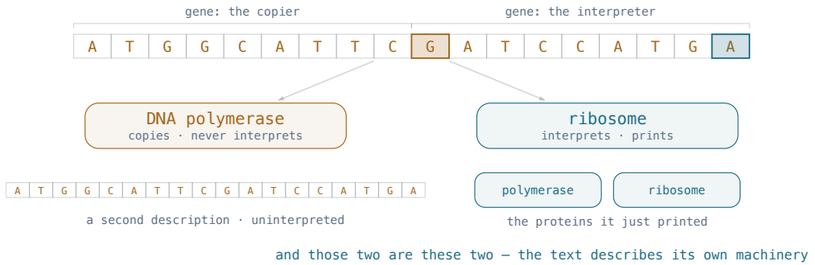
INTRODUCTION TO COMPUTER SCIENCE · WEEK 14 · STATION VI

So — what is intelligence?

A method that does not care which model is current, which company leads, or what happens next year.

EVERYWHERE · BIOLOGY

A copier that never interprets, and an interpreter that prints the copier and itself.



von Neumann's architecture, as run by a cell

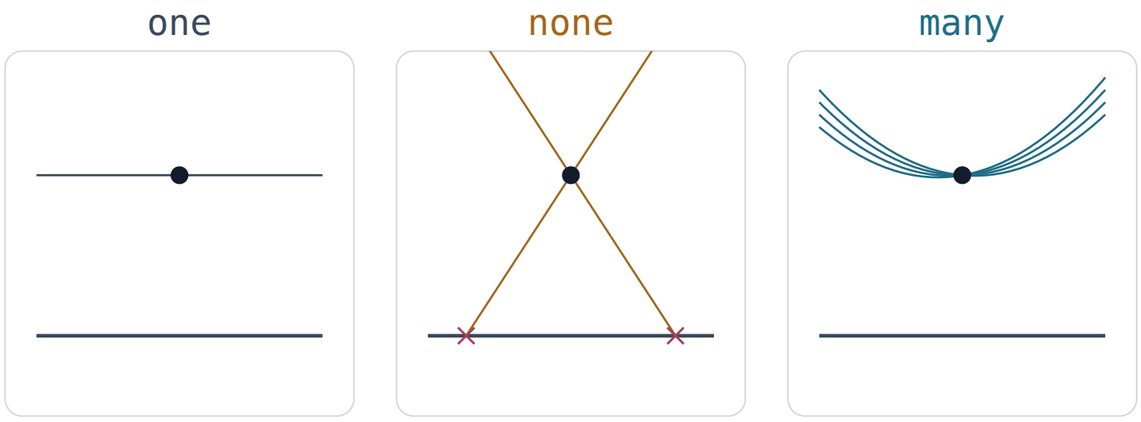
Copy. Polymerase walks the strand and duplicates it letter by letter, never asking what a letter means. Notation to notation.

Interpret. The ribosome walks the same strand and executes it, three letters at a time — description in, object out. Notation to meaning.

Close it. Among the objects it prints are the polymerase and the ribosome — and one ribosome prints any protein, because which protein is data on the strand. That is U from week 9, in chemistry. Turing did not invent universality. He noticed it.

EVERYWHERE · MATHEMATICS, PHYSICS, CHEMISTRY

The postulate nobody could prove was a door, not a wall.



all three are consistent · the postulate was a choice

through a point beside a line · how many parallels?

Euclid's fifth: exactly one. For two thousand years everybody failed to prove it. Bolyai and Lobachevsky changed the postulate instead. Many, or none — nothing breaks. Unprovable because *independent*: a choice, not a fact. Gödel's shape, a century early.

And then the notation knew first. Riemann built curved-space geometry for nothing in particular; Einstein could not have written relativity without it. Mendeleev left holes in his table and named elements nobody had seen. Dirac's equation had a solution nobody wanted; the positron turned up in 1932.

Working memory holds about four things — and what counts as a thing is up to you.

F B I C I A N A S A U S B

13 items · read once, look away, repeat

FBI CIA NASA USB

4 items · the same 13 letters, in the same order

Nothing on the page changed. The difference is entirely in the reader: a chunk is one symbol of a language you already own, and the gain came from the meaning side. A chess master shown a real position for five seconds puts nearly every piece back; shown a random board, barely better than a beginner.

The four slots never grow. What grows is what one slot can mean. That is what growing up is, and what a discipline is, run on one person.

THE QUESTION, REPLACED

Stop asking whether it is intelligent.

UNDECIDABLE · UNPRODUCTIVE

Is it intelligent? Does it truly understand? Is it conscious? Will it always behave? Each asks for a semantic verdict on a system, in a language with no semantics for the words used.

DECIDABLE · TODAY

What language did I state my requirement in? What are its semantics? What is my metric, and how will it be gamed? What is the budget? Who checks the result, from outside?

The right question is never about the machine's *essence*. It is about your language, your semantics, your metric, your check. This is why the method is future-proof: it never mentions a model, a vendor, or a year. The same five questions worked for the steam engine and the spreadsheet.

TAKE THIS WITH YOU

Six habits with no expiry date.

1. **Name the language.** State what you want in something with a semantics — a type, a schema, a unit, a test, an acceptance criterion. English negotiates; formality commits.
2. **Attach a metric, and name its failure.** If you cannot say what "better" means, you are wandering. Then write down how the metric will be gamed — it will be.
3. **Delegate generation. Own verification.** Accept work you could not have produced; never work you cannot check. That is the only shape delegation has ever had.
4. **Budget time, space, and energy.** Know how big your space is and how much of it you can afford to visit. Most doomed projects were doomed arithmetically on day one.
5. **Break every loop from outside.** Nothing certifies itself — not a model, not a company, not you. Import an independent check: an experiment, a proof, a person allowed to say no.
6. **Trust the unproven, then formalise it.** When answers get hard to check, what is missing is a language, not more effort. Take the truth you can see but not yet prove — and build the notation that proves it.

THE DEFINITION

So — what is intelligence?

WORKING DEFINITION

> *Intelligence is developing new formal languages — or at least new properties in existing ones — which requires discovering and understanding promising unproven truth. New languages and properties let us ask new questions about that truth, and then answer them in proofs. Forever.*

DEVELOPING THE SKILLS TO ANSWER

Proving, computing, measuring, building. Rigorous, cumulative, teachable — this is what an undergraduate programme gives you, and what this semester was.

ASKING THE RIGHT QUESTIONS

Choosing which unproven truth is worth formalising. Not deducible, not teachable by rote — this is what advanced, graduate-level study is for.

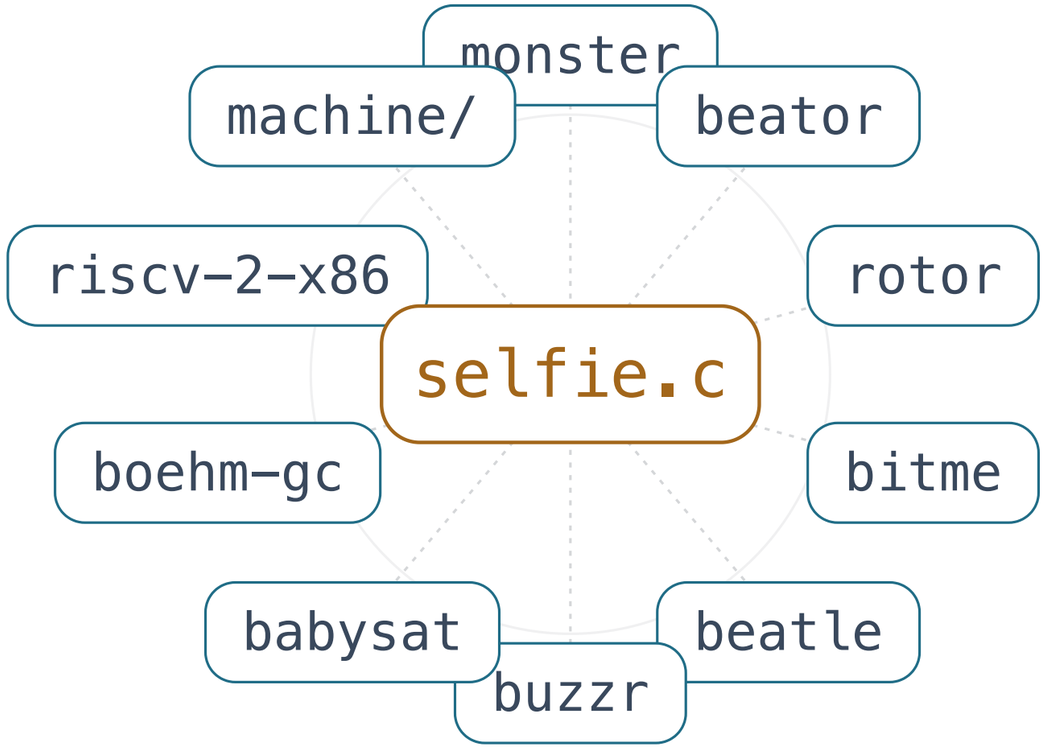
Not a threshold, a score, or a possession — an activity with a direction and no terminating condition. And notice where the difficulty sits: not in the proving, which machines do superbly, but in the finding, which nothing yet does reliably.

So — what is computer science?

WORKING DEFINITION

> *Computer science is the exact study of notation a machine can execute — what can be written down, what can be computed, what can be decided, and what can be afforded — and therefore of the gap, measured precisely, between notation and meaning.*

Not about computers. The machine is the instrument; the theorems hold for a statute, a genome, a score, a model. **Not finishable.** No final language, no complete system, no self-certificate, no decision procedure for meaning — by its own theorems.



the specimen, and the workshop around it

WHY DEPTH STILL MATTERS

This is exactly why you study a field in depth.

Not because AI is unavailable — it is extravagantly available, and it will get better every year of your career. But discovering and understanding promising unproven truth requires having lived inside a subject long enough to feel where it is thin, where it is wrong, and where it is about to give.

No summary, no search, and no generated answer transfers that. It is built, slowly, and it is yours. It is the four slots pointing at ten thousand patterns.

AND IT NEED NOT BE COMPUTER SCIENCE

Every field has unproven truth in front of it — history, medicine, ecology, music theory, economics, law. The method is field-independent; the depth has to be specific. Pick the field whose unsolved problems still make you *strangely comfortable*.

REFRAME · 5 OF 5

Truth can only be approximated. The approximating never stops.

WHAT WE LOST

Completeness. Certainty. A final theory, a final language, a decision procedure for meaning, and any machine that could hand us all of it.

WHAT WE HAVE INSTEAD

Work that cannot be finished, and therefore cannot be taken away: an unbounded frontier where every new notation makes yesterday's unreachable truths reachable, on every single day, forever.

The halting problem, applied to progress, gives the most hopeful result in science: this computation does not terminate.

THE EXAM · AND THE END

Whatever intelligence is, humour is the only way.

The exam draws on the recommended exercises of the fourteen weeks and on nothing else. Every question has the shape of one of them: place a number on the axis, build a diagonal, say what a self-check proves, sort text from behaviour, name the outside check.

For most of you, this semester has probably confirmed your conviction never to study computer science. Or the exact opposite. Outcome A: confirmed conviction. Outcome B: informed confusion. Regardless of which one you leave with: this is about becoming who you are and no one else.

> *Denn es ist zuletzt doch nur der Geist, der jede Technik lebendig macht.*
>
> GOETHE · FOR IN THE END IT IS ONLY THE GEIST THAT BRINGS ANY TECHNIQUE TO LIFE