

The Vocabulary of Knowledge & Reasoning

Twelve terms we will use every week — and a first program that **reasons** instead of printing.

Prof. Dr. Mostafa Elhosseini

Professor of Smart Systems Engineering
<https://elhosseini.com/>

TODAY'S CHAIN

Information



Knowledge



Proposition · Belief



Representation



Knowledge Base



Reasoning · Entailment



Prolog

One question, the whole course

How does a computer **think** — rather than merely execute?

NOT THIS

A program that does exactly what it was told, and nothing more.

THIS

A system that understands something about the world — and uses that understanding to decide.

الكورس ده مش عن السرعة ومش عن الكود — ده عن طريقة التفكير نفسها.

Roadmap for today

I

From information to belief

Information, knowledge, proposition, declarative sentence, belief

II

Representation and the KB

Symbols, formal encoding, the knowledge base

III

Reasoning and its limits

Inference, entailment, incomplete reasoning, the role of logic

IV

Prolog: the words become code

Two programs — one that prints, one that reasons

Every term today reappears in every later lecture. This is the shared language of the course.

I

From Information to Belief

Five terms, one thermometer reading.

Information

Data that has been given **context and a label** — but not yet interpreted.

EXAMPLE

Temperature = 40°C

We know **what the number refers to**.

We do **not** know what it means.

Information tells us what — not so what.

Knowledge

An agent's understanding of the world, expressed as **propositions it takes to be true** . Knowledge is about **judgments**, not raw data.

40°C is a high temperature.

High temperature causes discomfort.

Now the system understands **significance**, not just value.

Knowledge explains the meaning of information.

المعرفة يعني: معلومة + فهم معناها + نستخدمها صح.

Facts the system is willing to use

Knowledge here means **facts about the world** — things the system accepts and is ready to reason with.

A bird flies.

العصفور يطير

If there is fire, there is smoke.

لو في نار ← في دخان

These are not numbers. They are **logical facts**.

دي مش أرقام — دي حقايق منطقية.

Proposition

A statement about the world that can be **true or false**.

MEASUREMENT

The temperature is 40°C.

CLASSIFICATION

40°C is hot.

CONSEQUENCE

Hot weather is dangerous.

Each proposition represents a **possible state of the world**.

القضية جملة ينفع نقول عليها صح أو غلط.

Declarative sentence

A sentence used to **express a proposition**.

DECLARATIVE ✓

The temperature is high.

States a fact. Can be true or false.

QUESTION ✗

What is the temperature?

Asks. Cannot be true or false.

COMMAND ✗

Close the window.

Orders. Cannot be true or false.

Knowledge is written as **declarative sentences**.

Belief

A proposition an agent **accepts as true** — whether or not it is correct.

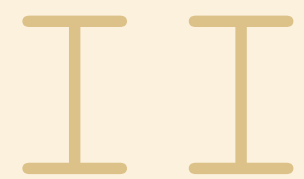
THE SYSTEM BELIEVES

“40°C is hot.”

WHY IT MATTERS

Even imperfect beliefs guide decisions — so a wrong belief produces a wrong action, confidently.

Remember: beliefs drive behaviour.



Representation & the Knowledge Base

In what form does knowledge live inside a machine?

Representation

Using **symbols** to stand for real-world concepts and facts.

Symbolic

Written in signs the machine can manipulate.

Explicit

Stated openly, not buried in procedure.

Structured = facts are arranged/linked (via rules or relationships) so the system can *connect the dots* between them — not just recall each one in isolation.

Structured

Arranged so that reasoning can happen.

```
parent(ali, sara).  
parent(sara, omar).
```

```
grandparent(X, Z) :- parent(X, Y), parent(Y, Z).
```

Without structure, you just have two isolated facts: "Ali is Sara's parent" and "Sara is Omar's parent."

With structure (the **grandparent** rule linking them), the system can now **derive**:

grandparent(ali, omar) — a fact it was never directly told, but can reason out because the facts were **arranged to connect**.

Now the system can **reason**, not just store.

Representation makes meaning computable.

Same clothes, different wardrobe

CLOTHES IN A HEAP

Data

هدوم مرمية = داتا

HUNG AND SORTED

Represented knowledge

هدوم متعلقة ومرتبة = معرفة ممثلة

Representation is not merely putting knowledge somewhere — it is putting it there **in a shape that has meaning** .

نرتّب المعرفة بطريقة الكمبيوتر يعرف يفكرّ بينها — مش نخطها وخلاص.

Knowledge representation

Encoding beliefs using a **formal symbolic language**.

```
HighTemp(x) → Dangerous(x)
```

```
HighTemp(40)
```

The first line is a **rule** — it holds for every x .

The second is a **fact** — it grounds one case.

Meaning is now **explicit and manipulable**.

We encode meaning so machines can think with it.

The Knowledge Base (KB)

A structured collection of **explicitly represented** knowledge.

KB

Temp(40)

HighTemp(40)

HighTemp(x) $\rightarrow$ Dangerous(x)

Dangerous(x) $\rightarrow$ AvoidOutdoorActivity(x)

WHAT A KB IS NOT

It does not tell the system **what to do**.

It tells the system **what is true**.

زي كشكول فيه كل اللي النظام يعرفه.

The KB is the system's memory of meaning.

Memorising vs understanding

PROGRAM WITHOUT A KB

Like a student who memorised the solution to one problem.

زي واحد حافظ حلّ مسألة.

KNOWLEDGE-BASED SYSTEM

Like a student who understood the rule — and can solve any problem like it.

زي واحد فاهم القاعدة ويحل أي مسألة شبيهها.

Generalisation is the visible difference between storing and knowing.

III

Reasoning & Its Limits

Inference, entailment — and why perfect reasoning is not always the rational goal.

Reasoning

The process of deriving **new knowledge from existing knowledge**.

Deriving **implicit** facts from explicit ones

Answering questions **never directly stored**

Detecting **inconsistencies**

40°C is high.

High temperature is dangerous.

Therefore → 40°C is dangerous.

الاستدلال = تشغيل المخ: نطلع نتيجة من حاجات معروفة.

Understanding what was never said

All students who passed are happy.

Ahmed passed.

Therefore: Ahmed is happy.

“كل الطلبة اللي نجحوا مبسوطين”

“أحمد نجح”

أقدر أستنتج: “أحمد مبسوط”

Nobody supplied that sentence. It is not new information — it is **thinking**.

Logical inference

Reasoning where conclusions **necessarily follow** from premises.

HighTemp (40)

HighTemp (x) $\rightarrow$ Dangerous (x)

Dangerous (40)

If the premises are true, the conclusion **must** be true.

Inference guarantees correctness.

Logical entailment

A proposition **p** is entailed by **S** if p is true in **every situation** where S is true.

If all high temperatures are dangerous,
and 40°C is high,
then 40°C is dangerous.

$$S \models p$$

Inference is the **procedure**. Entailment is the **guarantee**.

Entailment means “cannot be false.”

Reasoning does not have to be perfect

Reasoning that **fails to derive some valid conclusions** is still useful reasoning.

IT CONCLUDES

“It is dangerous.”

IT MISSES

“Increase water intake.”

Soundness and completeness are **ideals**.

Real systems reason incompletely or **defeasibly**.

Sometimes **not** reasoning fully is the rational choice.

THIS OPENS LATER

Defaults

Uncertainty

Non-monotonic reasoning

Knowledge-based system

A system whose behaviour depends on **reasoning over an explicit knowledge base**.

THE TEST — BOTH MUST HOLD

Its behaviour depends on reasoning over a KB.

Removing a fact **changes what it can conclude**.

A calculator gives the right answer and stays silent when you ask why. That is not a knowledge-based system.

النظام الذكي يجد يبقى عنده معرفة ويستخدمها في الاستدلال.

Logic: carefully, not religiously

LOGIC PROVIDES

Semantics — what statements mean.

Entailment — what follows from what.

FIRST-ORDER LOGIC IS

A starting point — not a religion.

We will meet problems it cannot state, and extend beyond it.

Use logic as a **tool for precision** — and know where its edges are.

IV

Prolog: The Words Become Code

Two programs. The same question. Only one of them knows anything.

Prolog in one slide

A logic-based language: you state what is **true**, then ask questions, and the system tries to **prove** the answer.

01

You write facts and rules

02

Then you ask questions

03

**Prolog proves the answer
using logic**

Instead of saying **how** to do something, you say **what is true**.

Program 1 — printing colours

```
printColor(snow) :- !, write("It's white").
```

If the input is snow, print “It's white” and stop.

```
printColor(grass) :- !, write("It's green").
```

If the input is grass, print “It's green”.

```
printColor(sky) :- !, write("It's yellow").
```

Wrong on purpose — and the program cannot tell.

```
printColor(X) :- write("Beats me").
```

X matches anything else: if nothing matched, say “I don't know”.

Symbol	Name	Meaning	Example
<code>:-</code>	"if" / neck operator	Head is true if body is true	<code>printColor(sky) :- write("blue").</code>
<code>;</code>	OR (disjunction)	Either side can be true	<code>color(X,blue) ; color(X,red).</code>
<code>!</code>	Cut	Stop backtracking; commit to this choice	<code>printColor(sky) :- !, write("y").</code>

What is wrong with Program 1?

Fixing "yellow" → "blue" doesn't help.

Why? Because nothing was ever *stored* as knowledge — right or wrong.

`write("blue")` is still just text glued to a print command, not a fact the program *knows*.

Correctness ≠ Intelligence. A program that memorizes 100 right answers is just as unintelligent as one that memorizes 100 wrong ones.

WHAT IS MISSING

There is **no knowledge**.

There are **no facts**.

Only if-this-then-print-that.

SO THE PROGRAM CANNOT

Know what **colour** means

Know any **relationships**

Generalise to a new object

Explain **why**

Program 2 — facts and a rule

```
color(snow, white).
```

A fact: snow is white. This is knowledge, not behaviour.

```
madeof(grass, vegetation).
```

A fact: grass is made of vegetation.

```
color(vegetation, green).
```

A fact: vegetation is green.

```
color(X,Y) :- madeof(X,Z), color(Z,Y).
```

A rule: if X is made of Z and Z has colour Y, then X has colour Y. **This is reasoning.**

```
printColor(X) :- color(X,Y), !, write("It's "),  
write(Y), write(".").
```

If the colour of X can be worked out, print it. Behaviour now depends on knowledge.

```
printColor(X) :- write("Beats me.").
```

If it cannot be reasoned out, say “I don't know”.

It was never told — it figured it out

```
?- printColor(grass) .
```

grass is made of vegetation



vegetation is green



therefore grass is green

The system was **never told directly** that grass is green.

NEXT LECTURE

Propositional logic: syntax, semantics, and how such proofs are found **systematically**.