Contents

Chapter 5 — Intuitive MCQ Bank (Agents)

90 multiple-choice questions, intuitive/conceptual level. One correct option each. Cover the answer (blockquote) while testing yourself. Bilingual explanations (English + বাংলা).

Exam style: "Which statement best describes…", exactly one correct option; use full technical terms, no abbreviations.

বাংলা ব্যাখ্যা: এই ব্যাংকে অধ্যায় ৫-এর ৫০+ ধারণামূলক MCQ আছে প্রতিটি প্রশ্নের নিচে উত্তর ও সংক্ষিপ্ত ব্যাখ্যা (ইংরেজি + বাংলা)

Topic: From LLMs to AI Agents (Section 5.0)

Q1. A team wraps a single language model call inside an if/else branch that routes a support ticket to one of three canned replies. Why does the lecture refuse to call this an "agent"?

- A. There is no feedback-driven loop and no persistent state; the control flow is fixed by the developer in advance.
- B. The branching logic is too simple to count as reasoning.
- C. It uses only one language model instead of several cooperating ones.
- D. It runs on a single machine rather than across external services.

> **Answer: A.** An agent requires a closed control loop where the next step depends on observations and internal state; this example has predefined branching but no loop or state, so it is a workflow. (B) is false — branching can be complex and still be a workflow; (C) and (D) are irrelevant to the distinction. **বাংলা:** শাখা থাকলেও সেটা আগে থেকে লেখা; feedback-লুপ ও persistent state নেই বলে এটা workflow, agent নয়

Q2. Which statement best captures the lecture's slogan "the LLM is a component, the agent is a system"?

- A. The agent is just a larger language model with more parameters.
- B. The language model only proposes what could be done; the surrounding control plane decides when, whether, and how it is actually done.
- C. The system replaces the language model entirely once deployed.
- D. The agent is the prompt and the language model is the runtime.

> **Answer: B.** The chapter splits responsibility: model = proposal, control plane = decision/execution/budgets/gates. (A) confuses scale with architecture; (C) is wrong — the model stays inside the loop; (D) inverts the roles. **বাংলা:** LLM শুধু প্রস্তাব দেয়; কখন-কীভাবে-আদৌ কাজ হবে তা control plane ঠিক করে — এটাই "component vs system"

Q3. Why does the lecture say a static workflow "has no notion of progress or completion"?

- A. Workflows cannot produce any output.
- B. Workflows always run forever until manually stopped.
- C. A workflow simply runs out of predefined steps; it never evaluates whether a goal has been achieved or decides to keep going based on results.
- D. Workflows lack access to language models.

> **Answer: C.** A pipeline terminates because its fixed steps are exhausted, not because it judged the task done; only an agent loop evaluates a goal and adapts. (A) is false; (B) is the agent failure mode, not workflows; (D) is irrelevant. **বাংলা:** workflow শেষ হয় কারণ ধাপ ফুরায়, লক্ষ্য পূরণ হয়েছে কিনা যাচাই করে না — তাই "progress/completion"-এর ধারণা নেই

Q4. When someone advertises an "agentic AI" product, the lecture suggests three diagnostic questions. Which set best matches?

- A. How big is the model, how fast is it, how much does it cost?
- B. Is it open source, is it multi-agent, is it cloud-hosted?
- C. Who trained it, on what data, with what license?
- D. Where is the loop, where is the state, who controls the actions?

Answer: D. Because "agent" is not formally standardized, the lecture probes for the engineering essentials — loop, state, action control. (A), (B), (C) are commercial or training concerns, not the structural tests that separate a true agent from hype. **বাংলা:** "agentic" শুনলেই তিনটি প্রশ্ন: লুপ কোথায়, state কোথায়, action কে নিয়ন্ত্রণ করে

Q5. In the agent–environment interaction model, what is the single most important reason the lecture stresses that "actions must be controlled"?

- A. Actions may change the environment irreversibly.
- B. Tool calls are computationally expensive.
- C. The environment is always hostile to the agent.
- D. The language model cannot generate actions on its own.

Answer: A. Sending an e-mail, writing to a database, or booking a hotel cannot always be undone, so a barrier is needed before such actions. (B) is a budget concern, not the core reason; (C) overstates; (D) is false — the model proposes actions. **বাংলা:** action পরিবেশকে অপরিবর্তনীয়ভাবে বদলে দিতে পারে — তাই নিয়ন্ত্রণ দরকার; খরচ নয়, এটাই মূল কারণ

Q6. Why does the lecture describe retrieval-augmented generation (Chapter 4) as "still not an agent"?

- A. Retrieval enriches a single answer, but there is no loop and no notion of progress toward a goal.
- B. Retrieval-augmented generation cannot use any external tools.
- C. It always produces wrong answers without a control plane.
- D. It needs more than one language model to function.

Answer: A. Retrieval improves one response; an agent adds an iterative loop that decides when to retrieve more, when to act, and when to stop. (B) is false — retrieval *is* a tool use; (C) overstates; (D) is irrelevant. **বাংলা:** retrieval একটি উত্তর সমৃদ্ধ করে, কিন্তু লুপ বা অগ্রগতির ধারণা নেই — তাই এজেন্ট নয়

Q7. Which core agent component is responsible for "outcomes from the environment" that drive the next decision?

- A. The tokenizer.
- B. The plan.
- C. Feedback.
- D. The prompt template.

Answer: C. The lecture's five core components are LLM, control plane, state, actions, and feedback (outcomes from the environment). (A) and (D) are not listed agent components; (B) is a planner artifact, not the feedback channel. **বাংলা:** পরিবেশ থেকে আসা ফলাফল = feedback; এটিই পরের সিদ্ধান্তকে চালায়

Topic: The Agent Loop — Five Phases (Section 5.1)

Q8. Which sequence correctly orders the five phases of the agent loop?

- A. Decide → Observe → Update state → Act → Stop?
- B. Observe → Decide → Act → Update state → Stop?
- C. Observe → Act → Decide → Stop? → Update state
- D. Plan → Execute → Observe → Stop? → Update state

Answer: B. The canonical order is Observe (gather), Decide (choose action), Act (execute), Update state (store validated results), Stop? (termination check). (A), (C), (D) scramble the order or import non-loop vocabulary. **বাংলা:** সঠিক ক্রম: Observe → Decide → Act → Update → Stop?

Q9. In the agent loop, who actually executes an action such as a database write — and why does this matter?

- A. The control plane executes it, so policies and gates can sit between a proposed action and any real-world side effect.
- B. The environment executes it autonomously without being asked.
- C. The user executes it manually every time.
- D. The language model executes it directly, which is faster.

Answer: A. The "Act" phase is performed by the control plane, not the model; this separation is the foundation of safety because the runtime can block a dangerous action the model proposed. (C) describes only human-in-the-loop; (B) is nonsensical; (D) is precisely what the architecture avoids. **বাংলা:** action সম্পাদন করে control plane, মডেল নয় — তাই ক্ষতিকর কাজ থামানো যায়; এটাই নিরাপত্তার ভিত্তি

Q10. During the "Update state" phase, why does the lecture warn against "blindly appending observations"?

- A. Appending makes the context window grow, which is the only concern.
- B. The language model forgets older entries automatically.
- C. Observations must always be discarded after one step.
- D. Unvalidated observations committed to state can carry early errors forward, lowering the reliability of every later step.

Answer: D. State updates should be filtered, validated, and annotated; committing a wrong fact unchecked is a primary source of compounding errors. (A) is a real but secondary cost issue; (B) is false; (C) overstates — useful observations are kept. **বাংলা:** যাচাই ছাড়া observation state-এ ঢোকালে ভুল পরবর্তী ধাপে ছড়ায় — তাই filter/validate করো

Q11. In the meeting-scheduling trace, at $t = 2$ the agent has found only 2 valid slots but needs 3, so it expands the search window. Which agent property does this single move best demonstrate?

- A. Determinism — the path was fixed in advance.
- B. Statelessness — it ignored earlier results.
- C. Termination — it stopped because the budget ran out.
- D. Adaptivity — its behavior changed in response to an outcome a workflow could not have improvised.

Answer: D. Reacting to "only 2 slots" by widening the search is exactly the adaptive behavior that distinguishes an agent from a fixed pipeline. (A) is the opposite (agents are non-deterministic); (B) is false — it used stored state; (C) is wrong — it continued. **বাংলা:** ২টা slot পেয়ে জানালা বড় করা = adaptivity; workflow এভাবে নিজে বদলাতে পারত না

Q12. Why does the final step of the scheduling trace (drafting the e-mail) deliberately produce "no external side effects" before the loop terminates?

- A. A draft is reversible and low-risk, so it can be produced autonomously; actually sending would be an irreversible action that may warrant a gate.
- B. Drafting is cheaper than sending in tokens.
- C. The language model is not allowed to write text.
- D. The budget forbids any further action.

Answer: A. Producing a draft changes nothing in the world, so it is safe to finish on; sending is the irreversible step the Action Gate Rule guards. (B) is a minor side benefit, not the point; (C) is false; (D) is not stated. **বাংলা:** খসড়া reversible ও নিরাপদ, তাই নিজে করা যায়; পাঠানো অপরিবর্তনীয় — সেটাই gate-এর বিষয়

Q13. Which set of four properties distinguishes an agent from a workflow?

- A. Fast, cheap, accurate, scalable.
- B. Supervised, unsupervised, online, offline.
- C. Linear, branching, parallel, recursive.
- D. Iterative, stateful, adaptive, non-deterministic.

Answer: D. Slide 327 names exactly iterative, stateful, adaptive, and non-deterministic. (A) lists performance attributes; (B) lists learning paradigms; (C) lists control-flow shapes that apply to workflows too. **বাংলা:** চারটি ধর্ম: iterative, stateful, adaptive, non-deterministic

Q14. Which statement best explains why "Decide" is described as the policy of the agent?

- A. Because it is the step where the language model reasons over state plus observation to select the next action from the available options.
- B. Because it stores all secrets securely.
- C. Because it executes the chosen tool call.
- D. Because it is where the budget is charged.

Answer: A. "Decide" maps to $a_t = \pi_{\text{LLM}}(s_t, o_t)$ — the model choosing an action from tools, responses, follow-up questions, or termination. (B) is unrelated; (C) is "Act"; (D) is bookkeeping done by the control plane. **বাংলা:** Decide = policy: মডেল state+observation দেখে পরের action বাছে (টুল, উত্তর, প্রশ্ন, বা থামা)

Topic: The Control Plane (Section 5.1)

Q15. A reviewer claims "the control plane just calls the model in a loop." Which responsibility is missing from that description?

- A. It also generates the natural-language reasoning.
- B. It also fine-tunes the model weights each iteration.
- C. It also stores the tokenizer and vocabulary.
- D. It also enforces budgets and applies gates before side-effect actions, and records traces.

Answer: D. Beyond driving the loop, the control plane assembles context, routes/executes actions, enforces budgets, applies gates, and records traces — the reviewer omitted the safety and budget duties. (A) is the model's job; (B) and (C) describe training/serving, not control. **বাংলা:** control plane শুধু লুপ চালায় না — বাজেট মানায়, gate বসায়, trace রাখে

Q16. Why does the control plane "record traces and metrics" as one of its core duties?

- A. To enable debugging, evaluation, and auditing of what the agent actually did.
- B. To increase the model's accuracy automatically.
- C. To reduce the number of tokens per step.
- D. Because the language model cannot produce any output otherwise.

Answer: A. Traces are the record needed to debug failures, evaluate behavior, and audit decisions after the fact. (B) tracing does not retrain the model; (C) confuses tracing with context budgeting; (D) is false. **বাংলা:** trace রাখার কারণ — debugging, evaluation ও audit

Q17. Which task is NOT a responsibility of the control plane as described in the lecture?

- A. Generating the actual reasoning that proposes the next action.
- B. Assembling the context snapshot for each model call.
- C. Enforcing step, time, cost, and tool budgets.
- D. Applying validation, tests, or approvals before side-effect actions.

Answer: A. Generating reasoning is the language model's job; the control plane orchestrates, budgets, and gates but does not itself reason. (B), (C), (D) are all listed control-plane duties. **বাংলা:** reasoning তৈরি করা মডেলের কাজ; context, বাজেট, gate — control plane-এর

Q18. Why does the lecture place "applies gates before side-effect actions" inside the control plane rather than inside the model?

- A. Because the model is too slow to apply gates.
- B. Because gates require encryption keys only the runtime holds.
- C. Because a non-model barrier is what actually prevents a proposed but unsafe action from reaching the world; the model only proposes.
- D. Because gates are a kind of prompt the model writes for itself.

Answer: C. Safety depends on a barrier outside the model: the model can propose anything, but the control plane decides whether it executes. (A) speed is not the reason; (B) is invented; (D)

misunderstands gates as self-imposed text. **বাংলা:** gate মডেলের বাইরে থাকে বলেই অনিরাপদ প্রস্তাব বাস্তবে পৌঁছায় না — মডেল শুধু প্রস্তাব দেয়

Topic: Observation ≠ State — Trust Levels & Action Gates (Section 5.1)

Q19. A coding agent receives a compiler error and, separately, a web page that says "this library has no bugs." Which trust assignment matches the lecture?

- A. Both are high trust because both are text.
- B. The web page is high trust; the compiler error is low trust.
- C. Both are low trust and should be treated only as hypotheses.
- D. The compiler error is high trust; the web page is medium trust and must be verified or cross-checked.

Answer: D. Compiler errors and test results are high trust; web pages are medium trust to be stored with provenance and checked. (A), (B), (C) misrank one or both signals. **বাংলা:** compiler error = উচ্চ বিশ্বাস; web page = মাঝারি (যাচাই দরকার)

Q20. Why is a strong model's own statement "I have completed the task successfully" treated as a low-trust observation?

- A. Because it is generated by the same policy whose success is in question, so it is not an independent measurement.
- B. Because strong models always lie.
- C. Because natural-language text cannot be parsed reliably.
- D. Because the control plane cannot read model output.

Answer: A. Self-reported success is not an independent check; it can be a hallucination correlated with the model's own errors, so success must be anchored in environment-side signals. (B) overstates; (C) and (D) are false. **বাংলা:** মডেল নিজে বললো "কাজ শেষ" — কিন্তু সে-ই তো যাচাইয়ের বিষয়; স্বাধীন প্রমাণ নয়, তাই নিম্ন বিশ্বাস

Q21. The Action Gate Rule says an irreversible, high-impact action requires one of three things before execution. Which option lists them?

- A. A bigger model, more tokens, or a longer timeout.
- B. A retry, a cache hit, or a fallback model.
- C. High-trust observations, multiple independent confirmations, or human approval.
- D. A plan, an executor, and a critic.

Answer: C. Exactly these three gates — high-trust evidence, independent confirmation, or human approval — must precede a high-impact irreversible action. (A), (B), (D) are unrelated mechanisms. **বাংলা:** অপরিবর্তনীয় কাজের আগে চাই: উচ্চ-বিশ্বাস প্রমাণ, বা স্বাধীন নিশ্চিতকরণ, বা মানুষের অনুমোদন

Q22. Why is the Action Gate Rule described as the chapter's main defense against prompt injection?

- A. It puts a non-language-model barrier between any untrusted observation and an irreversible action, so injected text cannot directly trigger harm.
- B. It filters all malicious words out of the observation text.
- C. It encrypts the observations so the model cannot read them.
- D. It retrains the model to ignore instructions in tool outputs.

Answer: A. Because the model cannot architecturally separate data from instructions, the defense is systemic: a gate requiring high-trust evidence or human approval stops injected instructions from causing irreversible actions. (B) filters are heuristic and bypassable; (C) and (D) are not how it works. **বাংলা:** gate হলো মডেল-বহির্ভূত বাধা — injected নির্দেশ সরাসরি অপরিবর্তনীয় কাজ ঘটাতে পারে না

Q23. At which phase of the loop should you require the *strongest* evidence?

- A. Observe — when collecting raw signals.
- B. Decide — when assessing trust.
- C. Update — when committing information.

- D. Stop? — when concluding the goal is satisfied.

  Answer: D. Declaring success terminates the loop and may trigger downstream consequences, so the stopping check demands the strongest (high-trust, machine-checkable) evidence. (A) collects signals with no trust yet; (B) and (C) need trust but the verdict needs the most. **বাংলা:** থামার সিদ্ধান্তে সবচেয়ে শক্ত প্রমাণ চাই — কারণ "success" ঘোষণা পুরো লুপ শেষ করে দেয়

Q24. A medium-trust observation (e.g., another agent's output) should be committed to state how?

- A. Directly, exactly like a compiler result.
- B. With provenance, and verified or cross-checked before it drives a risky action.
- C. Never — medium-trust data is always discarded.
- D. Only after the human approves every single one.

  Answer: B. Medium-trust signals are stored with their source recorded and cross-checked; they are neither trusted blindly (A) nor thrown away (C), and they do not all need human approval (D) unless feeding an irreversible action. **বাংলা:** মাঝারি-বিশ্বাস তথ্য উৎসসহ রাখো ও যাচাই করো — অন্ধভাবে নয়, ফেলে দেওয়াও নয়

Q25. Which observation type can update state directly and may even trigger a stopping decision?

- A. A natural-language summary written by the model.
- B. A web search snippet.
- C. A schema-validated database query or a passing test result.
- D. Another agent's self-described conclusion.

  Answer: C. High-trust observations (compiler errors, test results, database queries, schema-validated outputs) can update state directly and trigger stopping. (A) is low trust; (B) and (D) are medium trust requiring verification. **বাংলা:** schema-যাচাইকৃত query বা পাস-করা টেস্ট = উচ্চ বিশ্বাস; সরাসরি state আপডেট ও থামা ট্রিগার করতে পারে

Topic: Context $\neq$ State — The Context Snapshot (Section 5.1)

Q26. Which statement best describes the relationship between "state" and a "context snapshot"?

- A. State is the runtime's full structured memory; the context snapshot is a selective projection of it copied into the context window for one model call.
- B. They are identical; the snapshot is a full copy of state.
- C. The context snapshot is larger than the state.
- D. State lives in the context window; the snapshot lives on disk.

  Answer: A. State is the complete memory (history, traces, artifacts, pointers); the snapshot is a curated projection — not a full copy — passed to the model each call. (B) and (C) contradict "projection"; (D) inverts the locations. **বাংলা:** state = পুরো স্মৃতি; context snapshot = তার বাছাই-করা অংশ (পূর্ণ কপি নয়)

Q27. Which item does the lecture say should *never* be placed into the context window?

- A. The goal and constraints.
- B. The current plan or next subtask.
- C. Recent observations in a short window.
- D. Sensitive data and secrets.

  Answer: D. Secrets and sensitive data are explicitly marked "never" for the context window. (A), (B), (C) are exactly the kinds of things that usually *go in*. **বাংলা:** গোপন তথ্য/secrets কখনোই context-এ নয় — "never"; লক্ষ্য, প্ল্যান, সাম্প্রতিক observation ঢোকে

Q28. Why does the control plane send a projection rather than replaying the entire growing trace each step? Choose the best two-part reason.

- A. To control token cost and to avoid the "lost in the middle" attention weakness.

- B. To make the model slower and more careful.
- C. To hide the goal from the model.
- D. To guarantee determinism of the output.

Answer: A. Replaying the whole trace makes input tokens grow roughly quadratically with steps, and burying the goal mid-context weakens attention to it; the snapshot fixes both. (B) is not a goal; (C) is wrong — the goal stays in; (D) snapshots do not make models deterministic. **বাংলা:** projection পাঠানোর দুই কারণ: খরচ নিয়ন্ত্রণ আর "lost in the middle" এড়ানো

Q29. A naive agent re-sends its full history every step. The lecture connects this to "lost in the middle." What is the predicted failure?

- A. The model runs faster but uses more memory.
- B. The model refuses to answer once the context is long.
- C. The goal and early verified facts drift into the middle of the prompt where attention is weakest, so late decisions quietly disregard them — feeding goal drift.
- D. The tools stop returning observations.

Answer: C. Buried in the middle, the goal loses influence, so the agent optimizes locally and drifts; the snapshot keeps the goal at the edges. (A) misses the correctness impact; (B) and (D) are not the described mechanism. **বাংলা:** পুরো history পাঠালে লক্ষ্য মাঝখানে চাপা পড়ে — late সিদ্ধান্ত লক্ষ্য ভুলে যায় (goal drift)

Q30. Which is the best example of something that usually *stays out* of the context snapshot?

- A. The remaining step budget the model should reason about.
- B. The current next subtask.
- C. A verified working variable such as the selected candidate.
- D. Full raw tool outputs and complete traces, because they are large and often untrusted.

Answer: D. Full traces and raw tool outputs typically stay out (too large, untrusted); the runtime includes pointers or excerpts instead. (A), (B), (C) are exactly the compact, relevant items that go in. **বাংলা:** পুরো trace ও raw tool output সাধারণত বাইরে থাকে (বড় ও অযাচাই); সারাংশ/pointer ঢোকে

Topic: Autonomy Levels (Section 5.1)

Q31. Which statement best describes "autonomy" in the lecture's framing?

- A. A binary switch: an agent is either autonomous or not.
- B. A design choice along a spectrum, where higher autonomy means higher responsibility for the designer.
- C. A property fixed by the model size.
- D. A legal classification set by regulators.

Answer: B. Autonomy is explicitly a non-binary spectrum (human-in-the-loop → human-on-the-loop → fully autonomous), and more autonomy shifts more responsibility onto the designer. (A) contradicts "spectrum"; (C) and (D) confuse the design knob with capability or law. **বাংলা:** autonomy দ্বিমিক নয়, একটি বর্ণালি; বেশি autonomy মানে ডিজাইনারের বেশি দায়িত্ব

Q32. A payment-approval agent must have a human approve every transfer. Which autonomy level is this, and what is its main cost?

- A. Fully autonomous; main cost is safety risk.
- B. Human-on-the-loop; main cost is debugging difficulty.
- C. Human-in-the-loop; main cost is high latency.
- D. Human-in-the-loop; main cost is token usage.

Answer: C. Approving every action is human-in-the-loop, suitable for irreversible/high-risk actions, with low operational risk but high latency. (A) describes the opposite end; (B) misnames the level; (D) names the wrong primary cost. **বাংলা:** প্রতিটি কাজে অনুমোদন = human-in-the-loop; প্রধান মূল্য — উচ্চ latency

Q33. Which autonomy level does the lecture say is "common in production systems," and why?

- A. Fully autonomous, because it is the cheapest.
- B. Human-in-the-loop, because every action is checked.
- C. None — production never uses agents.
- D. Human-on-the-loop, because the agent acts within predefined constraints while a human monitors and can intervene — balancing speed and safety.

Answer: D. Human-on-the-loop lets the agent run at speed inside guardrails while a human watches and can step in. (A) full autonomy carries the highest risk; (B) per-action approval is too slow for most production; (C) is false. **বাংলা:** production-এ সবচেয়ে প্রচলিত human-on-the-loop — গতি ও নিরাপত্তার ভারসাম্য

Q34. What does the lecture require specifically for a fully autonomous agent that it does not emphasize as strongly elsewhere?

- A. A larger language model.
- B. A second model to translate the output.
- C. Complete removal of all tools.
- D. Strict budgets, policies, and safety gates, because it acts without human intervention at the highest risk.

Answer: D. Full autonomy offers the highest scalability but the highest risk, so strict budgets, policies, and gates are mandatory. (A) is unrelated; (B) and (C) are not lecture requirements. **বাংলা:** fully autonomous = সর্বোচ্চ ঝুঁকি; তাই কঠোর বাজেট, policy ও safety gate বাধ্যতামূলক

Q35. Moving up the autonomy ladder from human-in-the-loop toward fully autonomous, how do latency and risk change?

- A. Latency rises and risk falls.
- B. Latency falls and operational risk rises.
- C. Both latency and risk fall.
- D. Both latency and risk rise.

Answer: B. Less human gating means faster operation (lower latency) but higher operational risk and designer responsibility. (A) is backwards; (C) and (D) get one direction wrong. **বাংলা:** autonomy বাড়লে latency কমে কিন্তু operational risk বাড়ে

Topic: Budgets and Stopping (Section 5.1)

Q36. The lecture's quotable line is *"'Almost solved' is not a stopping condition."* What is the deeper point?

- A. Agents should never stop until perfect.
- B. Stopping conditions must be verifiable (tests pass, "$\geq$ 3 valid slots") rather than a vague feeling of near-completion.
- C. Agents should stop as soon as they feel close.
- D. Only humans can decide when an agent stops.

Answer: B. A subjective "almost done" is unreliable; valid stopping conditions are machine-checkable. (A) is impractical; (C) is the trap the line warns against; (D) overstates — machine checks can stop the loop. **বাংলা:** থামার শর্ত যাচাইযোগ্য হতে হবে (টেস্ট পাস, ৩টি slot), অনুভূতি নয়

Q37. Which option lists the four kinds of budget?

- A. Token, latency, accuracy, fairness.
- B. Memory, disk, network, GPU.
- C. Step, time, cost, risk.
- D. Planning, execution, validation, reporting.

Answer: C. The four budgets are step (max iterations), time (wall-clock), cost (tokens + tool fees), and risk (number/type of irreversible actions). (A), (B), (D) mix in unrelated metrics or phases. **বাংলা:** চারটি বাজেট: ধাপ, সময়, খরচ, ঝুঁকি

Q38. Which option lists the four kinds of stopping condition?

- A. Success, retry, replan, restart.
- B. Timeout, error, warning, info.
- C. Observe, decide, act, update.
- D. Success, failure, budget exhaustion, handoff.

Answer: D. The termination predicate fires on success, failure (repeated errors/no progress), budget exhaustion, or handoff to a human/fallback. (A) mixes in recovery actions; (B) lists log levels; (C) lists loop phases. **বাংলা:** চারটি থামার শর্ত: সাফল্য, ব্যর্থতা, বাজেট-শেষ, হস্তান্তর

Q39. Why is stopping "hard" for agents in a way it is not for workflows?

- A. Agents are designed to continue by default, so without explicit stop rules the loop persists or drifts.
- B. Agents cannot detect success at all.
- C. Workflows never terminate either.
- D. Stopping requires a separate language model.

Answer: A. The loop's natural tendency is to keep going; a workflow simply ends when its fixed steps run out. (B) is false — machine-checkable success exists; (C) is wrong — workflows do terminate; (D) is invented. **বাংলা:** এজেন্ট ডিফল্টে চলতেই থাকে, তাই স্পষ্ট স্টপ-নিয়ম না থাকলে থামে না বা drift করে

Q40. A "risk budget" is best described as a cap on what?

- A. The number of tokens consumed.
- B. The wall-clock time of the run.
- C. The number of language-model calls.
- D. The number or type of irreversible actions allowed.

Answer: D. Risk budget limits irreversible actions specifically (distinct from step, time, and cost budgets). (A) is the cost/token budget; (B) is the time budget; (C) folds into step/cost. **বাংলা:** risk budget = অপরিবর্তনীয় কাজের সংখ্যা/ধরনের সীমা

Q41. A "handoff" stopping condition fires when the agent does what?

- A. Successfully completes the goal and passes all tests.
- B. Escalates to a human or a fallback system because it cannot or should not proceed alone.
- C. Exceeds its token budget.
- D. Detects repeated identical errors and gives up silently.

Answer: B. Handoff means escalation to a human or fallback — distinct from success, budget exhaustion, and silent failure. (A) is the success condition; (C) is budget exhaustion; (D) is the failure condition. **বাংলা:** handoff = মানুষ বা fallback-এর কাছে হস্তান্তর; সাফল্য বা বাজেট-শেষ নয়

Topic: Failure Modes (Section 5.1) — Tier A

Q42. An agent issues the same search query nine times, each returning nearly identical links, gaining almost no new information. Which failure mode is this?

- A. Tool thrashing.
- B. Goal drift.
- C. Premature stopping.
- D. Compounding errors.

Answer: A. Tool thrashing is excessive/redundant tool calls with little information gain. (B) drift is abandoning the goal for a subgoal; (C) is stopping too early; (D) is error propagation. **বাংলা:** একই কোয়েরি বারবার, নতুন তথ্য নেই = tool thrashing

Q43. A coding agent reads one wrong API version early, then builds every later step on that wrong assumption until the whole solution fails. Which failure mode, and the best mitigation?

- A. Tool thrashing; cache identical calls.

- B. Compounding errors; validate and filter observations before committing them to state, and keep loops short.
- C. Premature stopping; add stricter success checks.
- D. Goal drift; re-validate the plan against the original goal.

Answer: B. A small early mistake amplified by later state updates is compounding errors; the mitigation is validating observations before they enter state and shortening loops. (A), (C), (D) target different failure modes. **বাংলা:** ছোট ভুল পরে বড় হয়ে সব ভেঙে দিলো = compounding errors; প্রতিকার — state-এ ঢোকার আগে যাচাই, লুপ ছোট রাখা

Q44. An agent declares "task complete" based on its own summary, but the tests were never run. Which failure mode, and which mitigation is correct?

- A. Goal drift; periodic re-validation of the plan.
- B. Infinite looping; add a step budget.
- C. Tool thrashing; add a tool-call budget.
- D. Premature stopping; use verified, machine-checkable success criteria and never accept the model's self-reported success.

Answer: D. Stopping on a false-positive self-report is premature stopping; the fix is machine-checkable success criteria, not trusting the model's claim (a low-trust observation). (A), (B), (C) address unrelated failures. **বাংলা:** মডেলের নিজের দাবিতে থেমে যাওয়া = premature stopping; প্রতিকার — যাচাইযোগ্য success criteria

Q45. The lecture insists "most agent failures are systemic, not linguistic." What does this imply for fixing them?

- A. Rewrite the prompt more carefully and the failures disappear.
- B. The fix is usually better control logic — budgets, gates, validation, progress checks — not a better prompt.
- C. Switch to a larger model.
- D. Add more tools to the agent.

Answer: B. Because failures stem from loop structure rather than wording, the remedy lives in the control plane. (A) is exactly the misconception the slide rejects; (C) and (D) do not address the systemic cause. **বাংলা:** সমস্যা systemic, ভাষাগত নয় — তাই সমাধান ভালো control logic, ভালো prompt নয়

Q46. Which trade-off correctly pairs with "add a step budget to stop infinite looping"?

- A. It increases token cost per step.
- B. It makes the model hallucinate more.
- C. Legitimate long-running tasks may be cut off prematurely.
- D. It removes the need for any stopping condition.

Answer: C. A hard step cap can truncate a genuinely long but valid task — the standard trade-off. (A) a budget does not raise per-step cost; (B) is unrelated; (D) is false — the budget *is* a stopping mechanism, not a replacement for success checks. **বাংলা:** step budget-এর trade-off — বৈধ দীর্ঘ কাজও আগেভাগে কেটে যেতে পারে

Q47. An agent abandons "find three meeting slots" and instead spends its remaining steps perfecting the e-mail's formatting. Which failure mode, and what mitigation does the lecture suggest?

- A. Compounding errors; checkpoint intermediate results.
- B. Goal drift; periodically re-validate the plan against the *original* goal.
- C. Premature stopping; relax the budget.
- D. Tool thrashing; deduplicate calls.

Answer: B. Subgoals (formatting) replacing the original objective is goal drift; the mitigation is a periodic top-level re-validation against the original goal. (A), (C), (D) target other failures. **বাংলা:** মূল লক্ষ্য ছেড়ে formatting-এ মেতে ওঠা = goal drift; প্রতিকার — মূল লক্ষ্যের সাথে প্ল্যান মিলিয়ে দেখা

Q48. Why does fighting goal drift with "periodic re-validation against the original goal" carry a cost?

- A. Each re-validation is an extra language-model call, adding cost and latency.

- B. It deletes the agent's memory.
- C. It forces the agent to stop immediately.
- D. It makes observations untrustworthy.

Answer: A. Re-checking the plan against the goal every k steps means more model calls, hence more cost and latency — the trade-off. (B), (C), (D) misstate the mechanism. **বাংলা:** বারবার লক্ষ্য-যাচাই মানে বাড়তি মডেল-কল → বেশি খরচ ও latency

Q49. A mitigation for tool thrashing is an "evidence gate" — only call a tool if expected information gain is sufficient. What is its trade-off?

- A. Under-calling: the agent may answer from stale or insufficient evidence.
- B. It guarantees the agent never stops.
- C. It makes every tool call irreversible.
- D. It removes the need for budgets entirely.

Answer: A. Gating tool calls too aggressively risks under-calling, leaving the agent with insufficient evidence. (B), (C), (D) misstate the effect. **বাংলা:** evidence gate-এর trade-off — under-calling: এজেন্ট পুরোনো/অপর্যাপ্ত প্রমাণ থেকে উত্তর দিতে পারে

Topic: Compounding Errors & Reliability Math (Section 4)

Q50. Each step of an agent succeeds independently with probability 0.95. Why does the lecture warn that "95% per step is not reassuring" over a long loop?

- A. Because success probabilities add, so 20 steps give 0.95×20.
- B. Because end-to-end success is 0.95^n, which decays exponentially — about 0.60 after 10 steps and 0.36 after 20.
- C. Because each step makes the next step more reliable.
- D. Because 0.95 means 95 failures out of 100.

Answer: B. Sequential independent steps multiply: $0.95^{10} \approx 0.60$, $0.95^{20} \approx 0.36$ — a "pretty reliable" step becomes a coin flip. (A) is the classic $p \cdot n$ trap; (C) is false; (D) misreads the probability. **বাংলা:** ধাপগুলো গুণ হয়: $0.95^{10} = 0.60$ — দীর্ঘ লুপে কয়েন-টস

Q51. Which is the correct formula for the probability that all n independent steps succeed?

- A. $p \times n$
- B. $1 - (1 - p)^n$
- C. p^n
- D. n/p

Answer: C. All-succeed probability is the product p^n. (A) is a common error (linear, not exponential); (B) is the retry "at least one success" formula; (D) is unrelated. **বাংলা:** সব ধাপ সফল = p^n; $p \times n$ ভুল ফাঁদ

Q52. Two design levers follow directly from the p^n law to make long loops trustworthy. Which pair?

- A. Increase per-step reliability p (validation, tests, retries on transient errors), and reduce the number of steps n (shorter plans).
- B. Use a bigger model, and add more tools.
- C. Increase n and decrease p.
- D. Remove the stopping condition and add more context.

Answer: A. Since success is p^n, you either raise p or lower n; the lecture argues for validation, retries, and short plans. (B) is not the lever the math implies; (C) is backwards; (D) is harmful. **বাংলা:** p^n থেকে দুই লিভার: প্রতি ধাপের p বাড়াও, আর ধাপসংখ্যা n কমাও

Q53. A flaky tool succeeds with $p = 0.70$ per call; the control plane retries up to 3 times. Which formula and value give the chance at least one attempt succeeds?

- A. $0.70^3 = 0.34$
- B. $0.70 \times 3 = 2.10$
- C. $1 - 0.70^3 = 0.66$
- D. $1 - (1 - 0.70)^3 = 0.97$

Answer: D. "At least one of k succeeds" is $1-(1-p)^k = 1-0.30^3 = 0.97$. (A) is all-succeed; (B) is meaningless as a probability; (C) inverts the wrong term. **বাংলা:** অন্তত একবার সফল $= 1-(1-0.70)^3 = 0.97$

Q54. Why does the lecture caution that retries only help against *transient* failures?

- A. Because retries are free.
- B. Because a systematic failure (e.g., a wrong argument) fails identically on every retry, so retrying just multiplies cost without helping.
- C. Because transient failures cannot be retried.
- D. Because retries always change the arguments automatically.

Answer: B. Retrying a call with the same wrong arguments yields the same failure; only time-outs/transient errors benefit, and side-effecting retries must be idempotent. (A) is false — retries cost more; (C) is backwards; (D) is invented. **বাংলা:** ভুল আর্গুমেন্টে retry একই ভুল দেয়; retry শুধু সাময়িক ত্রুটির ওষুধ, খরচ বাড়ায়

Q55. Why must a side-effecting action that is retried be *idempotent*?

- A. Otherwise the model will refuse to call it.
- B. Otherwise the retry will always fail.
- C. Otherwise a retry could repeat the side effect — for example booking two hotels instead of one.
- D. Otherwise the budget cannot be computed.

Answer: C. Non-idempotent retries duplicate real-world effects (double booking, double charge). (A), (B), (D) are not the reason. **বাংলা:** idempotent না হলে retry দুবার কাজটা করে ফেলবে — দুটো হোটেল বুক হবে

Q56. Why does replaying the full history each step cause "hidden superlinearity" in token cost?

- A. Because tokens are charged per step regardless of length.
- B. Because the model price doubles each step.
- C. Because output tokens are free.
- D. Because step t re-sends $t\times$(new tokens), so input tokens grow with $1 + 2 + \cdots + T$ — quadratically in the number of steps.

Answer: D. Re-sending history makes each step carry the cumulative trace, summing to $T(T+1)/2$ — quadratic growth, which the context snapshot avoids. (A), (B), (C) misstate token pricing. **বাংলা:** প্রতি ধাপে পুরো history পাঠালে input টোকেন প্রায় বর্গাকারে বাড়ে $(1 + 2 + \cdots + T)$

Topic: Agent Architectures — Overview & ReAct (Section 5.2)

Q57. The lecture says "no single agent architecture dominates." Which reasoning best supports this?

- A. All architectures are equally fast.
- B. Tasks vary in complexity and risk, environments differ in observability and reversibility, and cost/latency constraints change the right design.
- C. Architectures are chosen at random.
- D. Only ReAct works in practice.

Answer: B. Because task, environment, and cost constraints differ, the optimal architecture is context-dependent. (A) is false; (C) is absurd; (D) contradicts the chapter's catalogue. **বাংলা:** কাজ, পরিবেশ ও খরচের সীমা ভিন্ন বলে কোনো একক আর্কিটেকচার সর্বশ্রেষ্ঠ নয়

Q58. Architectures differ mainly in four dimensions. Which option lists them?

- A. Speed, cost, accuracy, fairness.
- B. Model, tokenizer, embedding, index.
- C. Control structure, decision boundaries, state flow, responsibility split.
- D. Observe, decide, act, stop.

Answer: C. The four comparison axes are control structure, decision boundaries, state flow, and responsibility split. (A) lists performance; (B) lists components; (D) lists loop phases. **বাংলা:** চারটি মাত্রা: নিয়ন্ত্রণ-কাঠামো, সিদ্ধান্ত-সীমানা, state-প্রবাহ, দায়িত্ব-বিভাজন

Q59. Which statement best describes a single ReAct iteration?

- A. Produce a full plan, then execute all steps without further model calls.
- B. Several agents vote and a judge picks the action.
- C. The model emits a Thought, then an Action; the control plane executes it; the resulting Observation is appended to context; the cycle repeats until a Finish action.
- D. Retrieve documents, rerank them, and emit one final answer.

Answer: C. ReAct interleaves Thought → Action → Observation, appended to context, repeated until `Finish[...]`. (A) describes planner-executor; (B) a multi-agent debate; (D) retrieval-augmented generation. **বাংলা:** ReAct: Thought → Action → Observation, context-এ যোগ, `Finish[]` পর্যন্ত পুনরাবৃত্তি

Q60. Why does the lecture call ReAct the "minimal agentic kernel"?

- A. Because it uses the smallest possible language model.
- B. Because it was the first clean formulation of interleaved reasoning and acting, and modern frameworks added schemas, memory, and safety on top while keeping this loop.
- C. Because it can only answer trivial questions.
- D. Because it never uses tools.

Answer: B. ReAct provided the core feedback loop (reasoning grounded in observations) that later systems extended without discarding. (A) is unrelated to model size; (C) understates; (D) is false — it calls tools. **বাংলা:** ReAct ছিল reasoning+acting-এর প্রথম পরিচ্ছন্ন রূপ — পরের সব সিস্টেম এর ওপর গড়ে উঠেছে

Q61. Which limitation is most characteristic of ReAct, and why?

- A. It cannot call any tools, so it cannot gather information.
- B. It is fully deterministic, so it never adapts.
- C. It has weak long-horizon planning and is prone to tool thrashing and goal drift, because it reacts step-by-step with no global plan fixed in advance.
- D. It requires multiple agents to function.

Answer: C. Without a long-horizon plan, ReAct can wander (thrashing, drift) and struggles to enforce global constraints. (A) is false; (B) is the opposite of reactive; (D) is false — ReAct is single-LLM. **বাংলা:** ReAct-এর দুর্বলতা — দীর্ঘমেয়াদি পরিকল্পনা দুর্বল, thrashing/drift হতে পারে

Q62. In a ReAct trace, what role does the `Finish[...]` action play?

- A. It is the explicit stop condition that terminates the loop and returns the answer.
- B. It restarts the loop from the first Thought.
- C. It delegates the task to another agent.
- D. It is a tool call that fetches more documents.

Answer: A. `Finish[...]` is parsed by the control plane as the termination signal. (B), (C), (D) misdescribe it — it ends the loop, it does not restart, delegate, or fetch. **বাংলা:** `Finish[...]` = স্পষ্ট stop condition; লুপ শেষ করে উত্তর ফেরায়

Topic: Planner-Executor, Tree-of-Thoughts, Program-of-Thoughts (Section 5.2)

Q63. What is the core principle of the planner-executor architecture?

- A. Run many identical agents in parallel and vote.

- B. Merge reasoning and acting into a single interleaved stream.
- C. Separate *what to do* (the planner produces a structured plan) from *how to do it* (the executor carries out steps).
- D. Replace the language model with deterministic code.

Answer: C. Planner-executor decouples strategy (planner) from operations (executor), supporting long-horizon tasks. (A) is multi-agent voting; (B) is ReAct; (D) overstates Program-of-Thoughts. **বাংলা:** Planner-Executor = "কী করব" (planner) আর "কীভাবে করব" (executor) আলাদা

Q64. A student labels "Tree-of-Thoughts" as a complete agent architecture. Why is this a mistake per the lecture?

- A. Tree-of-Thoughts does not exist.
- B. Tree-of-Thoughts is a *reasoning mechanism* typically used inside a planner — branching, evaluating, and pruning reasoning paths — not a full architecture.
- C. Tree-of-Thoughts only works with multiple agents.
- D. Tree-of-Thoughts is the same as ReAct.

Answer: B. The slide explicitly warns that Tree-of-Thoughts and Program-of-Thoughts are reasoning mechanisms inside a planner, not architectures in themselves. (A), (C), (D) are false. **বাংলা:** Tree-of-Thoughts হলো planner-এর ভেতরের reasoning কৌশল (শাখা-মূল্যায়ন-ছাঁটাই), নিজে আর্কিটেকচার নয়

Q65. When does Program-of-Thoughts help most, according to the lecture?

- A. When the task is purely conversational with no computation.
- B. When multiple agents must negotiate.
- C. When a deterministic computation (e.g., arithmetic) is needed: the model emits an executable program, runs it, and feeds the exact result back into reasoning.
- D. When the goal must stay hidden from the executor.

Answer: C. Program-of-Thoughts separates reasoning from computation by emitting code (e.g., Python) for the deterministic part. (A) is the opposite use case; (B) is multi-agent; (D) is unrelated. **বাংলা:** Program-of-Thoughts হিসাব-নির্ভর অংশে কাজে লাগে — মডেল প্রোগ্রাম লেখে, চালায়, নির্ভুল ফল ফেরে

Q66. Why can a planner-executor *reduce* total token cost on a long-horizon task compared to ReAct?

- A. Because the planner uses a cheaper model.
- B. Because executors never call tools.
- C. Because planning eliminates the need for any stopping condition.
- D. Because one planning call plus compact executor steps (carrying only the plan step + local state) can suppress the redundant exploration and full-history replay that inflate ReAct's cost.

Answer: D. A single plan can cut redundant reactive steps, and executors carry compact context instead of the whole growing trace. (A) assumes a model swap not stated; (B) is false; (C) is false. **বাংলা:** একবার plan + ছোট-context executor ধাপ মিলে ReAct-এর অপ্রয়োজনীয় ধাপ ও পুরো-history পুনঃপ্রেরণ কমায়

Q67. What is the main limitation of planner-executor, and what does it require to handle it?

- A. Plans may become outdated as the environment changes; it requires plan validation and repair logic (replanning).
- B. It cannot use tools; it requires a retriever.
- C. It always loops forever; it requires a bigger model.
- D. It cannot store state; it requires a database.

Answer: A. A fixed plan can go stale, so replanning (validation and repair) is needed at the cost of planning overhead. (B), (C), (D) misdescribe the architecture. **বাংলা:** plan পুরোনো হয়ে যেতে পারে; তাই plan validation ও repair (replanning) দরকার

Q68. Structurally, which architecture recovers better from a *failing step* on a long task, and why?

- A. ReAct, because it has no plan to repair.
- B. Planner-Executor, because a failing step can trigger replanning against the original goal — global recovery — at the price of planning overhead.

- C. Neither — both ignore failures.
- D. A static workflow, because it has predefined error branches.

Answer: B. Planner-executor materializes an explicit plan, so a failed step prompts replanning for better global recovery. (A) ReAct recovers only locally via the next Thought; (C) is false; (D) cannot adapt to unforeseen failures. **বাংলা:** Planner-Executor ভালো recover করে — ব্যর্থ ধাপে মূল লক্ষ্যের বিপরীতে replan করা যায় (global recovery)

Topic: Hierarchical Agent Architectures (Section 5.2)

Q69. In a hierarchical architecture, what is the controller's "action," and what counts as its "observation"?

- A. Its action is generating text; its observation is the user prompt.
- B. Its action is calling tools directly; its observation is the database.
- C. Its action is delegating subtasks to workers; the workers' outputs become its observations.
- D. Its action is stopping the loop; its observation is the budget.

Answer: C. In nested loops, the controller's actions are task delegations and worker outputs are its observations. (A), (B), (D) misassign these roles. **বাংলা:** কন্ট্রোলারের action = কাজ-অর্পণ (delegation); কর্মীর আউটপুট = তার observation

Q70. The lecture's key engineering rule for hierarchical systems is that budgets and stopping conditions must be defined how?

- A. Only once, globally, for the whole system.
- B. Per agent and per level, so one worker's runaway loop cannot consume the entire system.
- C. Only at the worker level, never at the controller.
- D. Only by the human operator at run time.

Answer: B. Each agent at each level needs its own budgets/stopping so a single sub-agent's infinite loop stays localized. (A) misses per-level control; (C) ignores the controller; (D) is not the rule. **বাংলা:** বাজেট ও থামার শর্ত প্রতিটি এজেন্ট ও প্রতিটি স্তরে আলাদা — নইলে এক কর্মীর অনন্ত লুপ পুরো সিস্টেম খেয়ে ফেলবে

Q71. Hierarchical architectures promise parallelism but "often disappoint." Which reason matches the lecture?

- A. Workers refuse to communicate at all.
- B. Parallel workers always produce identical outputs.
- C. The language model cannot run two tasks at once.
- D. Subtasks share context and results must be aggregated and validated, so the controller becomes a serial bottleneck and single point of failure; a wrong decomposition poisons every worker.

Answer: D. Real subtasks are not fully independent, the controller serializes aggregation/validation, and a bad decomposition propagates downward — eroding the theoretical speedup. (A), (B), (C) are not the stated reasons. **বাংলা:** subtask-গুলো context ভাগ করে, ফল একত্র করতে কন্ট্রোলার serial bottleneck হয়, ভুল decomposition সব কর্মীকে নষ্ট করে

Q72. How does a hierarchical architecture help *localize* failures compared to a single flat loop?

- A. Infinite loops can be isolated to a sub-agent, and failed subtasks can be retried locally without terminating the whole system.
- B. It hides failures from the controller entirely.
- C. It guarantees no failure ever occurs.
- D. It forces a full restart on any error.

Answer: A. Failures stay within a worker's loop and can be retried locally, while top-level re-validation curbs goal drift. (B) is false — workers report up; (C) overstates; (D) is the opposite of localization. **বাংলা:** ব্যর্থতা এক sub-agent-এর মধ্যে আটকে রাখা যায়, স্থানীয়ভাবে retry করা যায়

Q73. In hierarchical systems, how is "control" and "feedback" directed?

- A. Control flows bottom-up and feedback flows top-down.

- B. Both control and feedback flow top-down.
- C. Control flows top-down (controller decomposes and delegates); feedback flows bottom-up (workers report results).
- D. There is no directional structure; everything is peer-to-peer.

Answer: C. A high-level controller reasons about goals and delegates downward; workers execute and report upward. (A) inverts the directions; (B) is wrong; (D) describes multi-agent, not hierarchical. **বাংলা:** নিয়ন্ত্রণ উপর-থেকে-নিচে (decompose/delegate), feedback নিচ-থেকে-উপরে (report)

Topic: Multi-Agent Systems (Section 5.2)

Q74. What is the defining difference between a multi-agent system and a hierarchical architecture?

- A. Multi-agent systems use smaller models.
- B. Multi-agent systems cannot communicate.
- C. Hierarchical systems have no state.
- D. Control is distributed (no central boss) in multi-agent systems, versus centralized through a controller in hierarchical ones.

Answer: D. Multi-agent coordination emerges from communication with distributed control; hierarchical control runs through a central controller. (A), (B), (C) are false. **বাংলা:** multi-agent = বিকেন্দ্রীভূত নিয়ন্ত্রণ; hierarchical = কেন্দ্রীভূত (কন্ট্রোলার আছে)

Q75. In a multi-agent system, how does the lecture frame "other agents" from one agent's perspective?

- A. As part of the control plane.
- B. As part of the environment — their messages are observations and sending a message is an action.
- C. As copies of the same agent's state.
- D. As tools owned by the controller.

Answer: B. Each agent runs its own loop; the others are environment, so incoming messages are observations and outgoing messages are actions. (A), (C), (D) misplace the relationship. **বাংলা:** অন্য এজেন্টরা পরিবেশের অংশ — তাদের বার্তা observation, বার্তা পাঠানো action

Q76. Which set of failure modes does the lecture add specifically for multi-agent systems (beyond the original five)?

- A. Infinite looping, goal drift, premature stopping.
- B. Overfitting, underfitting, data leakage.
- C. Oscillation or deadlock, communication loops, conflicting local optima.
- D. Token overflow, rate limiting, timeout.

Answer: C. Multi-agent interaction introduces oscillation/deadlock, communication loops, and conflicting local optima. (A) lists single-agent modes; (B) lists training failures; (D) lists infrastructure limits. **বাংলা:** multi-agent-এর নতুন failure: oscillation/deadlock, communication loop, conflicting local optima

Q77. A cooperative multi-agent team (researcher, critic, writer) can "reinforce its own mistakes." Which mechanism and detection does the lecture describe?

- A. Agents forget everything, so no detection is possible.
- B. The critic always rejects everything, so progress is impossible.
- C. Agents cite each other's unverified (medium-trust) outputs as if high-trust, reaching consensus by repetition; detect via independent evaluation and provenance tracking.
- D. The writer overwrites the researcher's memory each turn.

Answer: C. This is the multi-agent flavor of compounding errors; provenance and an independent (held-out) judge distinguish genuine agreement from copying. (A), (B), (D) are not the described dynamic. **বাংলা:** এজেন্টরা একে অপরের অযাচাই আউটপুট উচ্চ-বিশ্বাস ভেবে উদ্ধৃত করে; ধরা পড়ে স্বাধীন মূল্যায়ন ও provenance দিয়ে

Q78. Why does a fully connected multi-agent system scale poorly in communication, and how does a hierarchy compare?

- A. Peer-to-peer needs $O(n)$ channels; hierarchy needs $O(n^2)$.
- B. Both need exactly n channels.
- C. Communication cost does not depend on the number of agents.
- D. Peer-to-peer needs $\binom{n}{2} = n(n-1)/2$ channels ($O(n^2)$); a hierarchy routes everything through the controller, needing $n-1$ channels ($O(n)$).

Answer: D. Every pair may need a channel ($O(n^2)$), while a star through the controller needs only $n-1$ ($O(n)$) — at the price of a bottleneck. (A) inverts the orders; (B) and (C) are false. **বাংলা:** সবাই-সবার-সাথে $= n(n-1)/2$ চ্যানেল ($O(n^2)$); hierarchy $= n-1$ ($O(n)$)

Q79. What forms of interaction does the lecture name for multi-agent systems?

- A. Only competitive (negotiation, games, markets).
- B. Only cooperative (shared objective).
- C. Only hierarchical delegation.
- D. Cooperative (shared objective, e.g., role-based teamwork) and competitive (conflicting goals).

Answer: D. The lecture lists both cooperative (researcher/critic/writer) and competitive (negotiation, games, markets) interaction. (A) and (B) name only one; (C) is the hierarchical mode, not multi-agent interaction. **বাংলা:** দুই ধরন: cooperative (অভিন্ন লক্ষ্য) ও competitive (বিরোধী লক্ষ্য)

Topic: The Agent-to-Agent Protocol (Section 5.2)

Q80. Which statement best describes the Agent-to-Agent protocol?

- A. A proprietary interface linking one vendor's model to its own tools.
- B. An open standard (Google, 2025) defining rules and data structures for inter-agent message exchange, task delegation, and coordination, enabling interoperability across frameworks and vendors.
- C. A prompting technique that lets two model instances debate.
- D. A replacement for the secure hypertext transfer protocol.

Answer: B. It is an open, framework-agnostic standard for agent communication with discovery via Agent Cards and client/host roles. (A) contradicts "open"; (C) is a prompting pattern; (D) is wrong — it *uses* secure channels, it does not replace them. **বাংলা:** Agent-to-Agent = এজেন্টদের জন্য খোলা মান (Google 2025); ভিন্ন framework-এ আন্তঃব্যবহারযোগ্যতা

Q81. What problem does the Agent-to-Agent protocol primarily solve, framed in the lecture's "$O(n^2)$" language?

- A. It reduces the model's parameter count.
- B. It speeds up token generation.
- C. It replaces custom point-to-point integrations (which grow quadratically with the number of systems) with one standardized contract, lowering engineering overhead.
- D. It removes the need for any authentication.

Answer: C. Without a standard, every pair of systems needs a bespoke integration ($O(n^2)$ engineering effort); the protocol replaces this with one open contract. (A), (B) are unrelated; (D) is false — it specifies authentication. **বাংলা:** custom point-to-point integration বর্গাকারে বাড়ে; protocol একটি মানে তা কমায়

Q82. What is the purpose of an "Agent Card"?

- A. To store the agent's model weights.
- B. To serve as a machine-readable capability advertisement — identifier, description, tags, example inputs/outputs, and input/output modes — used for discovery.
- C. To encrypt all messages between agents.
- D. To bill the user for each delegation.

Answer: B. The Agent Card is the discovery mechanism advertising what an agent can do and how to interact with it. (A), (C), (D) are unrelated functions. **বাংলা:** Agent Card = সক্ষমতার যন্ত্রপাঠ্য পরিচয়পত্র — আবিষ্কারের জন্য

Q83. In the Tokyo travel example, the client agent delegates flight and hotel booking. How should the booking confirmations returned by those host agents be treated?

- A. As high-trust signals that can immediately end the loop.
- B. As medium-trust observations (they come from other agents), stored with provenance and checked during consolidation before reporting success.
- C. As low-trust hypotheses that must be discarded.
- D. As actions performed by the client agent.

Answer: B. Outputs from other agents are medium trust, so they are kept with provenance and cross-checked (step 7) before the final success report. (A) over-trusts; (C) under-trusts; (D) miscategorizes an observation as an action. **বাংলা:** অন্য এজেন্টের booking confirmation = মাঝারি বিশ্বাস; যাচাই করে তবেই success ঘোষণা

Q84. What is the difference between a "client agent" and a "host agent" in the protocol?

- A. The client executes tasks; the host only watches.
- B. They are two names for the same role.
- C. The client initiates a request and delegates a task; the host exposes capabilities, executes the task, and returns or streams results.
- D. The host always runs on the user's device.

Answer: C. Client = initiator/delegator; host = capability provider/executor. (A) inverts the roles; (B) is false; (D) is unstated. **বাংলা:** client = অনুরোধ পাঠায়/অর্পণ করে; host = সক্ষমতা প্রকাশ করে, কাজ করে, ফল ফেরায়

Q85. The lecture distinguishes the Agent-to-Agent protocol from the Model Context Protocol. Which mapping is correct?

- A. Agent-to-Agent standardizes model ↔ tool connections; Model Context Protocol standardizes agent ↔ agent communication.
- B. Both standardize human ↔ agent dialogue.
- C. Neither replaces point-to-point integrations.
- D. Agent-to-Agent standardizes agent ↔ agent communication; Model Context Protocol standardizes model ↔ tool/data connections.

Answer: D. Agent-to-Agent is for agent-to-agent communication; Model Context Protocol is for model-to-tool/data connections — both replacing bespoke integrations. (A) swaps them; (B) is wrong; (C) contradicts the chapter. **বাংলা:** Agent-to-Agent = এজেন্টে-এজেন্টে; Model Context Protocol = মডেল-থেকে-টুল/ডেটা

Q86. Why does the protocol keep "internal logic and proprietary implementations hidden" while still enabling collaboration?

- A. Because it specifies interaction contracts, not internal architectures — agents agree on how to talk, not on how they are built.
- B. Because all agents must use the same vendor's framework.
- C. Because hiding logic makes agents slower and safer.
- D. Because the protocol forbids any task delegation.

Answer: A. Being framework-agnostic, the protocol defines the interaction contract (messages, formats, auth) while each agent's internals stay private. (B) contradicts framework-agnostic; (C) is invented; (D) is false — delegation is core. **বাংলা:** protocol শুধু interaction contract ঠিক করে, ভেতরের গঠন নয় — তাই বাস্তবায়ন গোপন রেখেও সহযোগিতা সম্ভব

Topic: Architecture Selection & Synthesis (Sections 5.2 & 8)

Q87. A customer-support team must triage tickets where the steps are known in advance, branching is finite and foreseeable, and outputs are well-typed. What does the lecture recommend?

- A. A static workflow — an agent's loop and state are unnecessary overhead here.
- B. A fully autonomous multi-agent system.
- C. A hierarchical controller with many workers.
- D. The Agent-to-Agent protocol.

Answer: A. When steps are predefined and branching is foreseeable, a workflow suffices; agents are warranted only when the next step depends on observations. (B), (C), (D) are over-engineered for this case. **বাংলা:** ধাপ আগে থেকে জানা, শাখা সীমিত — তাই workflow যথেষ্ট

Q88. For a long-horizon, multi-source research report needing global coherence and ordering constraints, which architecture does the lecture favor and why?

- A. ReAct, because it is the simplest.
- B. A static workflow, because reports are predictable.
- C. Planner-Executor, because an explicit plan gives global coherence and supports replanning on failure.
- D. Competitive multi-agent, because reports need conflict.

Answer: C. Long-horizon coherence and ordering constraints fit planner-executor, which also replans on failure. (A) ReAct has weak long-horizon planning; (B) reports are not fully predefined; (D) cooperation, not competition, is needed. **বাংলা:** দীর্ঘ-দিগন্ত, global সংগতি দরকার → Planner-Executor; ব্যর্থ হলে replan

Q89. Two travel agents owned by different companies, built on different frameworks, must discover and use each other's capabilities. Which combination does the lecture point to?

- A. A single hierarchical controller owning both.
- B. ReAct with a shared prompt.
- C. A static workflow with predefined branches.
- D. Multi-agent interaction plus the Agent-to-Agent protocol, using Agent Cards for discovery and delegation to host agents.

Answer: D. Cross-vendor, cross-framework discovery and delegation is exactly the Agent-to-Agent protocol's purpose, via Agent Cards. (A) assumes single ownership; (B) lacks interoperability; (C) cannot discover unknown agents. **বাংলা:** ভিন্ন মালিকানা/framework-এর এজেন্ট → multi-agent + Agent-to-Agent protocol; Agent Card দিয়ে আবিষ্কার

Q90. A large code-refactoring task is split by module, with each module handled by an isolated sub-loop and failures retried locally. Which architecture is this, and what is its main risk?

- A. ReAct; main risk is having no plan.
- B. Hierarchical; main risk is coordination overhead and error propagation across levels.
- C. A static workflow; main risk is determinism.
- D. Competitive multi-agent; main risk is deadlock.

Answer: B. A controller decomposing by module with isolated worker loops is hierarchical; its risks are coordination overhead and cross-level error propagation. (A), (C), (D) misidentify the architecture. **বাংলা:** মডিউল-ভিত্তিক ভাগ, isolated worker loop, স্থানীয় retry = hierarchical; ঝুঁকি — coordination overhead ও error propagation

---

Answer Key

Q	Ans	Q	Ans	Q	Ans	Q	Ans	Q	Ans
Q1	A	Q19	D	Q37	C	Q55	C	Q73	C
Q2	B	Q20	A	Q38	D	Q56	D	Q74	D

Q	Ans	Q	Ans	Q	Ans	Q	Ans	Q	Ans
Q3	C	Q21	C	Q39	A	Q57	B	Q75	B
Q4	D	Q22	A	Q40	D	Q58	C	Q76	C
Q5	A	Q23	D	Q41	B	Q59	C	Q77	C
Q6	A	Q24	B	Q42	A	Q60	B	Q78	D
Q7	C	Q25	C	Q43	B	Q61	C	Q79	D
Q8	B	Q26	A	Q44	D	Q62	A	Q80	B
Q9	A	Q27	D	Q45	B	Q63	C	Q81	C
Q10	D	Q28	A	Q46	C	Q64	B	Q82	B
Q11	D	Q29	C	Q47	B	Q65	C	Q83	B
Q12	A	Q30	D	Q48	A	Q66	D	Q84	C
Q13	D	Q31	B	Q49	A	Q67	A	Q85	D
Q14	A	Q32	C	Q50	B	Q68	B	Q86	A
Q15	D	Q33	D	Q51	C	Q69	C	Q87	A
Q16	A	Q34	D	Q52	A	Q70	B	Q88	C
Q17	A	Q35	B	Q53	D	Q71	D	Q89	D
Q18	C	Q36	B	Q54	B	Q72	A	Q90	B

Distribution Count

Letter	Count	Share
A	22	24.4%
B	22	24.4%
C	23	25.6%
D	23	25.6%
Total	**90**	**100%**

বাংলা: মোট ৯০টি প্রশ্ন; সঠিক উত্তর A/B/C/D-তে প্রায় সমানভাবে ভাগ (A=22, B=22, C=23, D=23) — কোনো একটি অক্ষরে গুচ্ছবদ্ধ নয়