It regards:

- **The study of abstract mathematical machines (automata)**
- **The computational problems that can be solved by them**

Automaton (singular), Automata (plural)

Latinization of the Greek αὐτόματον (automaton): *self-moving*

- something is doing something by itself

The word automaton was first used by *Homer*

- describing automatic door opening
- automatic movement of wheeled tripods
- moving statues…

Why studying Automata Theory?

An automaton is a ***finite*** representation of a formal language that may be ***infinite***

Theoretical models for computing machines to be used for proofs about computability

Model of computation

- A **mathematical model of computation** describes how
  - a set of **outputs are computed** given a set of inputs
  - units of computations, memories, and communications are **organized**

- Theory: **automata theory**, **computability** and **computational complexity**

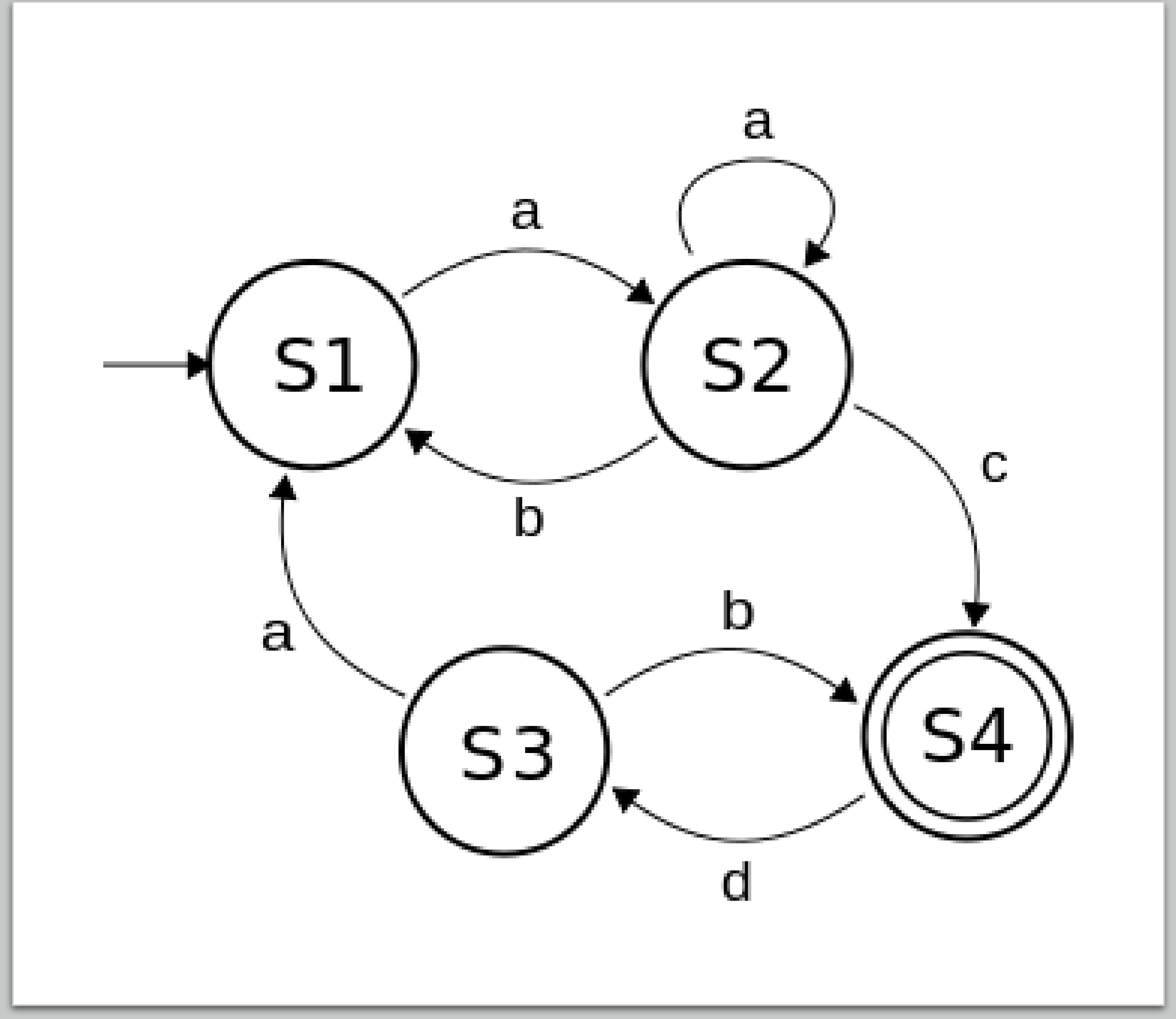
- Practice: **system specification**, compiler construction…

Different Models of computation

- Sequential
  - **Finite state automata**
  - **Pushdown automata**
  - Turing Machine
- Functional
  - Lambda calculus
- Concurrent
  - Petri nets
  - …
- **This list is not exhaustive**

Example: FSA

- Simple **model of computation**
- **Limited expressiveness**
  - Fixed memory
- Suitable to “brute force” analysis
  - **Model Checking**



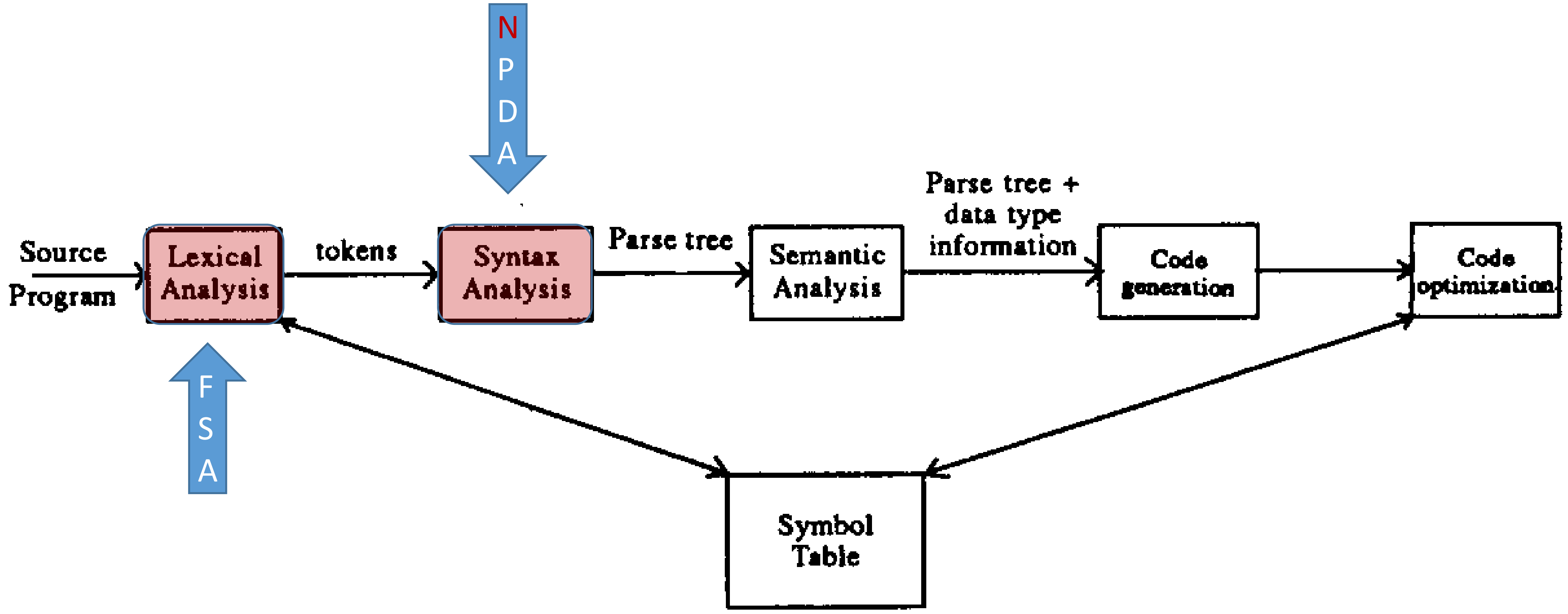
Applications

- FSA have several applications

- **Moore/Mealy** machines model computer circuits or electronic devices
  - Mealy machines are **finite state transducers**
  - The have both input and output tape

- Finite State Automata have a major application in compilers construction
  - **Lexical Analysis**

General Structure of a Compiler

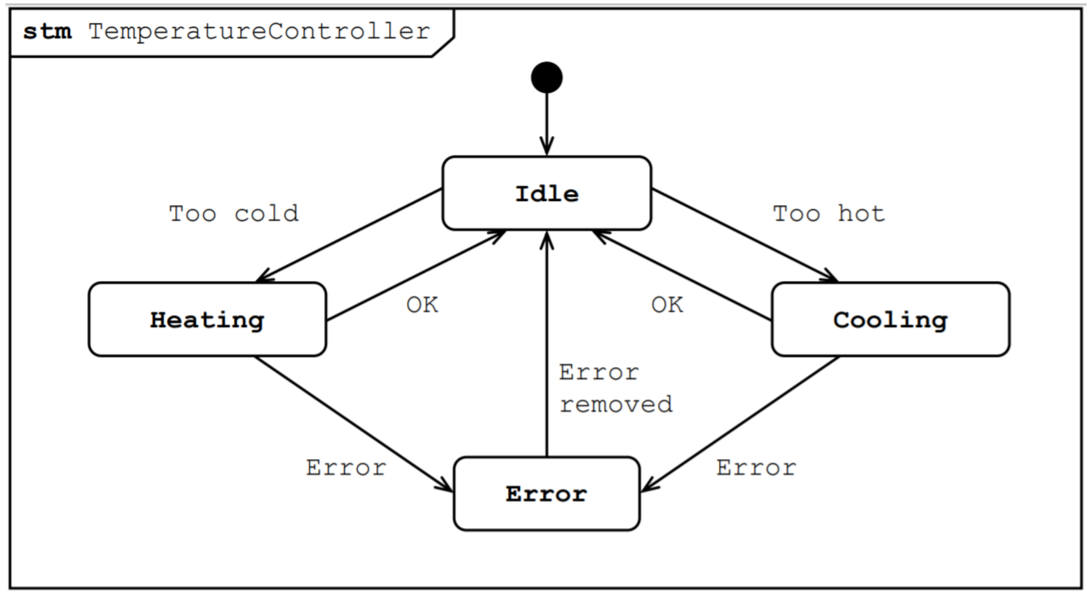


<u>**You will study this in Compilers course**</u>

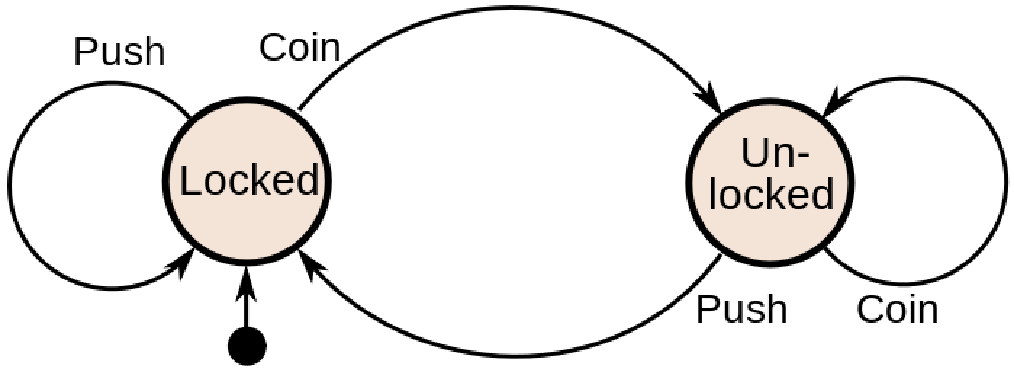
Finite State Automata can be used for analysis and design of systems

UML state machines

- The **Unified Modeling Language** has a notation for describing state machines
- This notation can be used for **analysis and design** of part of a system



Coin Operated Toilet Turnstiles

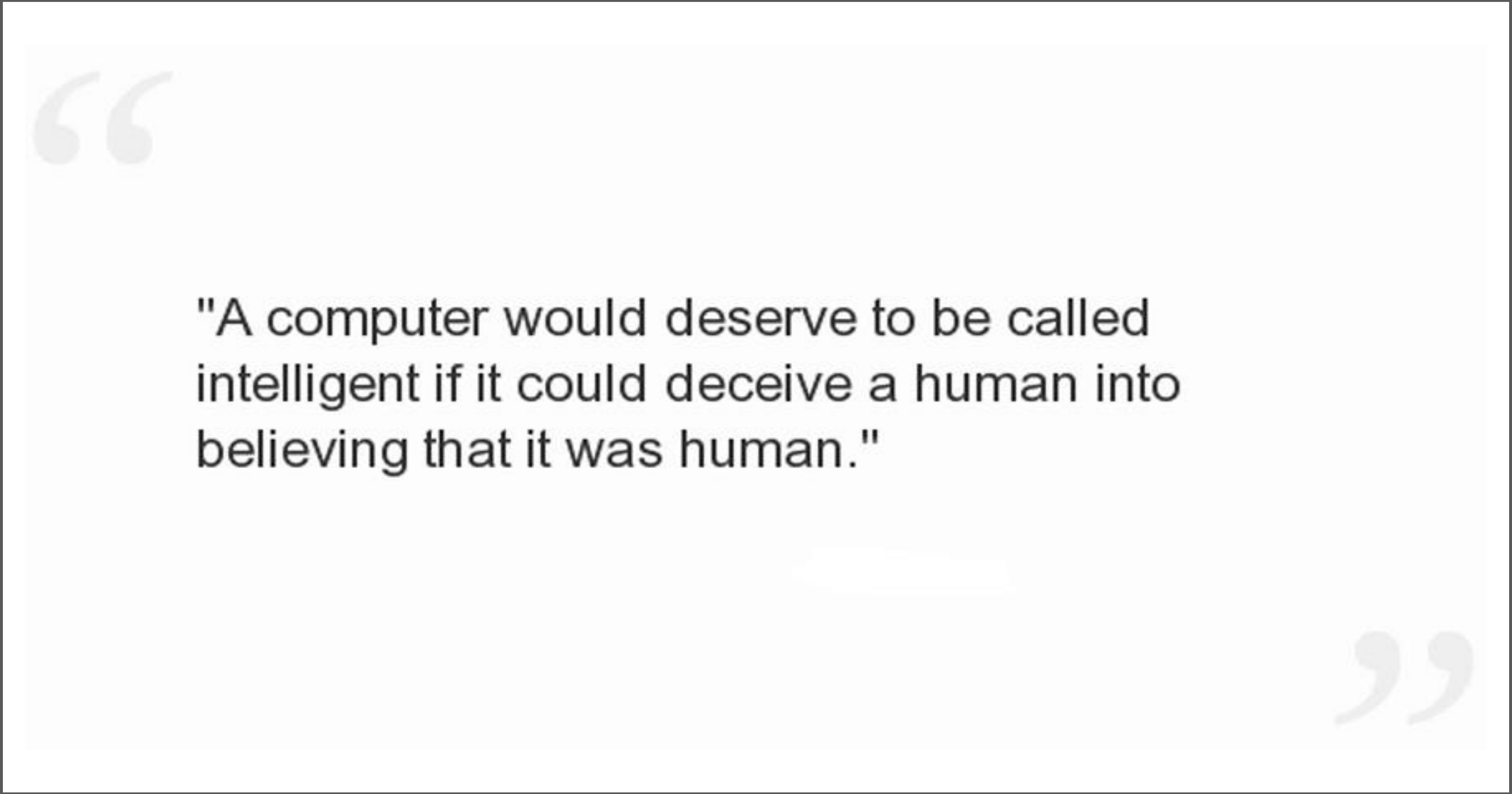


Would you be able to use FSA to design Innopolis University Turnstile system?

Is FSA all you need?

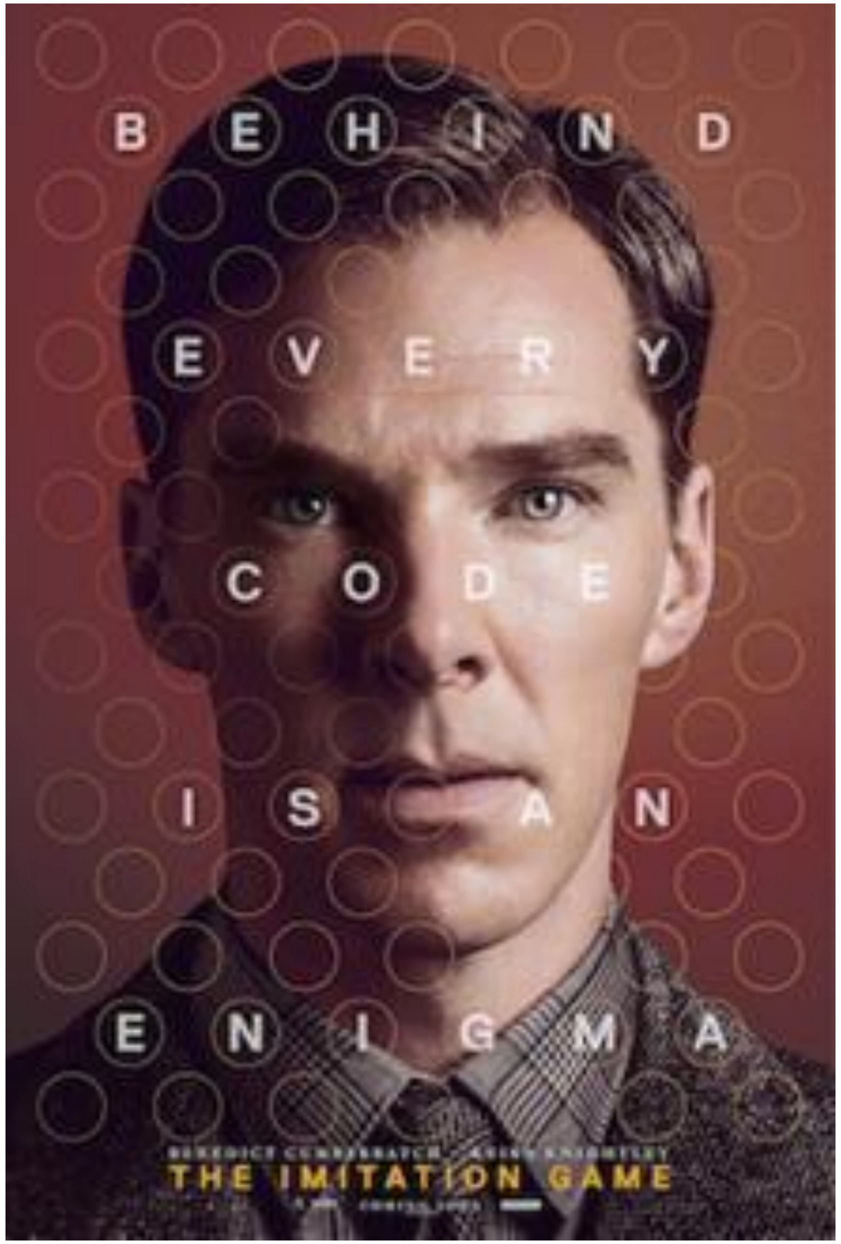
Is there anything missing?

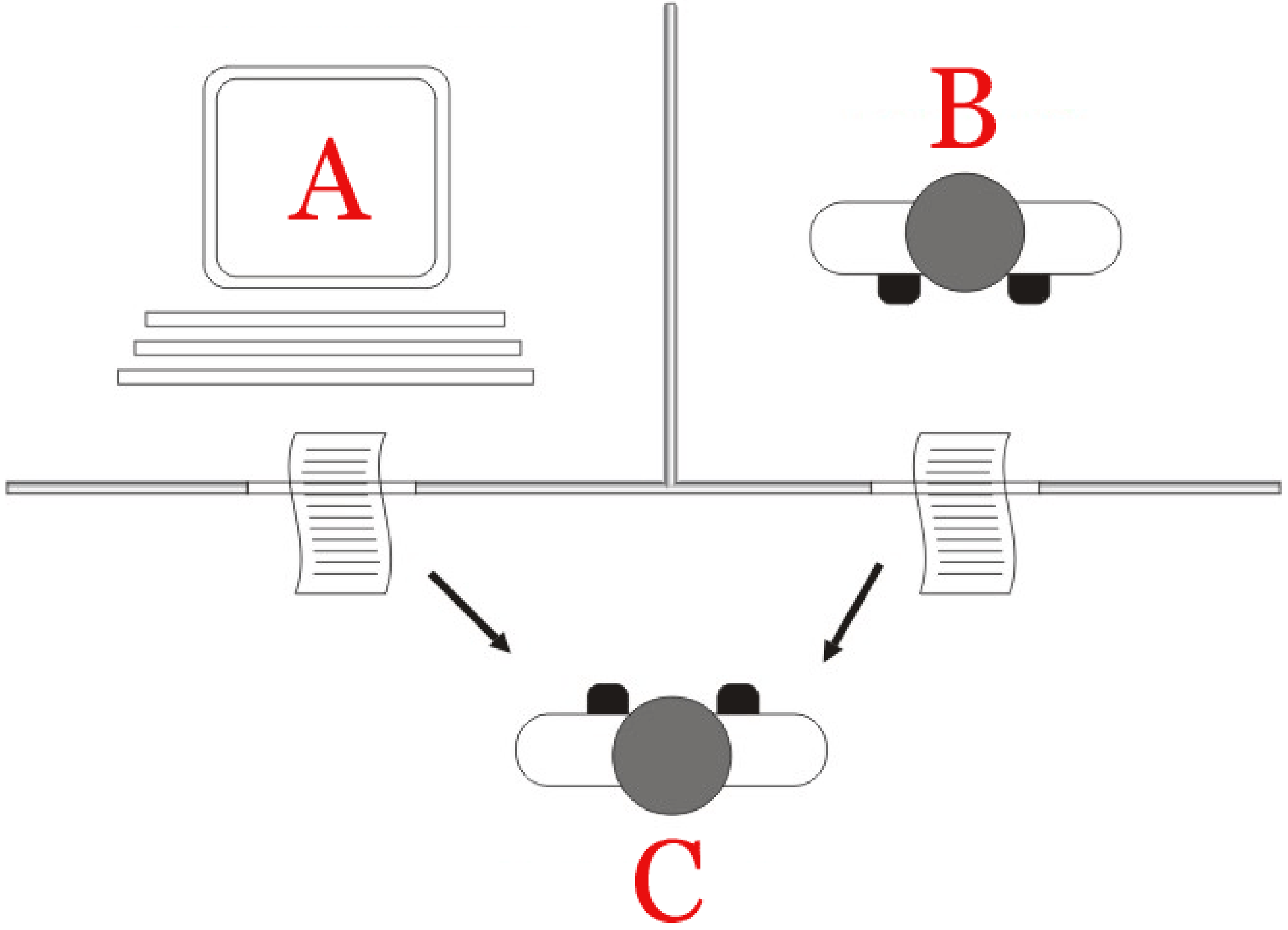
What and Why?



"A computer would deserve to be called intelligent if it could deceive a human into believing that it was human."

Who Said That?


BEHIND
EVERY
CODE
IS AN
ENIGMA
BENEDICT CUMBERBATCH KEIRA KNIGHTLEY
THE IMITATION GAME


A
B
C

Theoretical Computer Science

Alan Turing

Lecture 9 - Manuel Mazzara

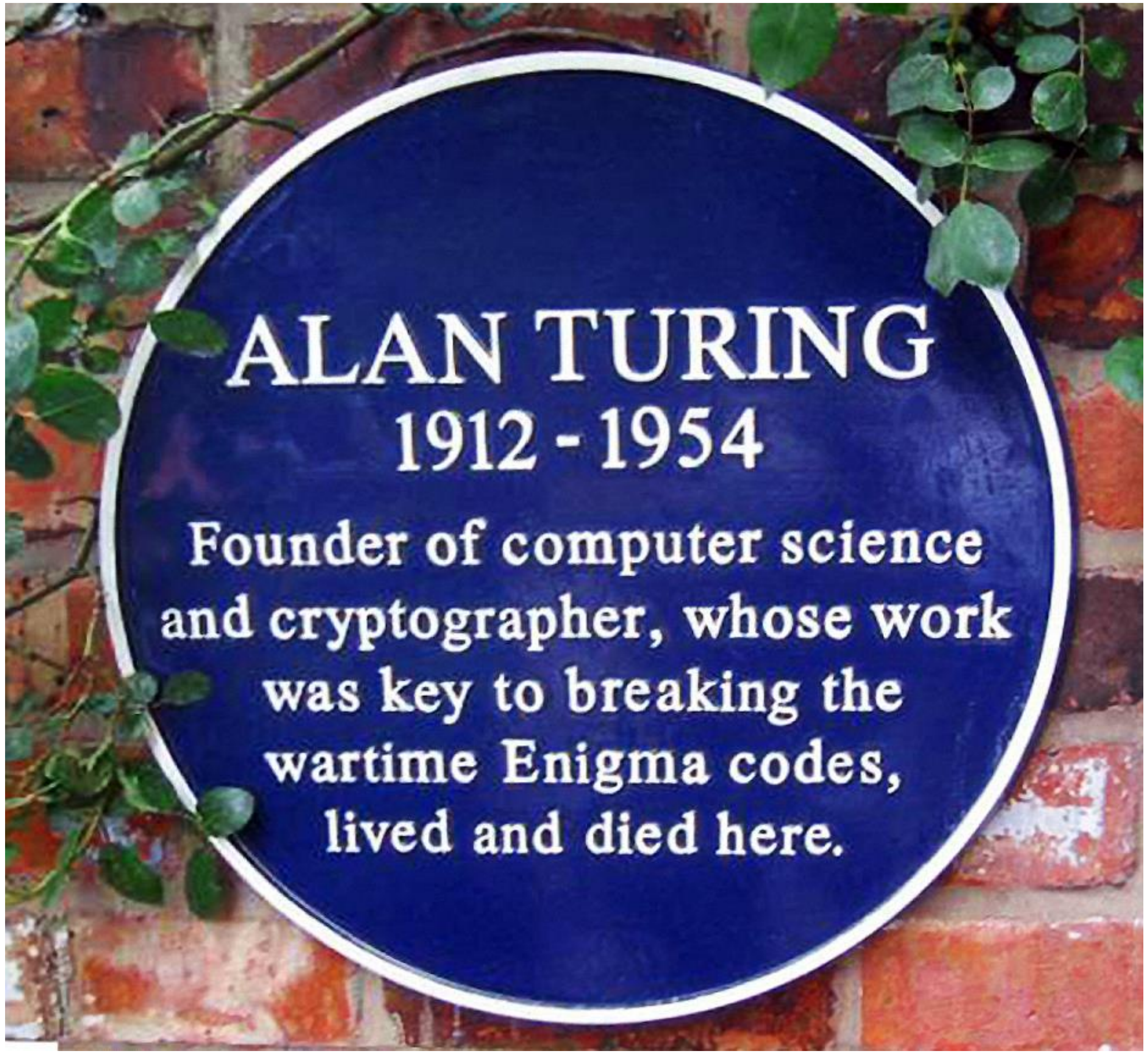
Alan Turing

23 June 1912 – 7 June 1954

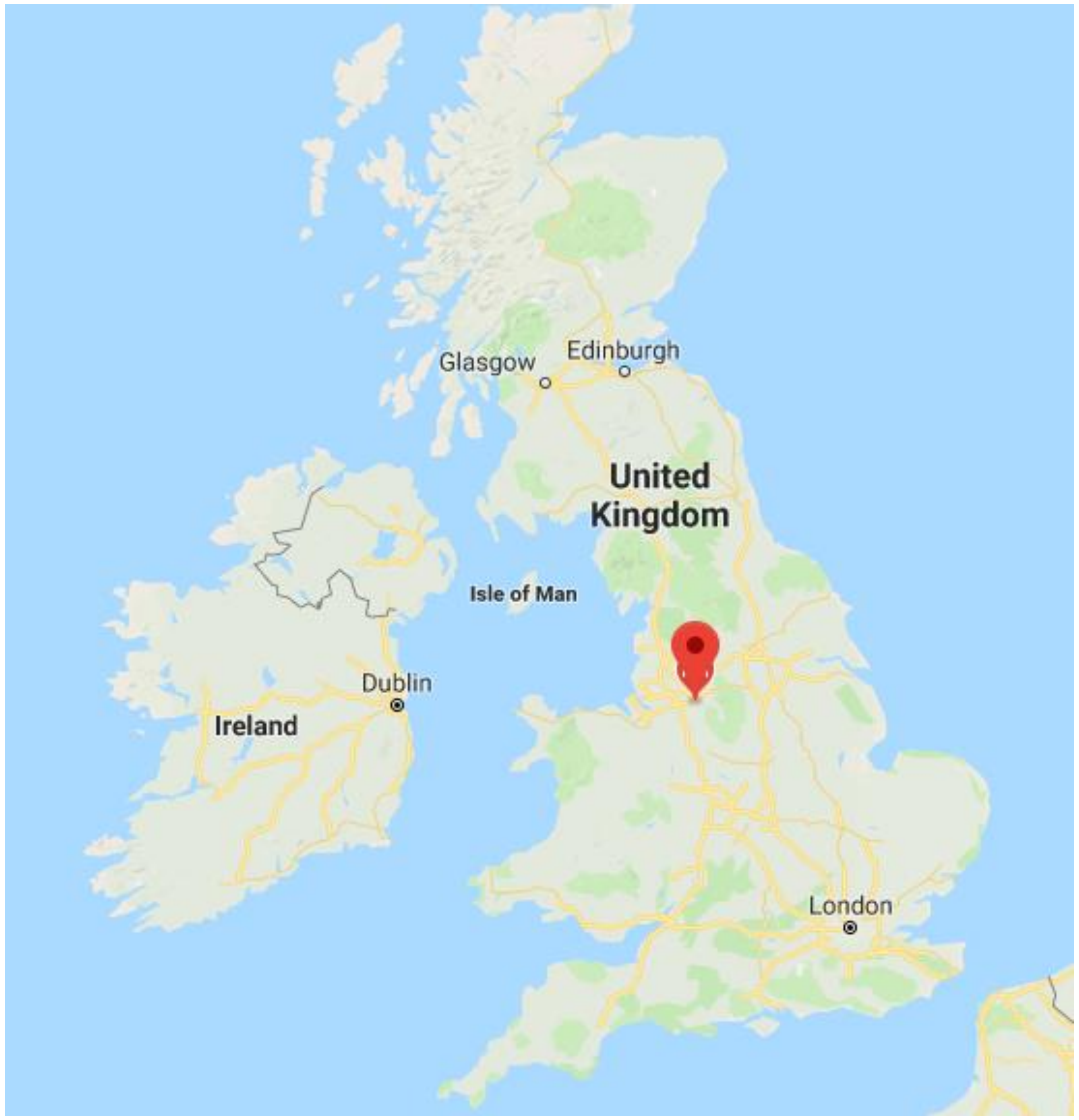
Major contributions in:

- **Computability**
- **Cryptography**
- **Artificial Intelligence**
- **Bioinformatics**

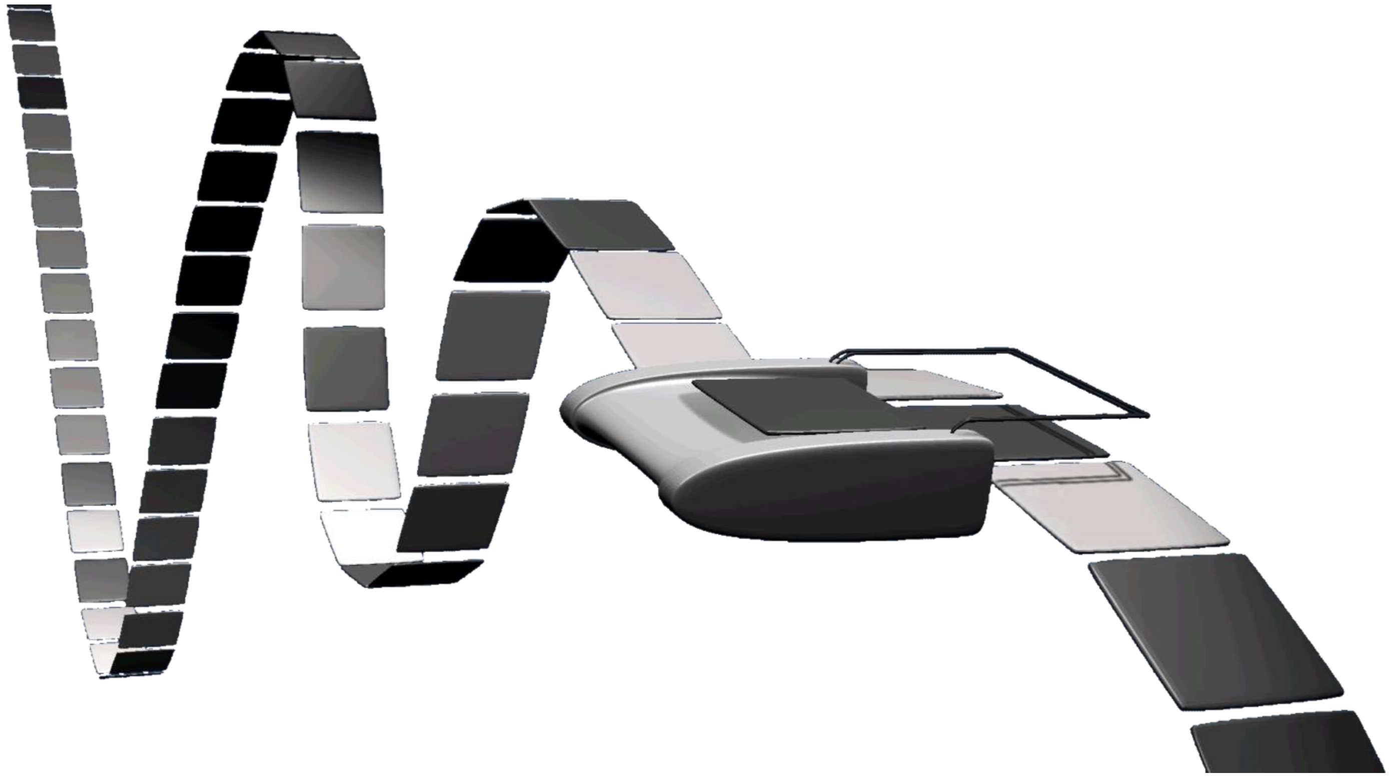
Turing's home at Wilmslow (UK)



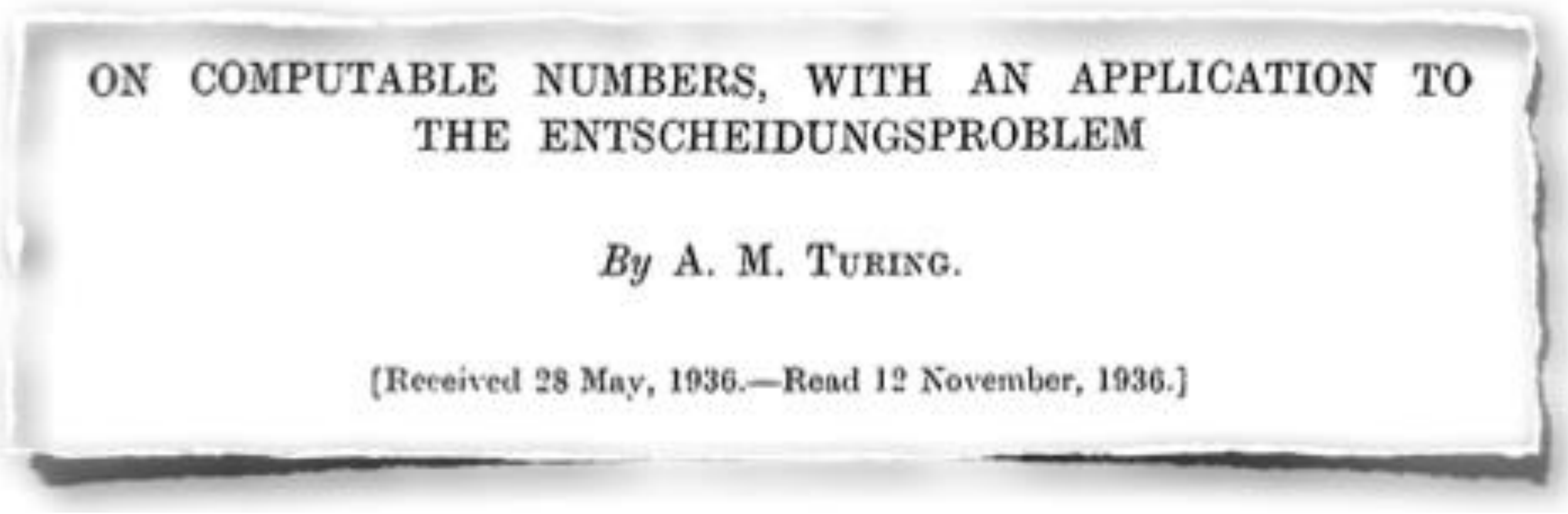
ALAN TURING
1912 - 1954
Founder of computer science and cryptographer, whose work was key to breaking the wartime Enigma codes, lived and died here.


Glasgow
Edinburgh
United Kingdom
Isle of Man
Dublin
Ireland
London

Turing machine (Computability)



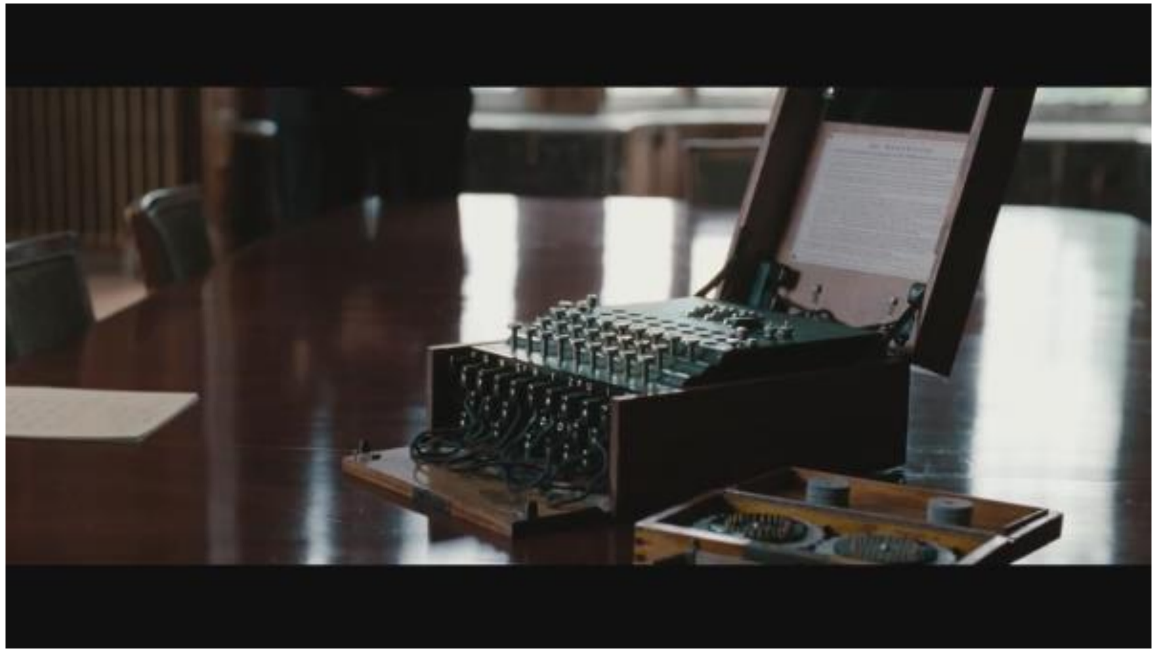
Decision Problem (Computability)



ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

By A. M. Turing.

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



Defeat of Enigma (Cryptography)

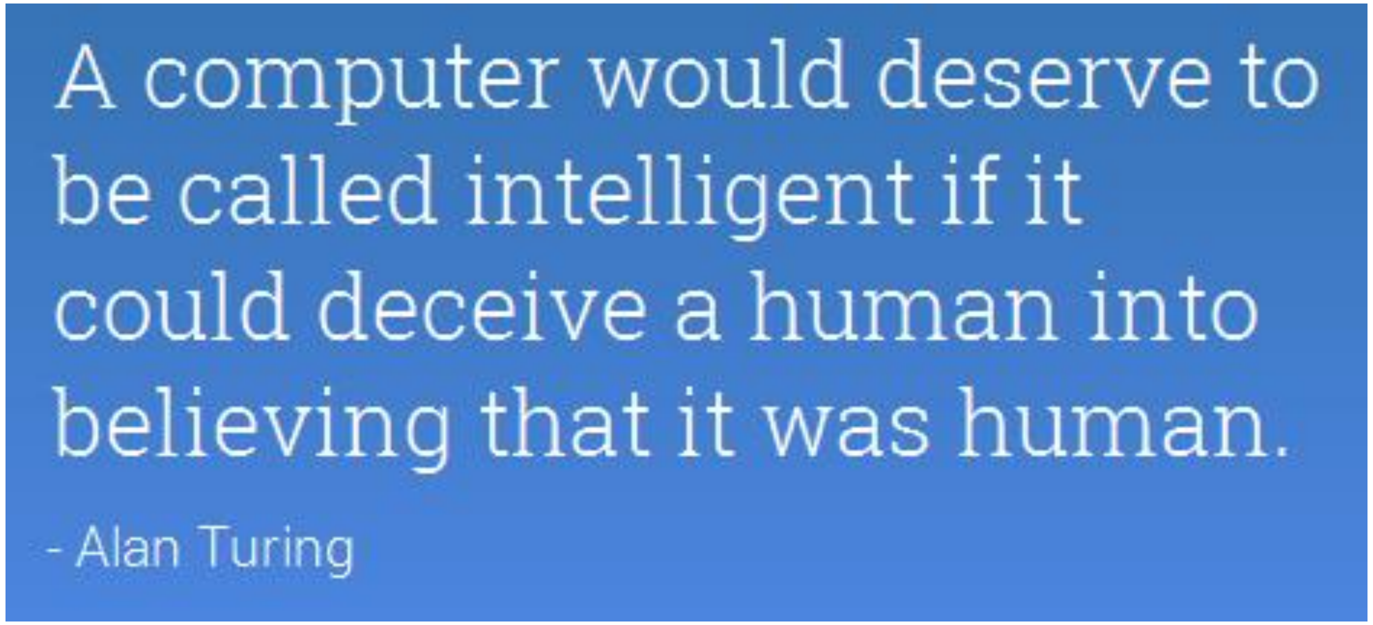
WWII and Turing

- Turing worked for the **Government Code and Cypher School** (GC&CS)
  - Britain's codebreaking centre

- He led **Hut 8**, section responsible for German naval **cryptanalysis**
  - He devised a number of techniques for speeding the breaking of German ciphers

- **Turing played a pivotal role in cracking intercepted coded messages**

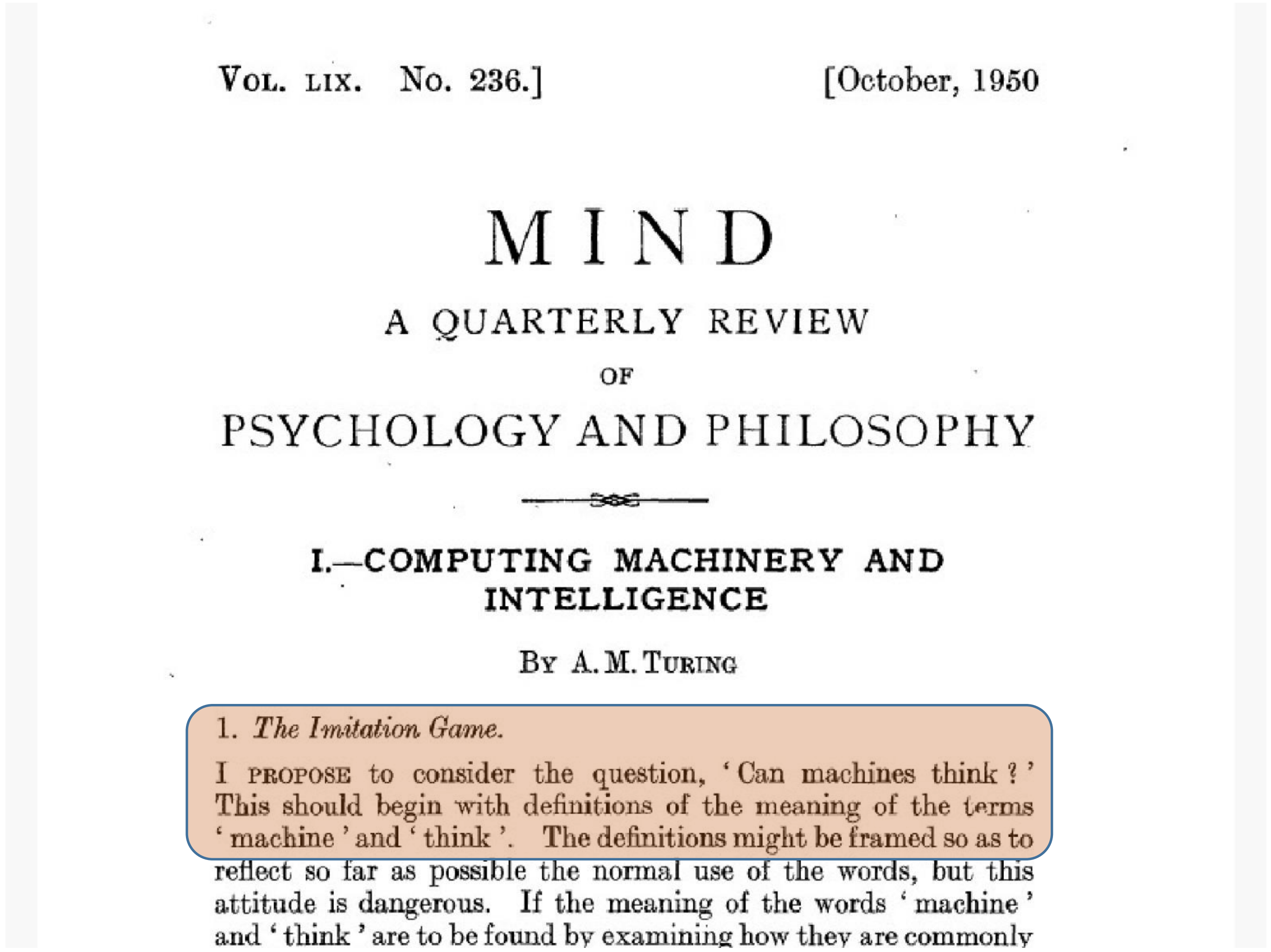
Turing test (AI)



A computer would deserve to be called intelligent if it could deceive a human into believing that it was human.

- Alan Turing

Seminal Turing's article on AI


Vol. LIX. No. 236.] [October, 1950

MIND

A QUARTERLY REVIEW

OF

PSYCHOLOGY AND PHILOSOPHY

I.—COMPUTING MACHINERY AND INTELLIGENCE

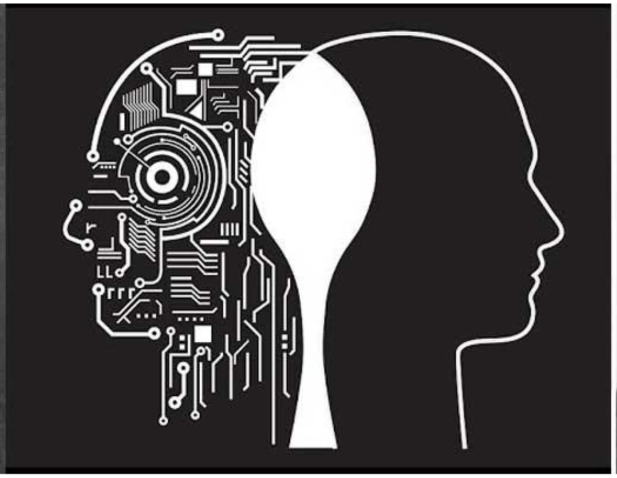
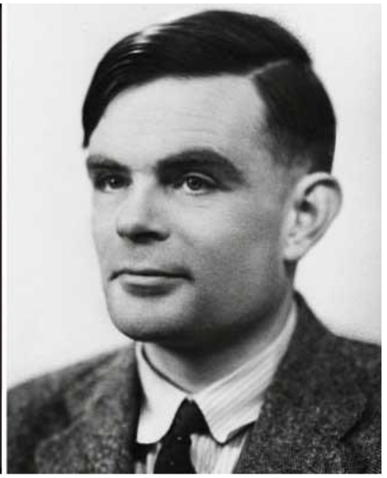
By A. M. Turing

1. *The Imitation Game.*

I propose to consider the question, 'Can machines think?' This should begin with definitions of the meaning of the terms 'machine' and 'think'. The definitions might be framed so as to reflect so far as possible the normal use of the words, but this attitude is dangerous. If the meaning of the words 'machine' and 'think' are to be found by examining how they are commonly

I propose to consider the question, 'Can machines think?

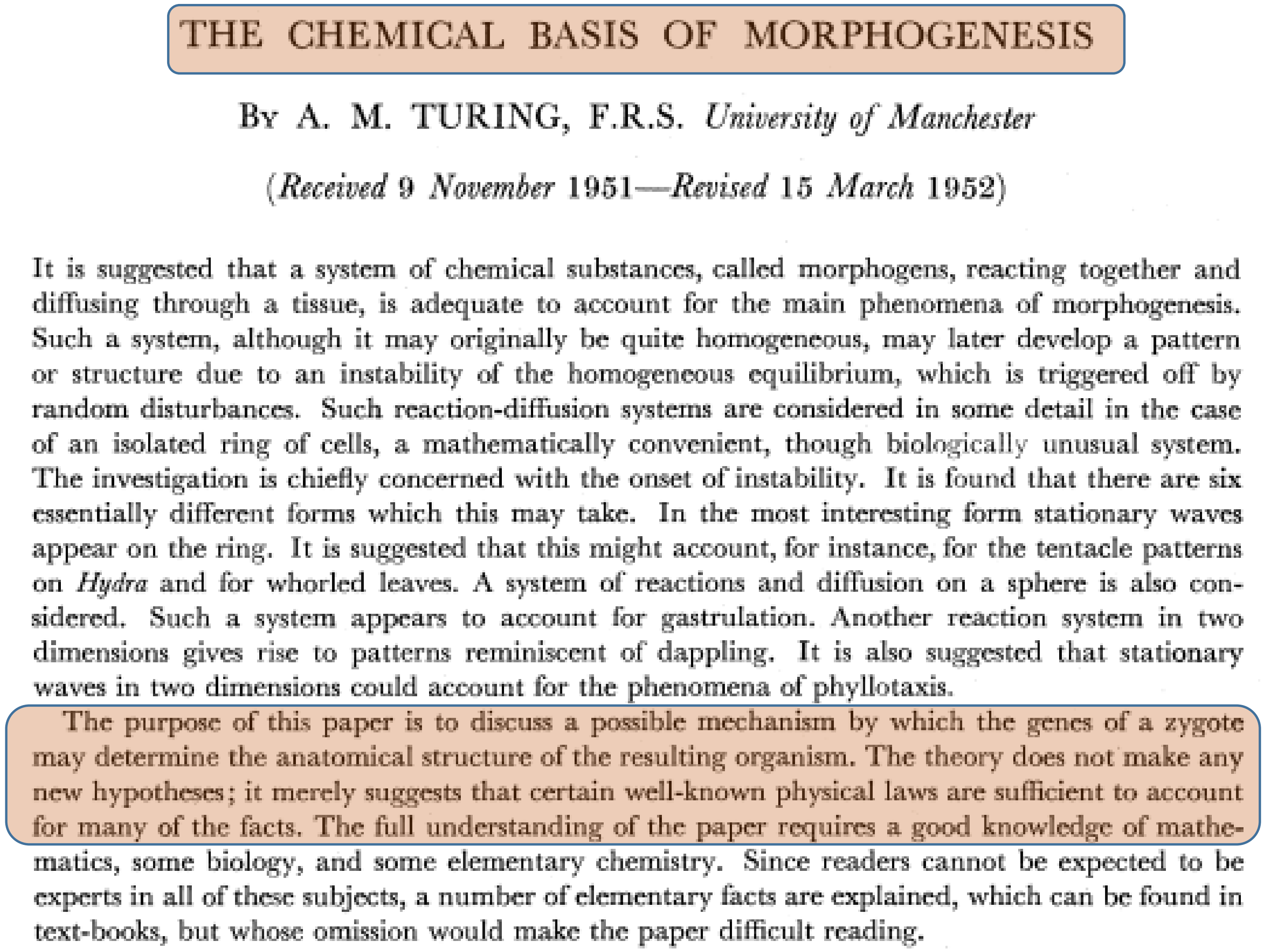
(Alan Turing)

"Let us return for a moment to Lady Lovelace's objection,
which stated that the machine can only do what we tell it to do."
- Alan Turing

TIME
CHINA: Our New Enemy?
THE HENSEL TWINS: Sharing a Body
Can Machines Think?
They already do, say scientists. So what (if anything) is special about the human mind?

Bioinformatics before Bioinformatics



THE CHEMICAL BASIS OF MORPHOGENESIS

BY A. M. TURING, F.R.S. *University of Manchester*

(*Received* 9 *November* 1951—*Revised* 15 *March* 1952)

It is suggested that a system of chemical substances, called morphogens, reacting together and diffusing through a tissue, is adequate to account for the main phenomena of morphogenesis. Such a system, although it may originally be quite homogeneous, may later develop a pattern or structure due to an instability of the homogeneous equilibrium, which is triggered off by random disturbances. Such reaction-diffusion systems are considered in some detail in the case of an isolated ring of cells, a mathematically convenient, though biologically unusual system. The investigation is chiefly concerned with the onset of instability. It is found that there are six essentially different forms which this may take. In the most interesting form stationary waves appear on the ring. It is suggested that this might account, for instance, for the tentacle patterns on *Hydra* and for whorled leaves. A system of reactions and diffusion on a sphere is also considered. Such a system appears to account for gastrulation. Another reaction system in two dimensions gives rise to patterns reminiscent of dappling. It is also suggested that stationary waves in two dimensions could account for the phenomena of phyllotaxis.

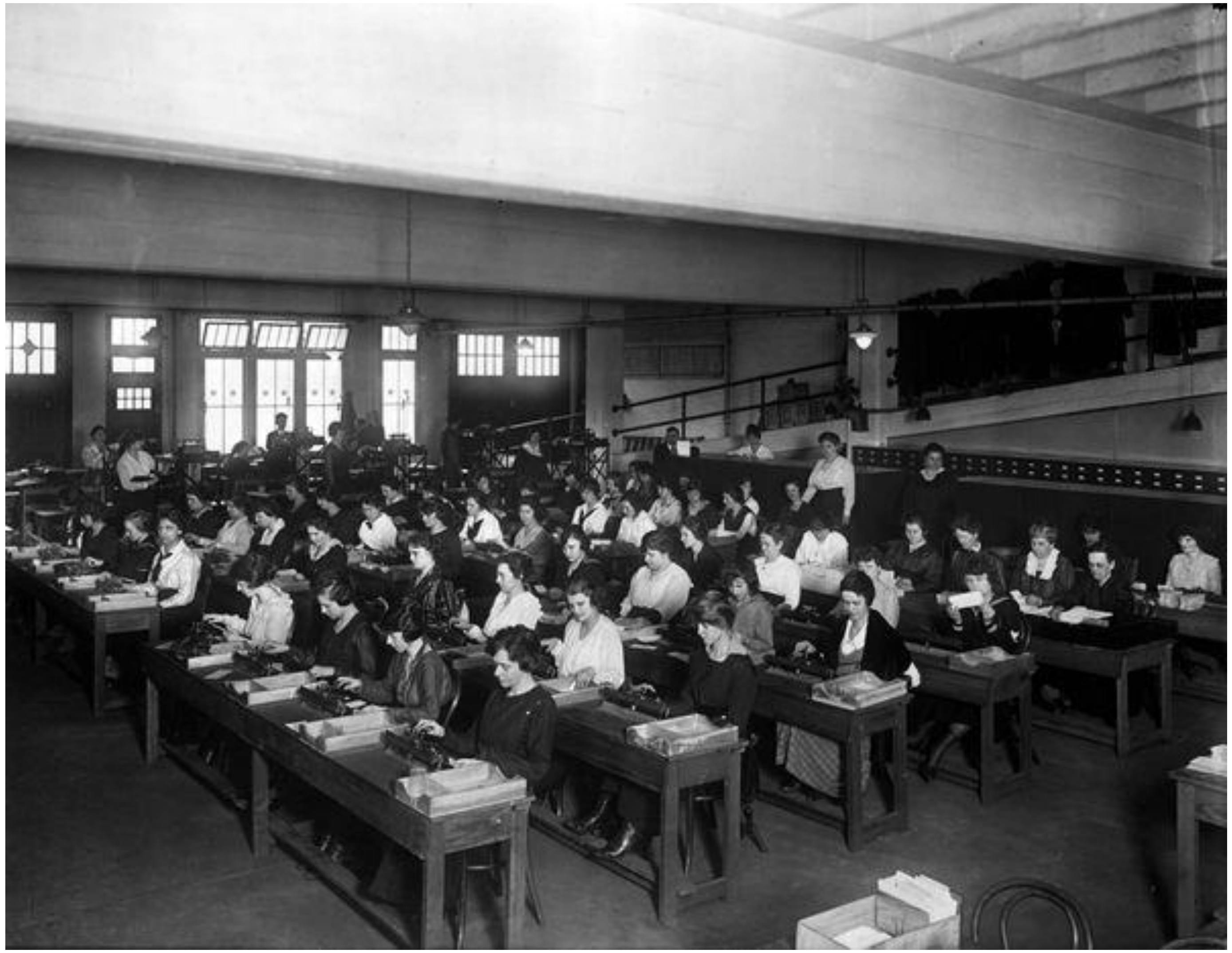
The purpose of this paper is to discuss a possible mechanism by which the genes of a zygote may determine the anatomical structure of the resulting organism. The theory does not make any new hypotheses; it merely suggests that certain well-known physical laws are sufficient to account for many of the facts. The full understanding of the paper requires a good knowledge of mathematics, some biology, and some elementary chemistry. Since readers cannot be expected to be experts in all of these subjects, a number of elementary facts are explained, which can be found in text-books, but whose omission would make the paper difficult reading.

Theoretical Computer Science

The historical context of Turing Machines

Lecture 9 - Manuel Mazzara

Computers *Before* Computers (1930s)



Hierarchy of calculation workers

Computation

- People did not have computers in the 1930s
  - Yet they needed them

- Scientists, engineers, businesses, and government agencies faced growing mountains of tedious, repetitive computations

- Many inventors worked on the automation of these tasks
  - Here we will discuss the work of a man that worked on modelling the idea of computations
  - Why?

Turing machine (1)

- The Turing machine (TM) is the historical model of "computer"
  - simple
  - conceptually important
- TMs use **tapes** as memory
  - Tapes are **not destructive**
  - They can be read many times

Turing Machine (2)

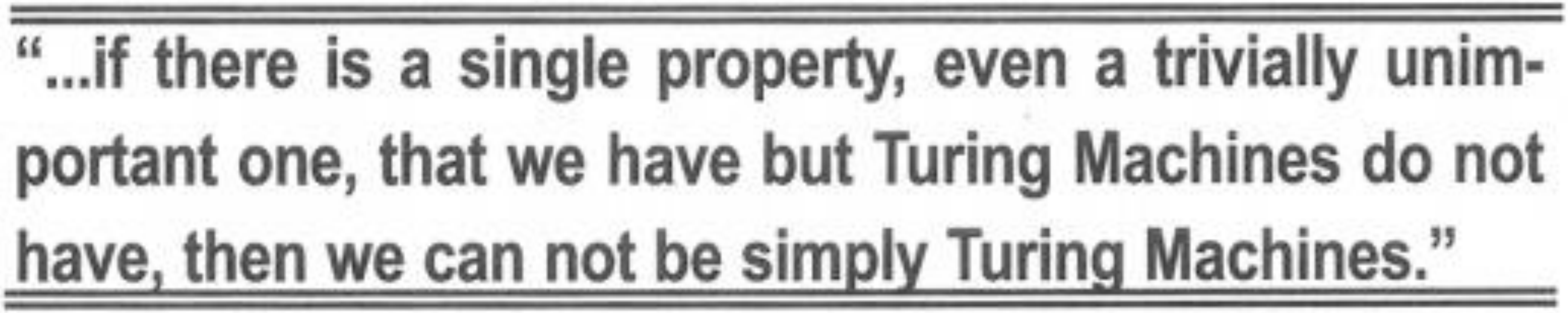
- It is intended **to emulate the human behavior when computing**
- Limits of mechanical computation are in common with *"human computation"*
- Performance is another issue

Turing Machine and brains (1)

- The fact that Turing devised the machine to emulate human's behaviour in the computation process **<u>does not directly imply that we are Turing's Machine</u>**
  - It may ore may not be, no one knows

THE FUNDAMENTAL DISTINCTION BETWEEN BRAINS AND TURING MACHINES

By Andrew Friedman


"...if there is a single property, even a trivially unimportant one, that we have but Turing Machines do not have, then we can not be simply Turing Machines."

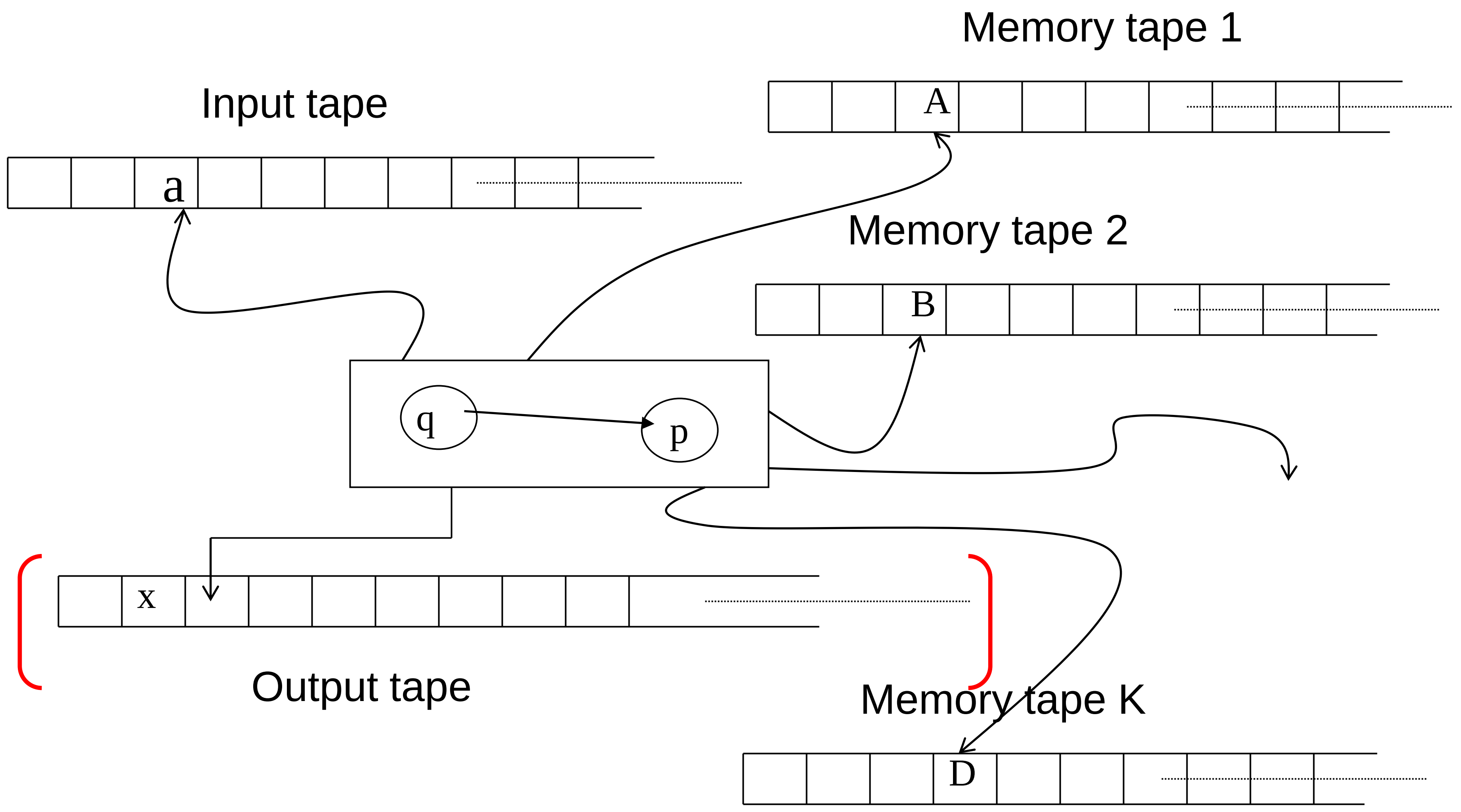
Turing Machine and brains (2)

Debate

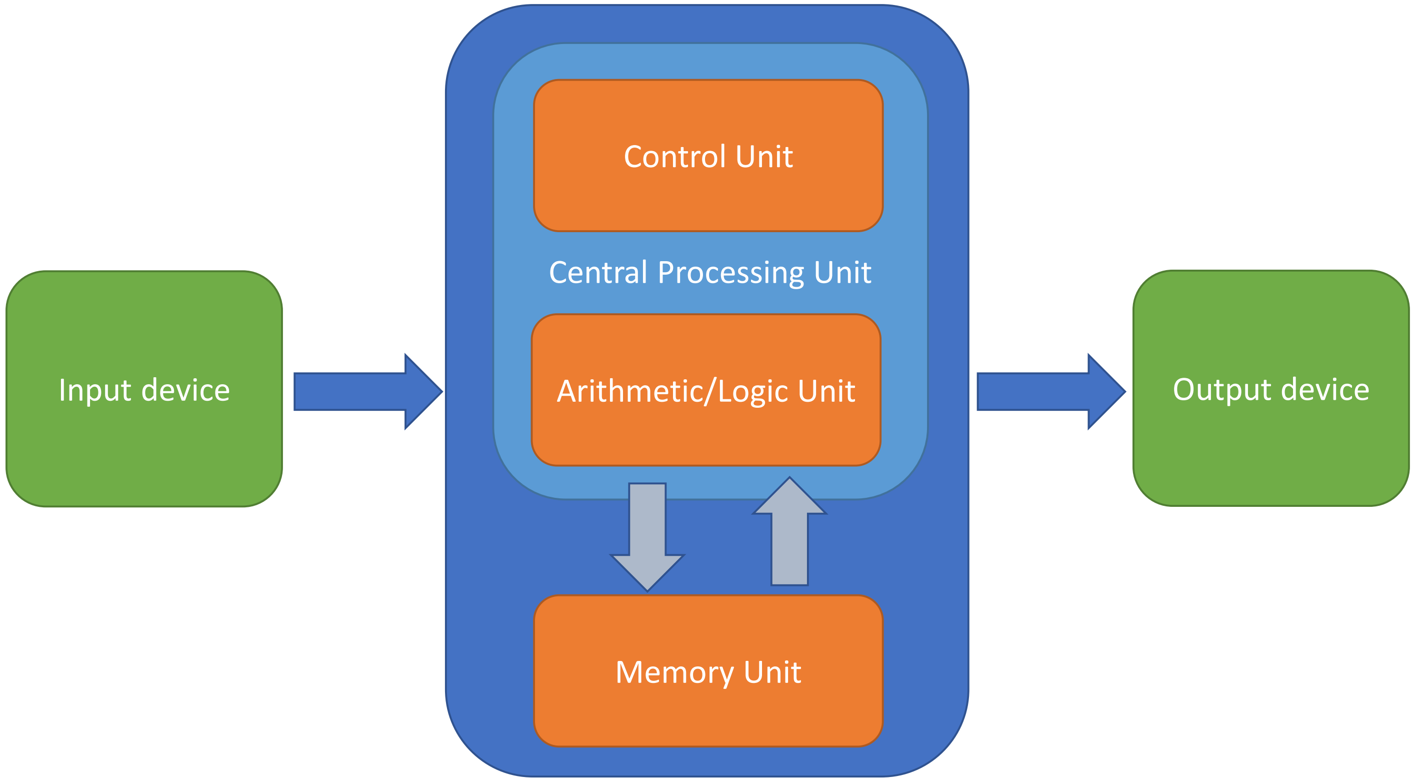
- The debate is open here, also in relation with **AI and Turing test**
- Turing's idea are still debated after 80 years showing the level of his genius

Figure 4: Mathematicians represent the different sizes of infinity with the Hebrew letter Aleph and the subscripts 1 and 0, where Aleph 1 represents the "larger" infinity of the real numbers, and Aleph 0 represents the standard, "smaller" infinity of the integers. What is relevant here is that while Turing Machines are most certainly limited to the smaller infinity, the minds of human beings may not be, making us fundamentally different from computers.

The general model



Von Neumann Architecture



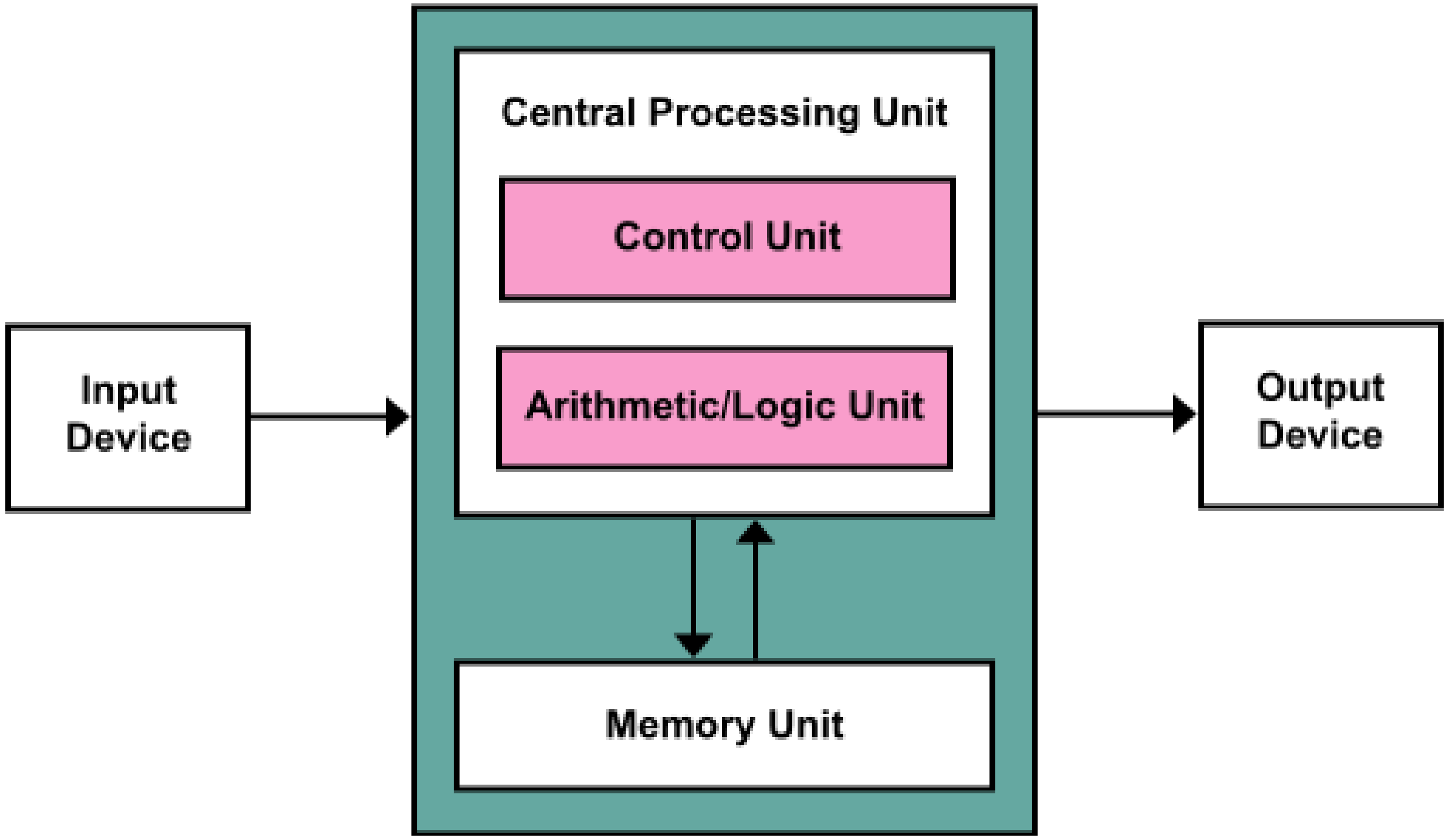
What is the key idea behind the Von Neumann Architecture?

What is a stored-program computer?

Stored-program computer

- Instructions are in memory and the program can be changed
- The **Northrop Loom** will always run a “loom program”

Von Neumann Machines (VNM)

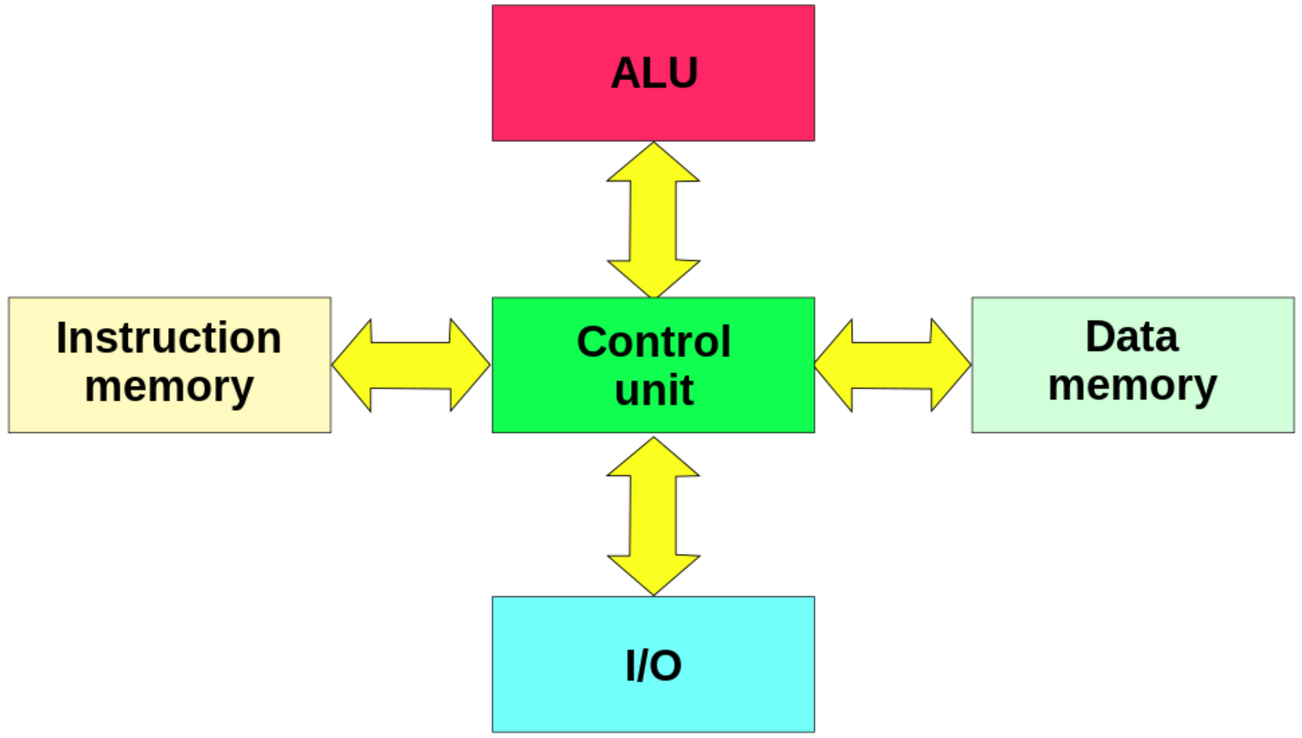


It seems a trivial idea these days, but it is very powerful and innovative!

Harvard Architecture

- **A computer that stores program instructions in electronic memory**

- Not a synonym for von Neumann architecture
    - **Harvard architecture** is a computer architecture with physically separate storage and signal pathways for instructions and data



Harvard architecture

TM vs Von Neumann Machines

- TMs can simulate a Von Neumann machine (VNM)
  - It is an abstract model of computers
- TM differs from VNM wrt. memory access
  - TM: sequential
  - VNM: direct

- This difference <u>does not affect the expressive power</u> of a machine
  - It does not change the class of problems solvable with a machine
  - It may affect the complexity

Why TMs?

- TMs have the **same expressive power** as high-level programming languages
  - **Church-Turing thesis**

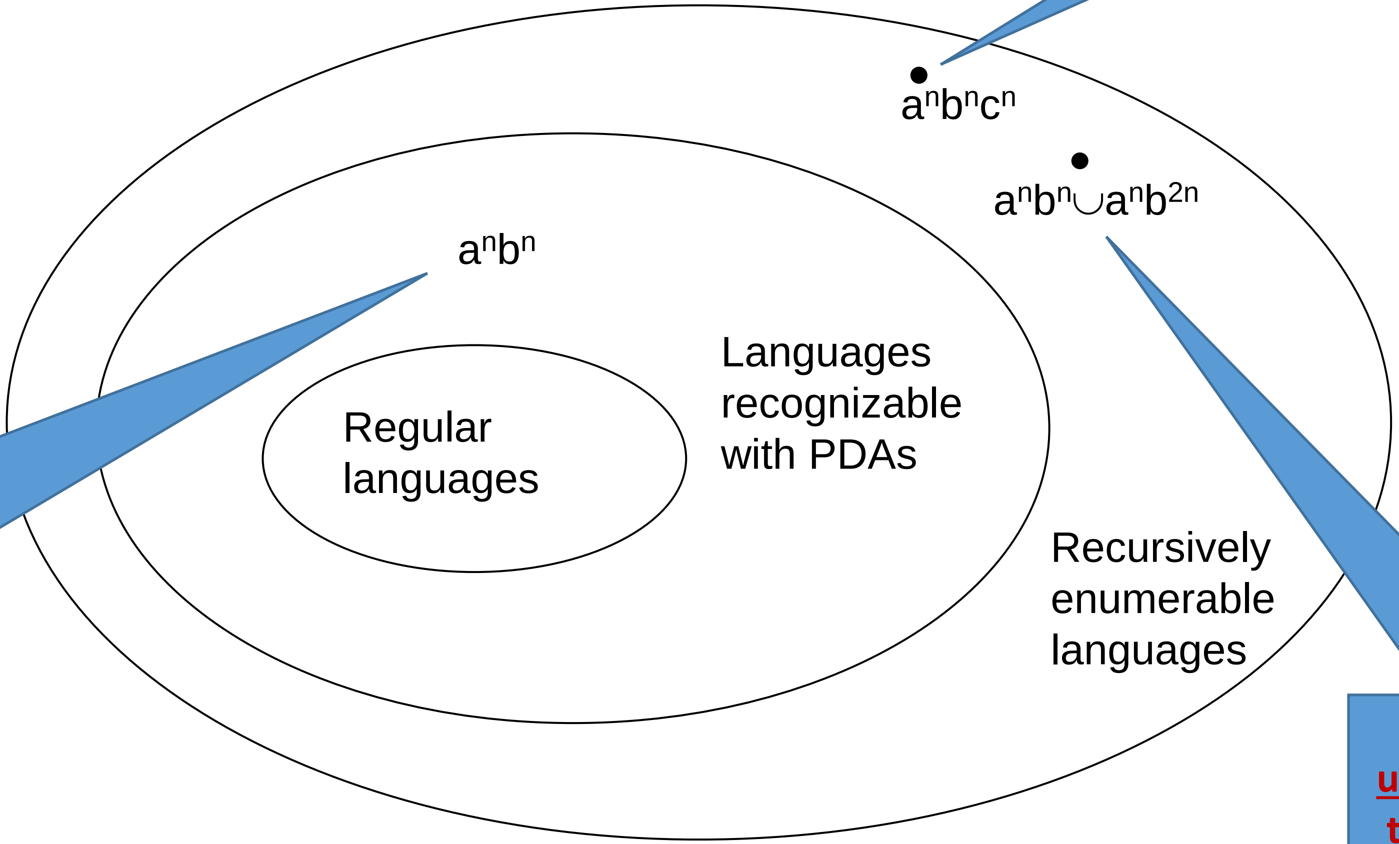
- TMs are **theoretical models** not really meant for programming but for proofs and understanding

Theoretical Computer Science

Turing Machines in context

Lecture 9 - Manuel Mazzara

Languages



$a^n b^n c^n$
$a^n b^n \cup a^n b^{2n}$
$a^n b^n$
Languages recognizable with PDAs
Regular languages
Recursively enumerable languages

PDAs have a destructive external memory

FSA can only count fixed numbers (finite states), no generic *n*

In programming languages we need to syntactically check nested structures

PDAs are not closed under union: stack needs to be prepared for what we are looking for

History

- The Turing machine was invented in 1936 by Alan Turing, (*a-machine*, automatic machine)
- It is a mathematical description of a very simple device for arbitrary computations
- It is intended to show theoretical results of computability theory

Entscheidungsproblem

David Hilbert (23 January 1862 – 14 February 1943)

- Was devoted to axiomatizing mathematics

- In 1928, proposed **Entscheidungsproblem**

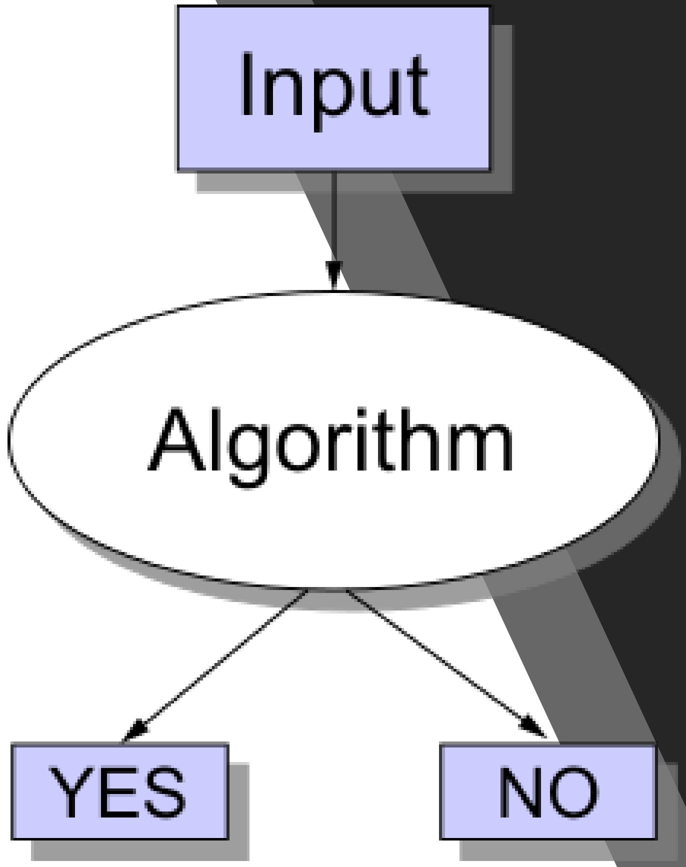
Entscheidungsproblem - decision problem

Hilbert Problems and Decision Problem

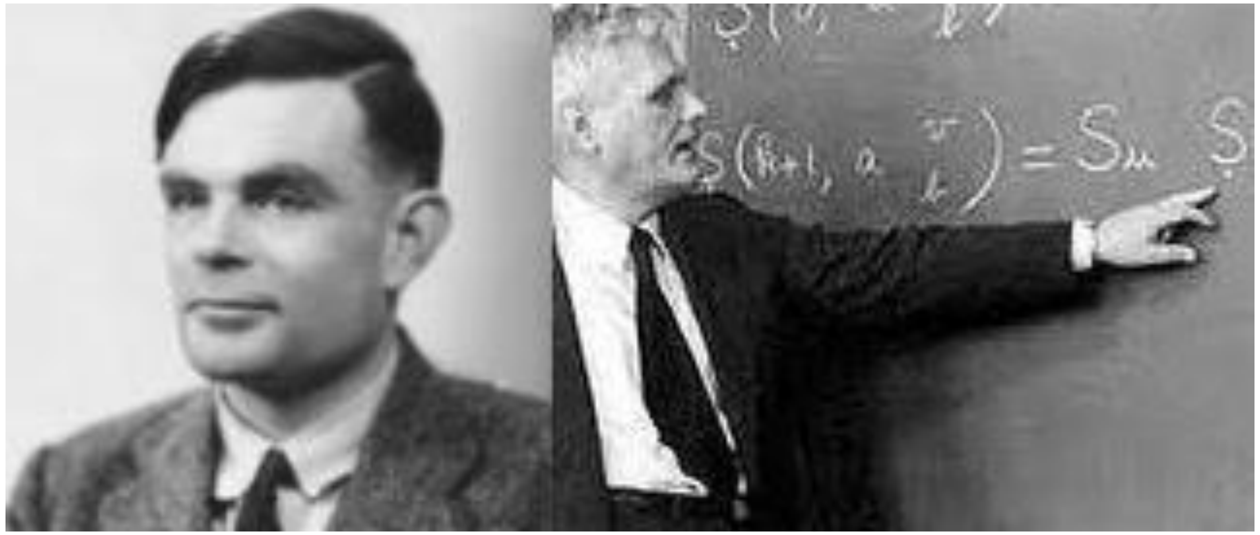
- *Hilbert Problems (1900)*
  - 23 unsolved problems in mathematics
  - Important mathematical challenges for the 20th century

- *Entscheidungsproblem* (1928)
  - in the original German
  - David Hilbert and Wilhelm Ackermann

Decision Problem



- Is it possible to find a set of *basic truths* (*axioms*) from which all statements in mathematics can be proven, without giving any contradictory answers such as 1=0?
- Is there an effective procedure (*algorithm*) which, given a set of axioms and a mathematical proposition, decides whether it is or is not provable from the axioms?
- Hilbert was asking whether mathematics was ***complete***, ***consistent*** and ***decidable***.

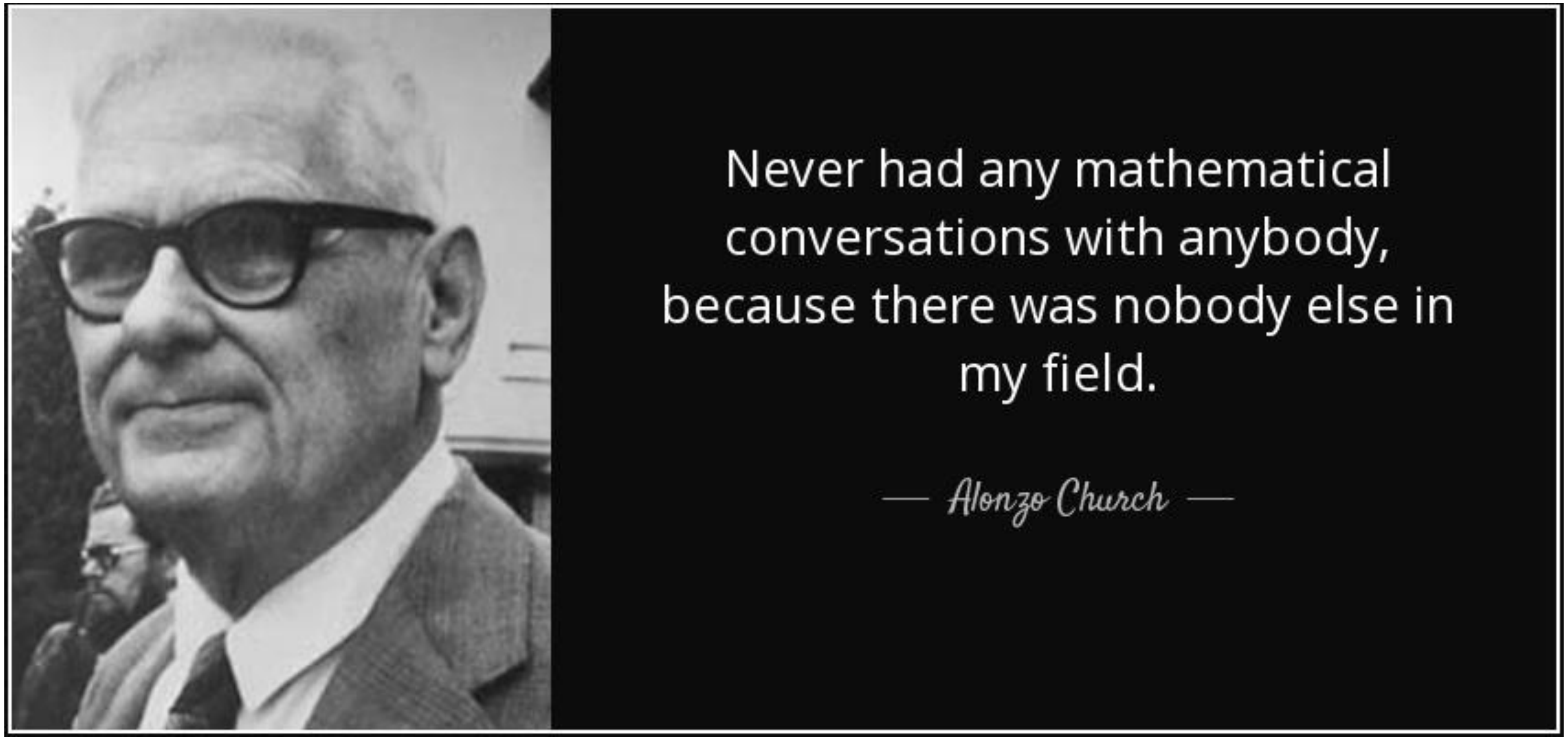


- In 1936, Alonzo Church and Alan Turing published independently papers showing that **a general solution to the problem is impossible**

- An algorithm to answer the general decision problem **does not exist**

Answer to the Decision Problem

Alonzo Church



Never had any mathematical conversations with anybody, because there was nobody else in my field.

— Alonzo Church —

Proving decision problem negative

ALAN TURING SHOWED THAT THE ALGORITHM FOR DECIDING THE DECISION PROBLEM COULD NOT EXIST

TURING MACHINE

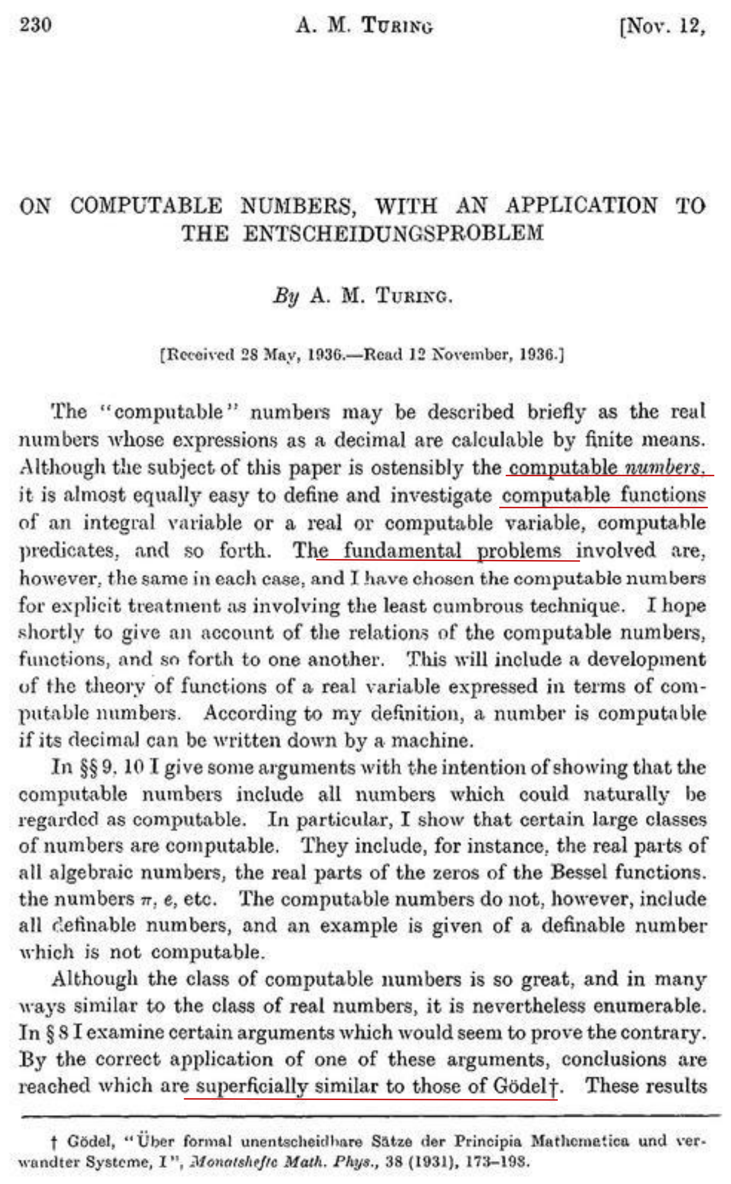
HALTING PROBLEM

Church–Turing thesis

- The intuitive notion of *"effectively calculable"* is captured by
  - the functions computable by a Turing machine
  - by those expressible in the Lambda calculus

- This assumption is now known as the **Church–Turing thesis**
  - The Church-Turing thesis conjectures that ***any computation*** done by ***any realizable physical process*** can be computed by the universal computational models described by Church and Turing

- We will see the details of this later



230 A. M. Turing [Nov. 12,

ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

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 cumbrous 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.

- In 1936 Turing devised a powerful yet simple **model of computer**

- It has provided the **standard definition of computability**

- He proved that **it is not logically possible to write a computer program which can distinguish between programs that halt, and those that "loop" forever**

Turing’s Philosophical Contribution

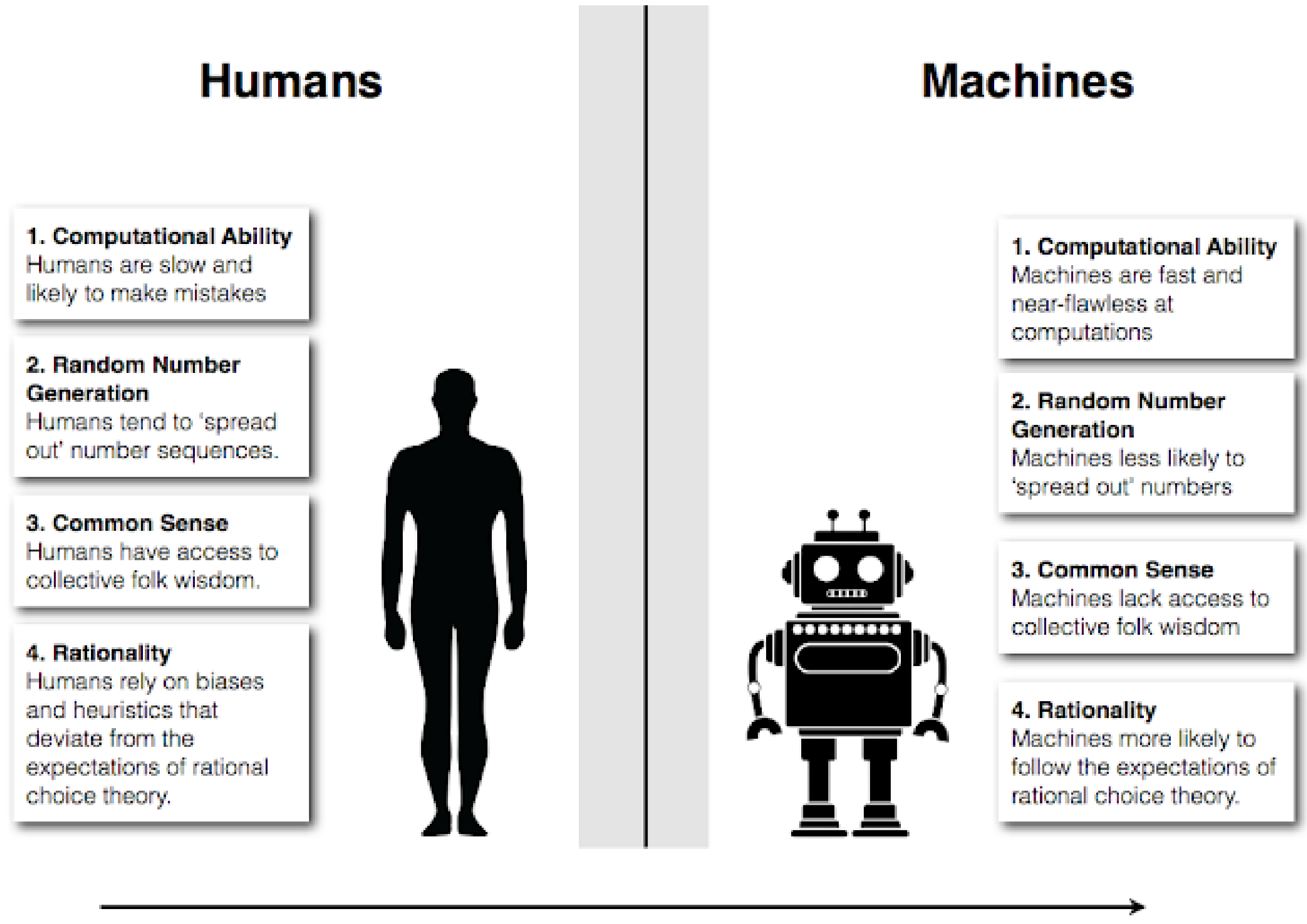
- Epistemological break between **idealism and materialism** in mathematics
- Before Turing: through mathematical reason, **the human mind can access a higher domain of Platonic truths (idealism)**
- After Turing: **TM is a model of human mathematician (materialism)**
  - By showing limits of this machine, he identifies constraints which bind mathematical reasoning in general
  - **Ground mathematics on the material world**
  - What can be allowed by the laws of physics

Conclusions

- The central contribution of Turing to science and philosophy:
  - **Treating symbolic logic as a new branch of applied mathematics to which giving a physical and engineering content**

- Distinction between **solvable** and **unsolvable problems**
  - The field of computability

The Turing line



1. Computational Ability
Humans are slow and likely to make mistakes

2. Random Number Generation
Humans tend to 'spread out' number sequences.

3. Common Sense
Humans have access to collective folk wisdom.

4. Rationality
Humans rely on biases and heuristics that deviate from the expectations of rational choice theory.

1. Computational Ability
Machines are fast and near-flawless at computations

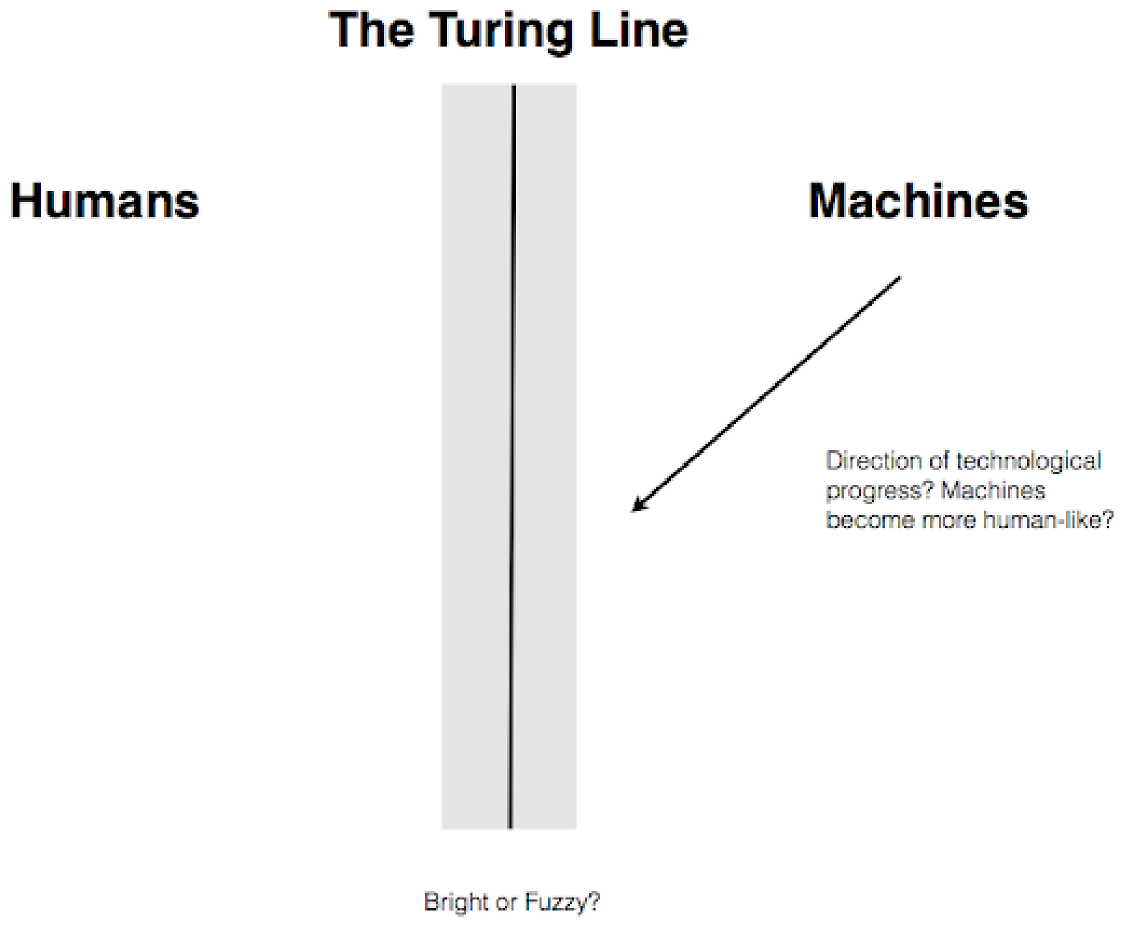
2. Random Number Generation
Machines less likely to 'spread out' numbers

3. Common Sense
Machines lack access to collective folk wisdom

4. Rationality
Machines more likely to follow the expectations of rational choice theory.

Is the techno-social environment making humans more machine-like? (from Frischmann 'Human Focused Turing Tests')

Really?


The Turing Line
Humans
Machines
Direction of technological progress? Machines become more human-like?
Bright or Fuzzy?

A little spoiler…

- **Programs** are countable (lexicographically)
- **Problems** (functions N->N) are not countable
- There are **infinitely fewer** programs than problems
- However, we are not interested in random problems
- Simple, well-structured problems are often decidable
- …but not always (e.g. Halting Problem)

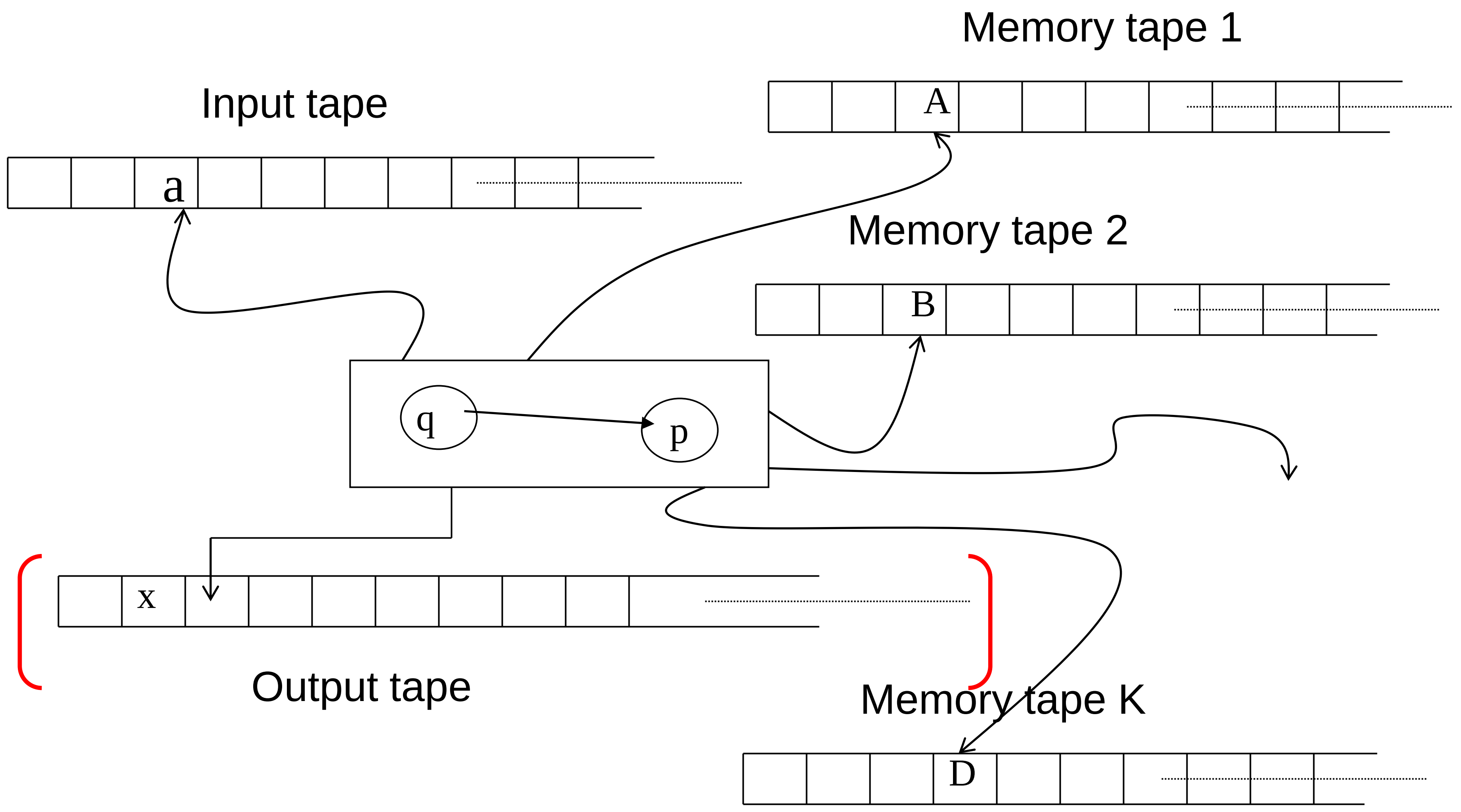
- We will see this in the detail at the end of the course!

Theoretical Computer Science

Turing Machines

Lecture 9 - Manuel Mazzara

Turing Machines



Informal description

- States and alphabet are as in the other automata
  - Input
  - Output
  - Control device
  - Memory alphabet
- Tapes are represented as infinite cell sequences with a special “blank” symbol (or barred ‘b’ or ‘_’ or ‘-’)
  - Tapes contain only a finite number of non-blank symbols

Moves

- Moves are based on
  - one symbol read from the input tape
  - K symbols, one for each memory tape
  - state of the control device
- Actions
  - Change state
  - Write a symbol replacing the one read on each memory tape
  - Move the K+1 heads

Moves of the heads

- Memory and input scanning heads can be “moved” in three ways
  - One position right (R)
  - One position left (L)
  - Stand still (S)
- The direction of each head must be specified explicitly

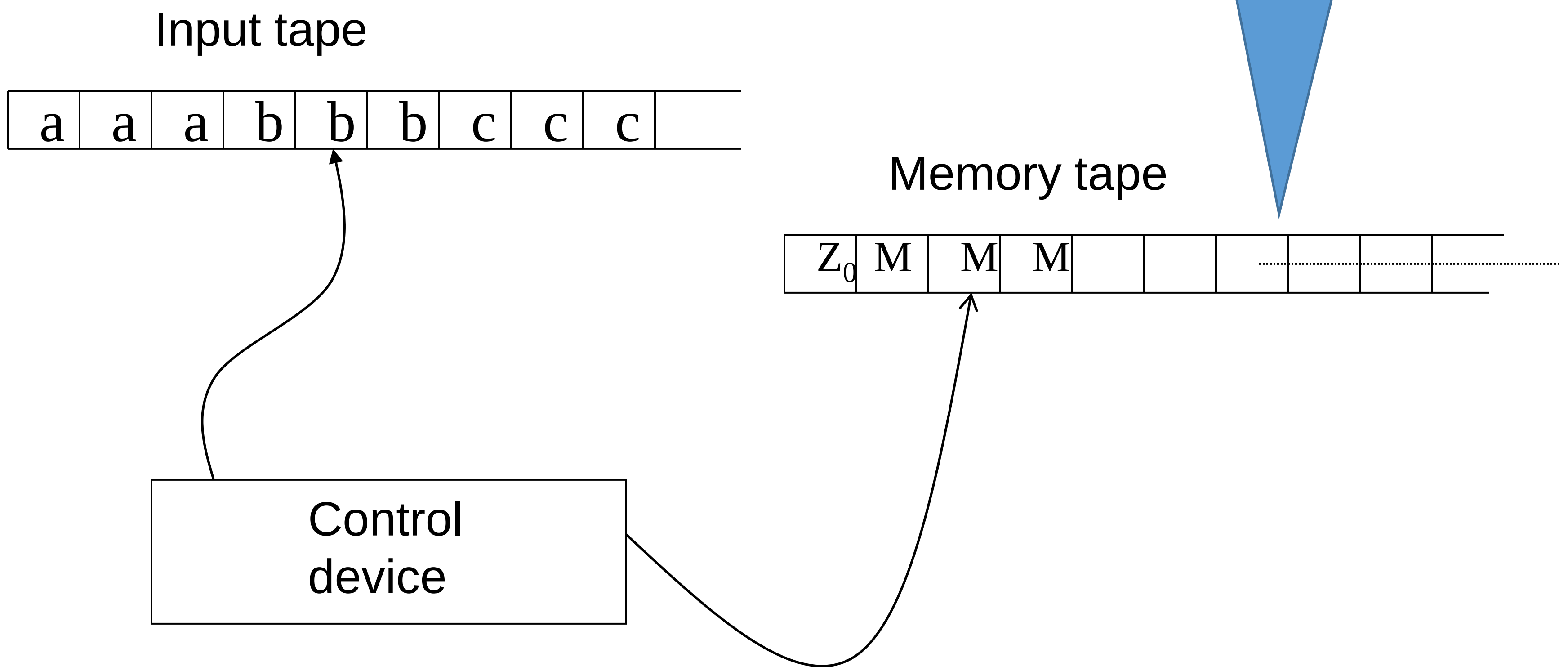
Graphically

$$\begin{tikzcd}[column sep=huge] q \arrow[r, "{i,\langle A_1,A_2,\ldots,A_k\rangle \mid \langle A'_1, A'_2,\ldots,A'_k\rangle , \langle M_0, M_1, \ldots,M_k \rangle}"] & q' \end{tikzcd}$$

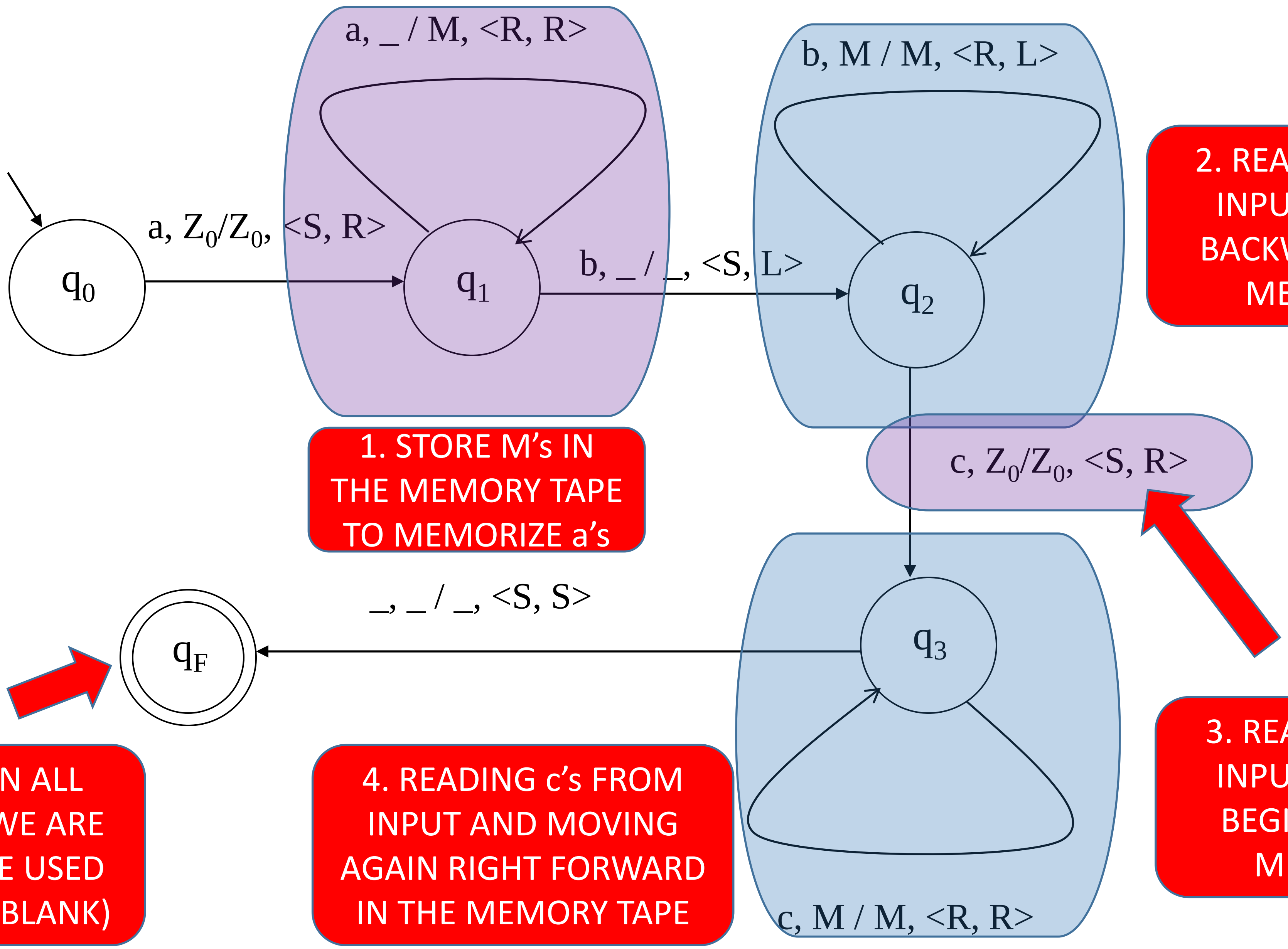
- ‘i’ is the input symbol
- A_j is the symbol read from the jth memory tape
- A_j’ is the symbol replacing A_j
- M_0 is the direction of the head of the input tape
- M_j $(1 \leq j \leq k)$ is the direction of the head of the jth memory tape

Example 1

$L=\{a^n b^n c^n \mid n > 0\}$



PDAs have a destructive external memory
TMs have persistent memory tapes


q_0
$a,\ Z_0/Z_0,\ \langle S, R\rangle$
q_1
$a,\ \_\ /\ M,\ \langle R, R\rangle$
$b,\ \_\ /\ \_,\ \langle S, L\rangle$
q_2
$b,\ M\ /\ M,\ \langle R, L\rangle$
$c,\ Z_0/Z_0,\ \langle S, R\rangle$
q_3
$c,\ M\ /\ M,\ \langle R, R\rangle$
$\_,\ \_\ /\ \_,\ \langle S, S\rangle$
q_F

1. STORE M's IN THE MEMORY TAPE TO MEMORIZE a's

2. READING b's FROM INPUT AND GOING BACKWARDS ON THE MEMORY TAPE

3. READING c's FROM INPUT ONCE AT THE BEGINNING OF THE MEMORY TAPE

4. READING c's FROM INPUT AND MOVING AGAIN RIGHT FORWARD IN THE MEMORY TAPE

5.INPUT HAS BEEN ALL CONSUMED AND WE ARE AT THE END OF THE USED TAPE (YOU READ A BLANK)

Formally

- A TM with K tapes is a tuple of 7 elements $M=<Q,I,\Gamma,\delta,q_0,Z_0,F>$
  - Q is a finite set of states
  - I is the input alphabet
  - Γ is the memory alphabet
  - δ is the transition function
  - $q_0 \in Q$ is the initial state
  - $Z_0 \in \Gamma$ is the initial memory symbol
  - $F \subseteq Q$ is the set of final states