Module 1: Introduction to AI & Intelligent Agents

What is AI? What is an agent? How do we formulate problems? — All the foundations you need.

PECST522 — Artificial Intelligence | B.Tech CSE S5 (2024 Scheme)

Module 1 — Roadmap

1.1	**What is Artificial Intelligence?**	4 approaches, history, foundations
1.2	**Agents and Environments**	Sensors, actuators, agent function vs program
1.3	**Rationality**	Performance measure, rational ≠ omniscient
1.4	**PEAS Description**	Task environment specification
1.5	**Environment Properties**	6 dimensions of classification
1.6	**Agent Architectures**	4 types + learning agent
1.7–1.9	**Problem Formulation**	5 components, vacuum world, 8-puzzle, 8-queens

8 contact hours | Maps to CO1 | Difficulty: Easy–Medium

1.1

What Is Artificial Intelligence?

Four ways to define AI — and why we pick one.

The Four Approaches to AI

	HUMAN-LIKE	RATIONAL / IDEAL
THINKING (cognitive process)	Think like a human (Cognitive Science)	Think rationally (Logic, Laws of Thought)
ACTING (external behaviour)	Act like a human (Turing Test)	Act rationally ← OUR FOCUS (maximise performance)

1. Thinking Humanly
Mimic brain processes (cognitive science)

2. Thinking Rationally
Use formal logic — always correct conclusions

3. Acting Humanly
Turing Test — behave indistinguishably from a human

4. Acting Rationally ★
Maximise goal achievement given what is known

One-liner: AI = building rational agents that perceive their environment and act to maximise their performance measure.

Why "Acting Rationally" Is Our Definition

You don't need to copy humans

Humans make mistakes. A rational agent can outperform humans by making better decisions.

It's measurable

We can define a performance measure and check if the agent maximises it.

It's general

Works for any task — driving, diagnosis, chess — without requiring human-like behaviour.

It's the course framework

Every algorithm in this course (search, logic, RL) is a rational agent doing something.

This is the textbook's primary definition and the one KTU uses

Brief History of AI

1943	McCulloch & Pitts neuron model
1950	Turing — "Computing Machinery and Intelligence"
1956	Dartmouth Workshop — AI named as a field
1966	ELIZA — first chatbot
1969	Shakey — first reasoning robot
1980s	Expert Systems boom (MYCIN, R1)
1997	Deep Blue beats Kasparov at chess
2016	AlphaGo beats Lee Sedol (deep RL)
2022+	ChatGPT / Large Language Models

Timeline — key milestones only

Foundations of AI — Interdisciplinary Roots

Philosophy

Can machines think?

Mathematics

Logic, probability, computation

Economics

Utility, decision theory, game theory

Neuroscience

How the brain works

Psychology

How humans think & learn

Linguistics

Natural language understanding

Computer Engg.

Hardware to run AI

Control Theory

Feedback & optimisation

AI = where philosophy meets engineering — "What should an ideal agent do?" meets "How do we build it?"

AI sits at the intersection of many fields

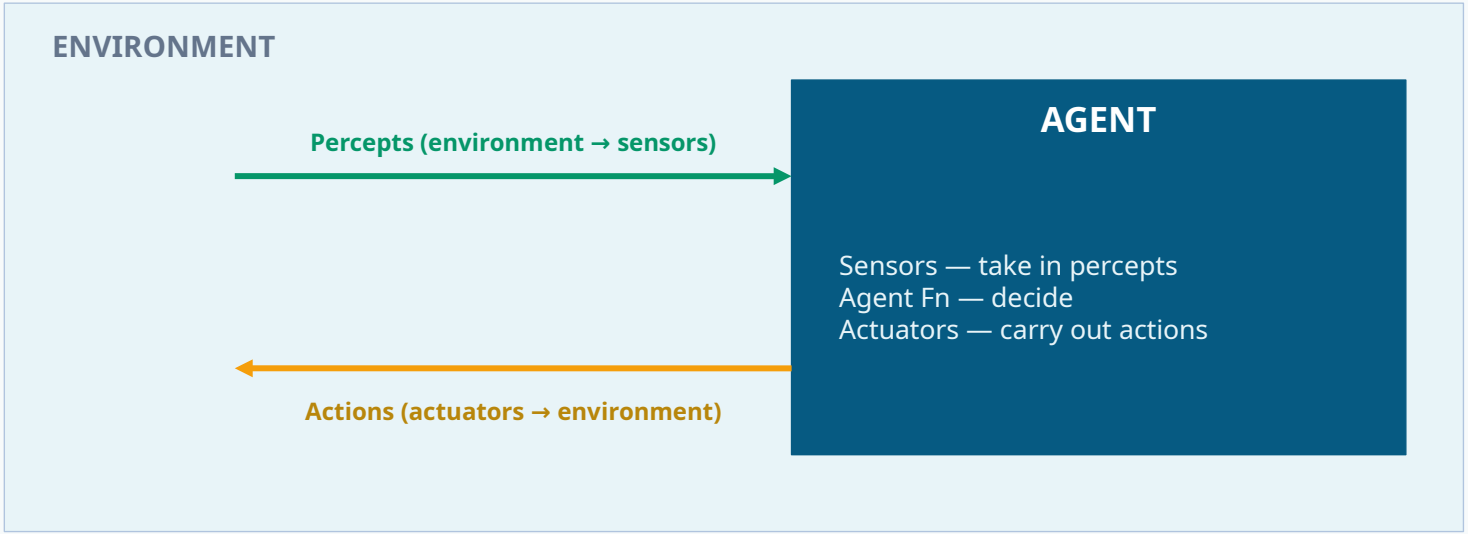
1.2

Agents and Environments

The core building block of the entire course.

What Is an Agent?

An agent is anything that perceives its environment through sensors and acts upon it through actuators.



Examples: Human (eyes→hands) | Robot vacuum (IR/bump→wheels) | Self-driving car (camera→steering) | Thermostat (sensor→heater)

Definition — memorise this

Agent Function vs Agent Program

Agent Function (Abstract)

Mathematical mapping:

$f: P^* \rightarrow A$

Maps entire percept history to an action.

This is the IDEAL — what the agent SHOULD do.

Like a complete rulebook.

Agent Program (Concrete)

Actual implementation:

Running on hardware

The program implements (or approximates) the agent function.

Limited by sensors, compute, time.

Like the actual player following the rulebook.

Abstract specification vs concrete implementation

1.3

The Concept of Rationality

What does it mean to "do the right thing"?

Rational Agent — Definition

A rational agent selects the action that is expected to maximise its performance measure, given:
1. The performance measure 2. Prior knowledge 3. Available actions 4. Percept sequence to date

Rational ≠ Omniscient

Omniscient: Knows the actual outcome (impossible)

Rational: Makes the best decision given what it KNOWS right now

Example: A driver who crosses carefully but gets hit by a falling satellite was still rational — correct decision with available info.

Performance Measure

Defined EXTERNALLY by the designer.

Measures WHAT we want, not HOW.

Vacuum example:

 "Dirt cleaned in 8 hours"

 "Number of times it moves" (would just pace around!)

⚡ EXAM: Define rational agent. Is it omniscient?

1.4

PEAS Description

How to specify a task environment before designing an agent.

PEAS — The Four Things You Must Specify

P	**Performance measure**	How do we score success?	Taxi: safety, speed, legal, comfort
E	**Environment**	What's the world like?	Roads, traffic, weather, pedestrians
A	**Actuators**	What can the agent DO?	Steering, brake, accelerator, horn
S	**Sensors**	What can the agent PERCEIVE?	Cameras, LIDAR, GPS, speedometer

Self-Driving Taxi example → Always state PEAS first when "designing an agent for X".

⚡ EXAM: Write PEAS for any given system

PEAS — Practice Examples

Agent	P	E	A	S
Medical Diagnosis	Healthy patient, low cost	Hospital, patients, tests	Display questions, treatments	Keyboard, symptom database
Spam Filter	Correctly classified emails	Email system, users	Flag/move/delete email	Email headers, content
Robot Arm	Parts assembled correctly	Factory floor, conveyor	Joints, gripper	Joint sensors, camera
Internet Shopping	Correct items, best price	Websites, databases	Display, add-to-cart, order	Page content, user input

⚡ Exam Q: [2023, 3 marks] Define PEAS. Give PEAS for a self-driving car.

Try these yourself before looking at the answers

1.5

Environment Properties

6 dimensions to classify any task environment.

The 6 Environment Properties

Observable	Fully vs Partially	Can the agent see EVERYTHING relevant?	Chess = full | Poker = partial
Deterministic	Deterministic vs Stochastic	Same action → same result every time?	8-puzzle = det. | Dice = stoch.
Episodic	Episodic vs Sequential	Are percept episodes independent?	Photo classify = episodic | Chess = seq.
Static	Static vs Dynamic	Does world change while agent thinks?	Crossword = static | Driving = dynamic
Discrete	Discrete vs Continuous	Countable states/actions or smooth?	Chess = discrete | Steering = continuous
Agents	Single vs Multi-agent	Other agents whose behaviour matters?	Solitaire = single | Chess = multi

⚡ EXAM: "Classify environment X by all its properties" — very common

Classifying Common Environments

Environment	Observable	Deterministic	Episodic	Static	Discrete	Agents
Chess	Fully	Deterministic	Sequential	Semi-dynamic	Discrete	Multi (comp.)
Self-driving	Partial	Stochastic	Sequential	Dynamic	Continuous	Multi
8-Puzzle	Fully	Deterministic	Sequential	Static	Discrete	Single
Crossword	Fully	Deterministic	Sequential	Static	Discrete	Single
Medical Dx	Partial	Stochastic	Sequential	Dynamic	Continuous	Single
Vacuum (simple)	Fully	Deterministic	Sequential	Static	Discrete	Single

Hardest combo: partially observable + stochastic + sequential + dynamic + continuous + multi-agent ≈ real driving

⚡ EXAM: Fill in the classification for any environment

1.6

Structure of Agents

Four architectures, each more capable than the last.

Type 1: Simple Reflex Agent

Percept → [Condition-Action Rules] → Action

```
IF dirty THEN suck
IF clean THEN move
IF obstacle THEN stop
```

How it works: Direct percept → action mapping. No memory.

Like: A thermostat, or flinching from heat.

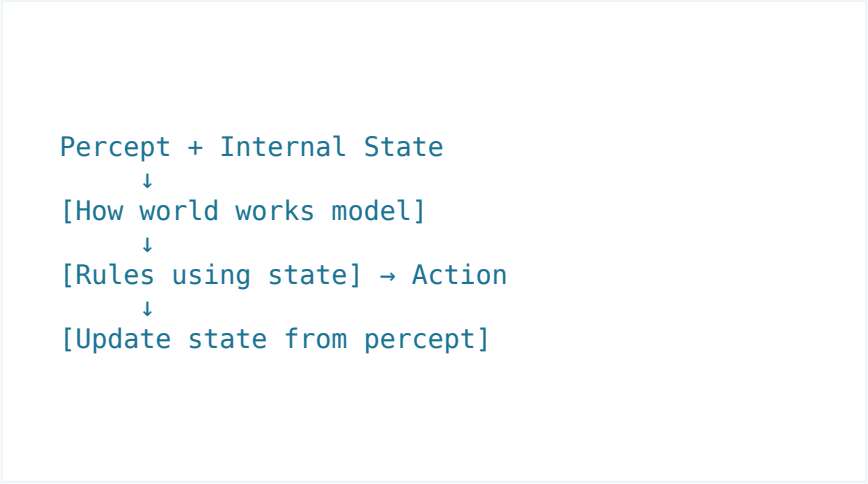
Strength: Very fast, simple.

Weakness: Only works in fully observable environments. Can loop.

Analogy: A cricket player's reflex catch — no thinking, just react to the ball. Works perfectly when you can see everything, fails when there's hidden information.

Condition-action rules only — no memory, no planning

Type 2: Model-Based Reflex Agent



How it works: Tracks things it can't currently see using an internal model of the world.

"Model" = NOT an ML model. It's a world model — how the world evolves + how actions affect it.

Strength: Works in partially observable environments.

Weakness: Still fixed rules — no goals, no optimisation.

Analogy: Navigating your house in the dark — you KNOW where the furniture is from memory (internal state), even though you can't see it right now. A driver remembering a car in the blind spot after looking away.

Maintains internal state — handles partial observability

Type 3: Goal-Based Agent

```
Percept + State + Goals
     ↓
[Search / Planning]
"What happens if I do X?
Will it achieve my goal?"
     ↓
Best action toward goal
```

How it works: Has an explicit goal. Looks ahead — "What will happen if I do X? Will it get me closer?"

This is where SEARCH comes in (Module 2).

 Strength: Flexible — change the goal and behaviour changes automatically.

 Weakness: Can't handle trade-offs between competing goals.

Analogy: Google Maps — knows your destination (goal), explores possible routes (search), picks the best one. Planning a vacation — you have a destination and figure out flights/hotels.

Has explicit goals — uses search and planning (Module 2!)

Type 4: Utility-Based Agent

```
Percept + State
      ↓
[Utility Function]
U(state) = "how happy?"
      ↓
[Maximise expected utility]
      ↓
Best action overall
```

How it works: Has a utility function — a number measuring "how happy" the agent is with a state. Picks actions that maximise EXPECTED utility.

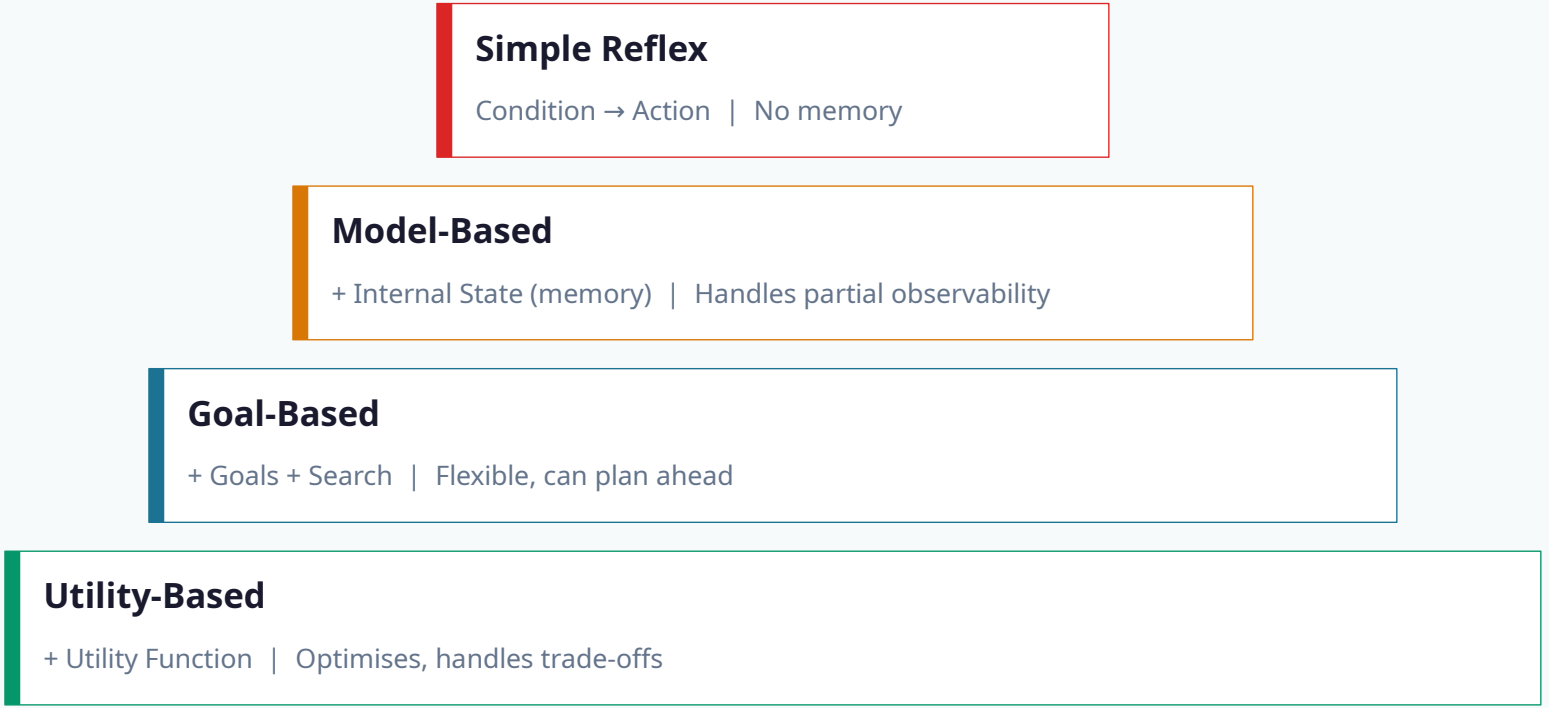
Strength: Handles competing goals, uncertainty, partial success.

Weakness: Computing expected utility can be expensive. Designing the utility function is hard.

Analogy: Choosing a restaurant — you weigh taste, price, distance, wait time (multiple factors) and pick the one that maximises overall satisfaction. A financial planner balancing risk and reward.

Utility function — handles trade-offs and uncertainty

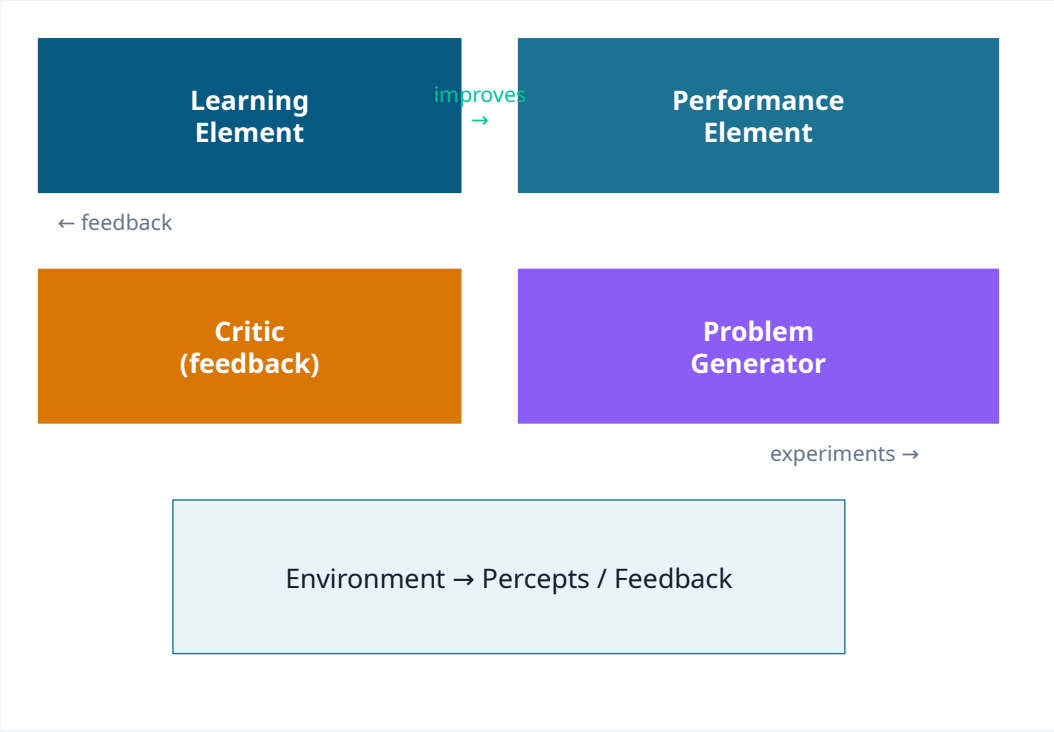
The Agent Architecture Ladder



All 4 can be rational — in environments they can handle. Rationality is the goal; architectures are the means.

Each level adds a capability the previous one lacks

The Learning Agent



4 Components:

1. Learning Element — improves components from experience

2. Critic — evaluates performance (feedback signal)

3. Performance Element — the actual decision-maker (any of the 4 types)

4. Problem Generator — suggests exploratory actions to learn new things

A wrapper around any of the 4 types — improves from experience

1.7

Problem-Solving Agents

How to formally state a problem so a computer can solve it.

5 Components of a Well-Defined Problem

1	**Initial State**	Where do we start?	`Robot in square A, both dirty`
2	**Actions**	What can we do?	`{Left, Right, Suck}`
3	**Transition Model**	What happens after an action?	`RESULT(A-dirty, Suck) = A-clean`
4	**Goal Test**	Are we done?	`Both squares clean`
5	**Path Cost**	How expensive is a path?	`1 per action`

Components 1–3 define the state space. A solution = path from initial to goal. Optimal = lowest path cost.

⚡ EXAM: "State and explain the 5 components" — guaranteed question

Example 1: The Vacuum World

Agent in A or B
Each square: Clean or Dirty

States: 2 locations × 2^2 dirt = 8 states

Actions: {Left, Right, Suck}

Goal: Both squares clean

Path cost: 1 per action

Solution path example:

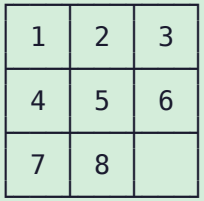
```
[A,dirty,dirty] --Suck--> [A,clean,dirty] --Right--> [B,clean,dirty] --Suck--> [B,clean,clean] ✓ GOAL

[A,dirty,dirty] --Right--> [B,dirty,dirty] --Suck--> [B,dirty,clean] --Left--> [A,dirty,clean] --Suck-->
[A,clean,clean] ✓ GOAL
```

The simplest search problem — 8 states

Example 2: The 8-Puzzle

START

GOAL



States: 9!/2 = 181,440 reachable states

Actions: Move blank {Up, Down, Left, Right}

Transition: Swap blank with adjacent tile

Goal test: Tiles in order 1-2-3 / 4-5-6 / 7-8-blank

⚡ EXAM: Formulate as a well-defined problem

Path cost: 1 per move

Example 3: The 8-Queens Problem

Place 8 queens on an 8×8 board so NO two queens attack each other (same row, column, or diagonal).

Incremental Formulation	Complete-State Formulation
States: 0–8 queens, none attacking	**States:** All 8 queens placed (1 per column)
Initial: Empty board	**Initial:** 8 queens randomly placed
Actions: Place queen in leftmost empty column	**Actions:** Move a queen within its column
Goal: 8 queens, none attacking	**Goal:** Zero pairs attacking
Used by: Systematic search (Module 2)	**Used by:** Local search / min-conflicts (Module 2)

Key insight: Problem formulation matters — different formulations → different solution methods.

⚡ EXAM: Two formulations — incremental vs complete-state

Why Some Problems Are Hard — State Space Size

Problem	State Space	Feasibility
Vacuum (2 sq)	8	Trivial — try everything
8-Puzzle	181,440	Easy for computers
15-Puzzle	~10.5 trillion	Needs good heuristics
8-Queens	92 solutions exist	CSP methods find one fast
Chess	~10^{40}	Too big to search completely
Go	~10^{170}	Needs learning (AlphaGo)

"The vacuum world has 8 states — try everything. Chess has 10^{40} — can't try everything. So we need smart search." → Module 2

This motivates Module 2 — we need smart search for large spaces

Common Doubt: Is This Like an Automaton (ToC)?

Shared Skeleton

Both have:

- Start state
- Set of states
- Transition function
- Accept/goal states
- Both are directed graphs

Key Differences

AI Search adds:

1. Path cost (optimise, not just reach)
2. Massive/infinite state spaces
3. No fixed input string — agent chooses
4. Heuristic strategies (A*, greedy)

Automaton: "Does string reach accept?"
AI: "Cheapest path to goal, found efficiently?"

The formal skeleton is shared — but AI search is different

Module 1 — Summary & Key Takeaways

1. AI = Acting Rationally (build agents that do the right thing)
2. Agent = Sensors → Decision → Actuators (in an Environment)
3. Rational = maximise expected performance given what you know (NOT omniscient)
4. PEAS = Performance, Environment, Actuators, Sensors (always specify first)
5. 6 Environment Properties: Observable, Deterministic, Episodic, Static, Discrete, Agents
6. 4 Agent Types: Simple Reflex → Model-Based → Goal-Based → Utility-Based (+ Learning wrapper)
7. Well-Defined Problem = Initial State + Actions + Transition + Goal + Path Cost
8. Classic examples: Vacuum World (8 states), 8-Puzzle (181K), 8-Queens (2 formulations)

Everything you need for the exam from this module

Module 1 — Exam Preparation

Part A (3 marks each)

- Define rational agent. Is it omniscient?
- Write PEAS for a self-driving car / spam filter / medical diagnosis
- List and describe the 4 agent types
- State the 5 components of a well-defined problem
- Distinguish episodic vs sequential environments
- Classify chess/driving/8-puzzle by all 6 properties

Part B (9 marks each)

- Explain all 4 agent architectures with diagrams
- Formulate 8-puzzle / vacuum world as a search problem (all 5 components)
- Explain the 6 environment properties with examples for each
- Explain the learning agent with a labeled diagram
- Compare incremental vs complete-state formulation of 8-queens

Past questions shown: [2022, 8 marks] 5 components + 8-queens | [2023, 3 marks] PEAS for self-driving car | [2023, 3 marks] episodic vs sequential

Module 1 Complete

Next: Module 2 — Search (Uninformed & Informed, Adversarial Search, CSPs)