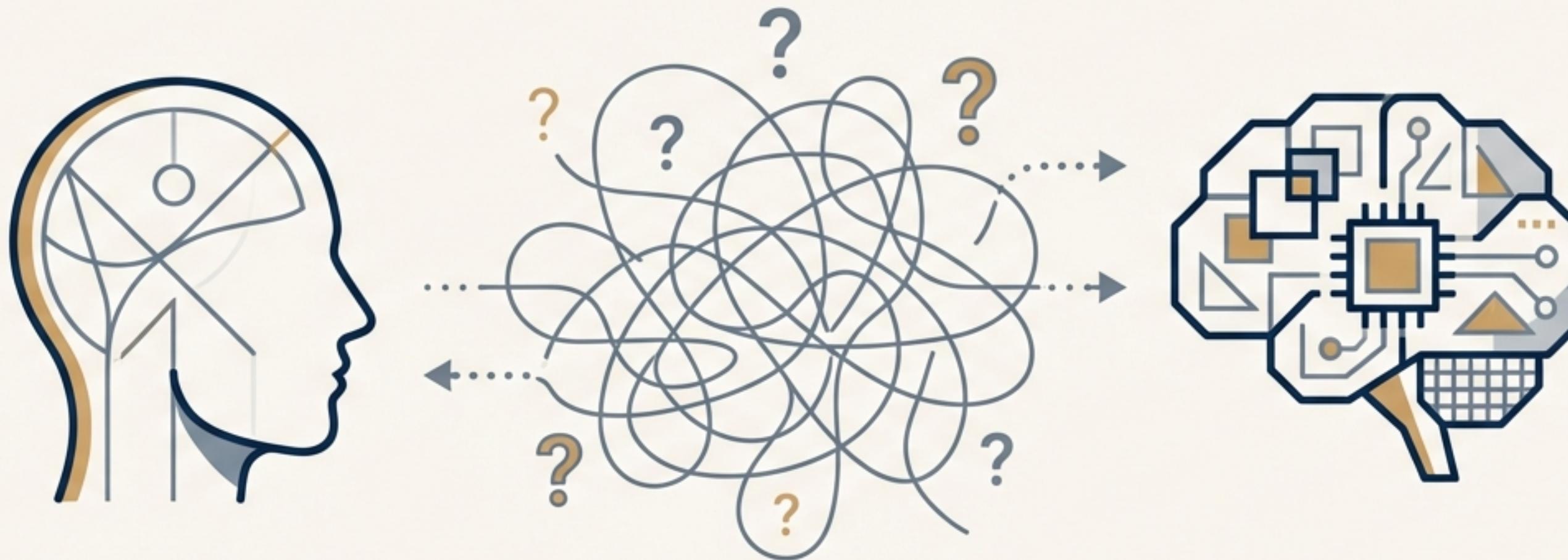


プロンプトエンジニアリングの基礎

なぜAIは指示を無視するのか？ 意図を正しく伝える技術

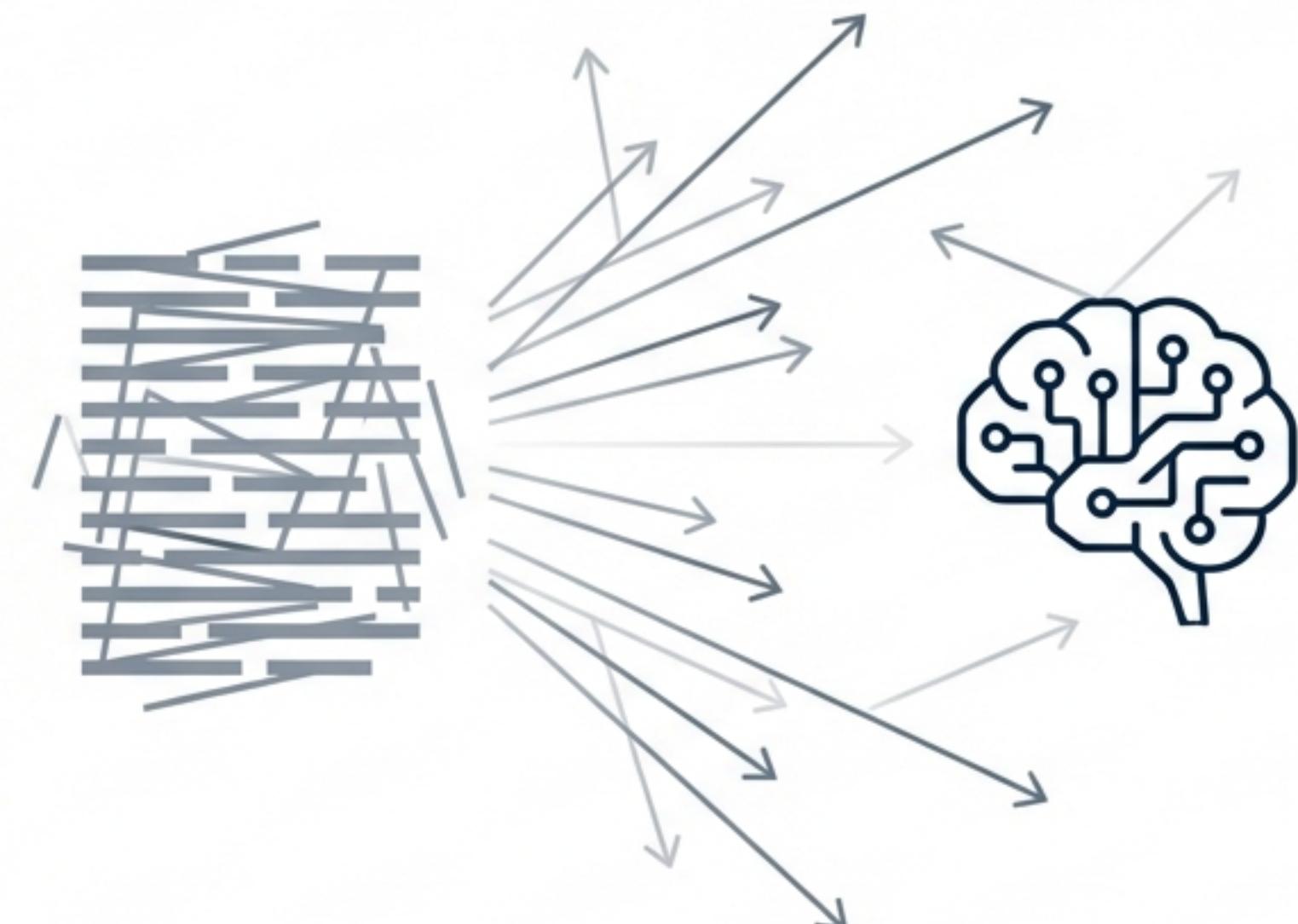


「AIを使ってみたものの、返ってきた内容がトンチンカンで、結局触らなくなってしまった経験はありませんか？」

【課題】なぜAIは指示を無視するのか？

AIは悪気があるわけではない。 原因は、私たちの「指示の出し方」にある。

- **現象**：長文で複雑な指示を出すと、「英語禁止」「200文字以内」といった重要な制約が無視される。
- **原因**：「命令」と「処理対象データ」が混ざった文章だと、AIの**注意（Attention）**が散漫になり、重要な制約を見落してしまう。
- **結論**：この「注意散漫」を防ぎ、意図を正確に伝える技術が「プロンプトエンジニアリング」である。



「注意（Attention）が散漫になる」

【解決策①】構造化（Markdown）で「命令」を明確にする

Before

あなたはプロの編集者です。以下の制約を守って、入力文を要約してください。制約は200文字以内、箇条書きで。入力文はこれです…



「指示とデータが混在した文章」

After

命令
制約
入力文



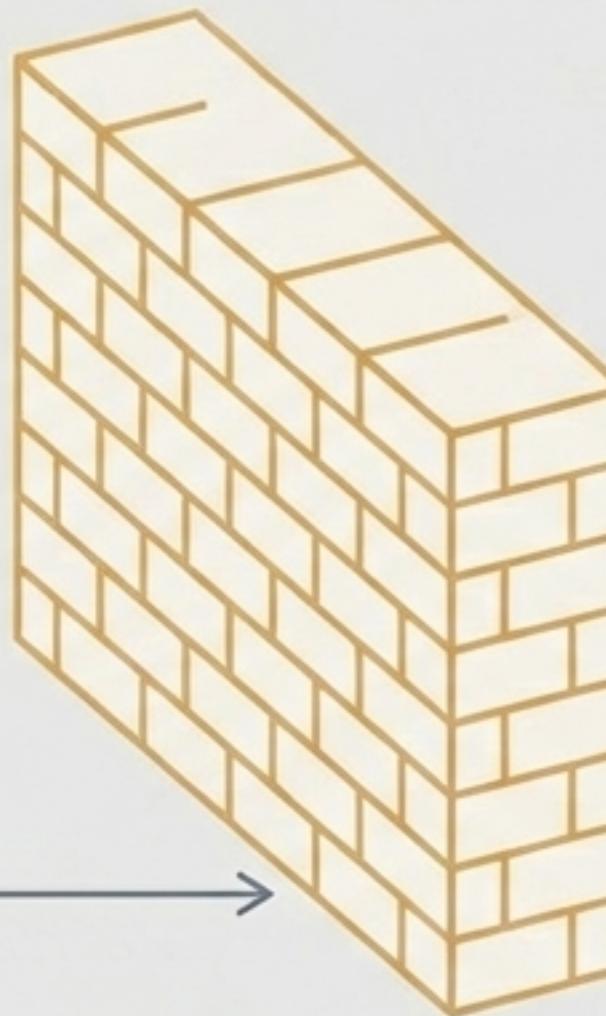
「#」（見出し）を使って情報をブロック分けすることで、AIは情報の「役割」を瞬時に理解し、指示の見落としが激減する。

【解決策②】「データ」の範囲をデリミタで区切る

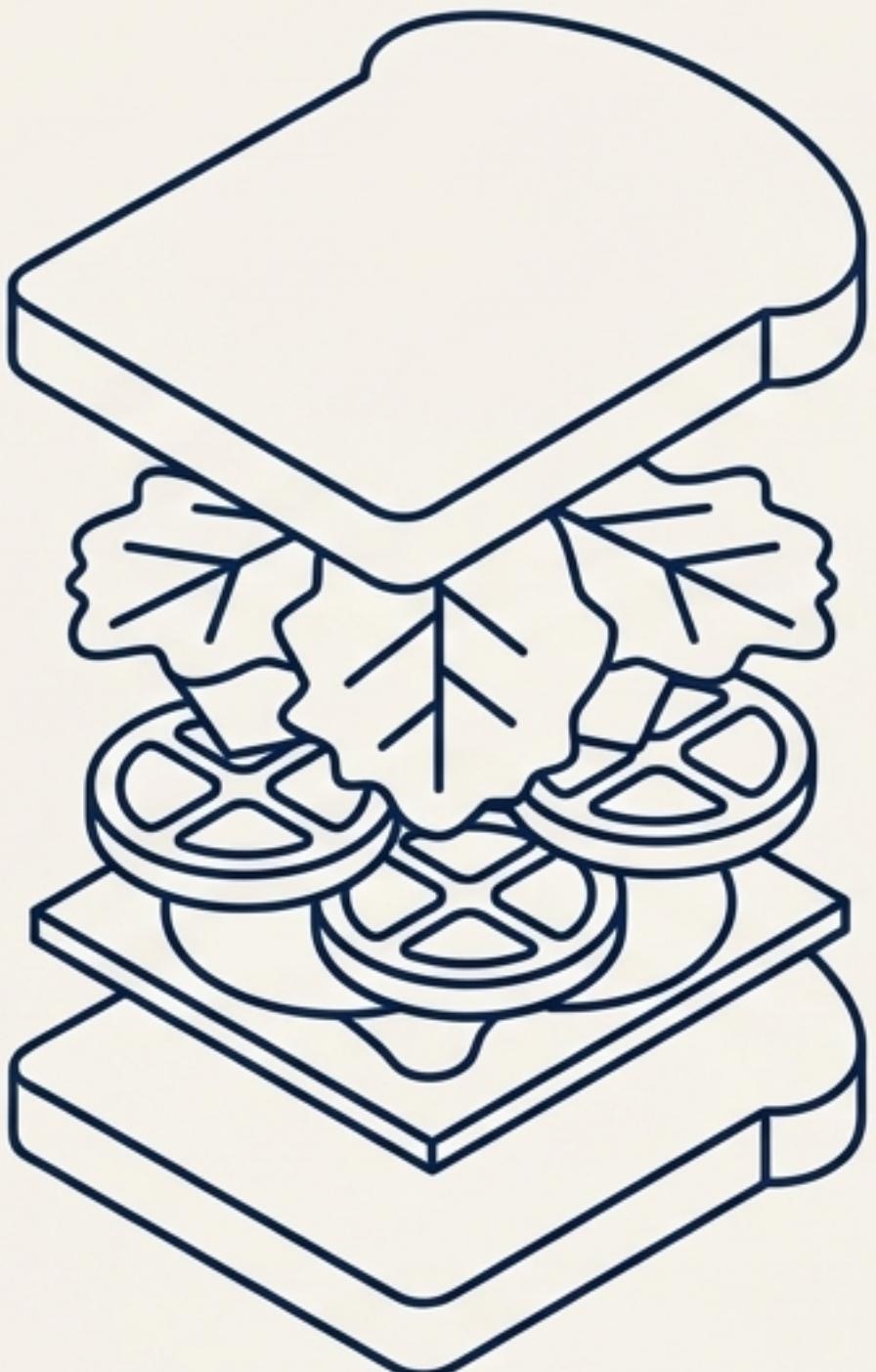
Key Technique: 区切り文字 (Delimiters) の使用

命令
以下のテキストを要約してください。

対象テキスト
"""\<-- ここからデータ (壁を作る) →
(文章の中に「要約」や「禁止」という
言葉が含まれていてもAIは惑わされない)
"""\<-- ここまでデータ →



「```」や「---」で囲むことで、AIに「ここからここまでが“作業対象”」という明確な境界線を示す。データ内の言葉に釣られる誤作動を防ぐ。



「念押しの指示（末尾）」

「指示（冒頭）」

「データ」

Recency Bias

【解決策③】 AIの記憶のクセを利用する 「サンドイッチ話法」

- **テクニック**：重要な指示は「最初」と「最後」の2回伝える。
- **技術的根拠**：大規模言語モデル（LLM）には、プロンプトの末尾にある情報ほど重視する「**Recency Bias**（親近性バイアス）」という特性がある。
(出典: 生成AI運用リスクマネジメント白書)
- **手法**：冒頭で指示し（#制約）、データを挟み、末尾で念押し（#念のため再掲: 英語は禁止です）をする。

【解決策④】言葉より強い「Few-Shot（例示）」

AIは、言葉で説明されるより 「具体例」 を見る方が得意。

「悪い指示」

`テキストから感情を分析して、肯定的か否定的か中立かを判定して、

*(Leads to inconsistent output like
「判定結果：肯定的」)*

「良い指示（Few-Shot）



出力例

入力: この商品は最高だ！ → 出力: Positive

入力: 届いたら壊れていた。 → 出力: Negative

入力: 特にコメントはありません。 → 出力: Neutral

本番の入力

入力: セットアップに少し時間がかったけど、機能には満足している。 → 出力:

理想の「入力と出力のペア」を見せることで、AIは出力フォーマットを正確に模倣する。

【実践】指示無視を防ぐ「構造化テンプレート」

これら4つのテクニックを組み合わせることで、指示の精度は飛躍的に向上する。

Role (役割)

あなたは熟練のアノテーション作業者です。

Constraints (制約)

- 出力はJSON形式のみ。
- 挨拶や余計な説明は一切不要。
- 該当なしの場合は`null`を返すこと。

Few-Shot (出力例)

Input: AはBを買収した。

Output: {"sub": "A", "pred": "買収",
"obj": "B"}

Target Text

Output: AはBを買収した。

Target Text (処理対象)

"""

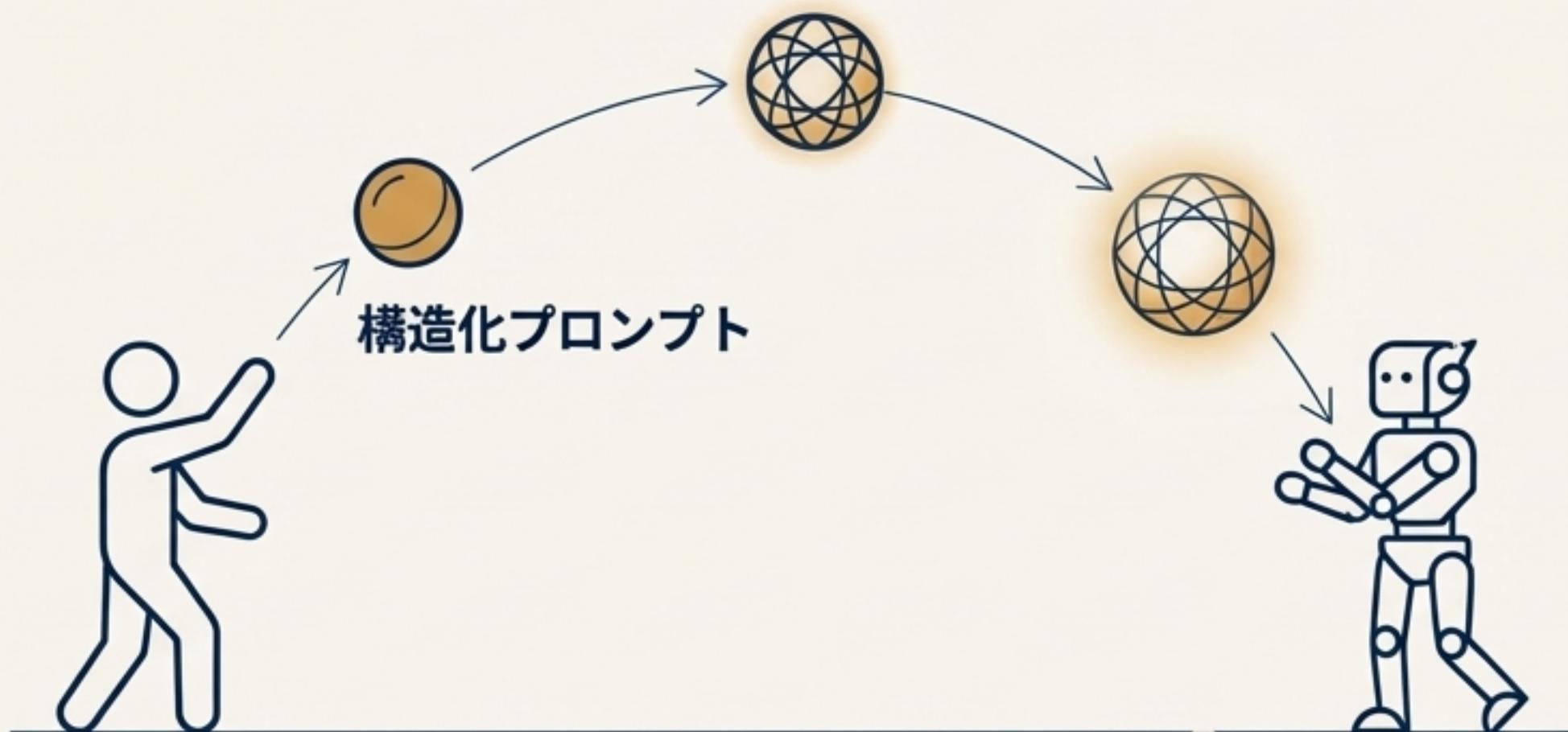
{{ここにテキストを入れる}}

"""

Reminder (念押し)

上記のテキストから情報を抽出してください。
出力は**JSON形式のみ**でお願いします。

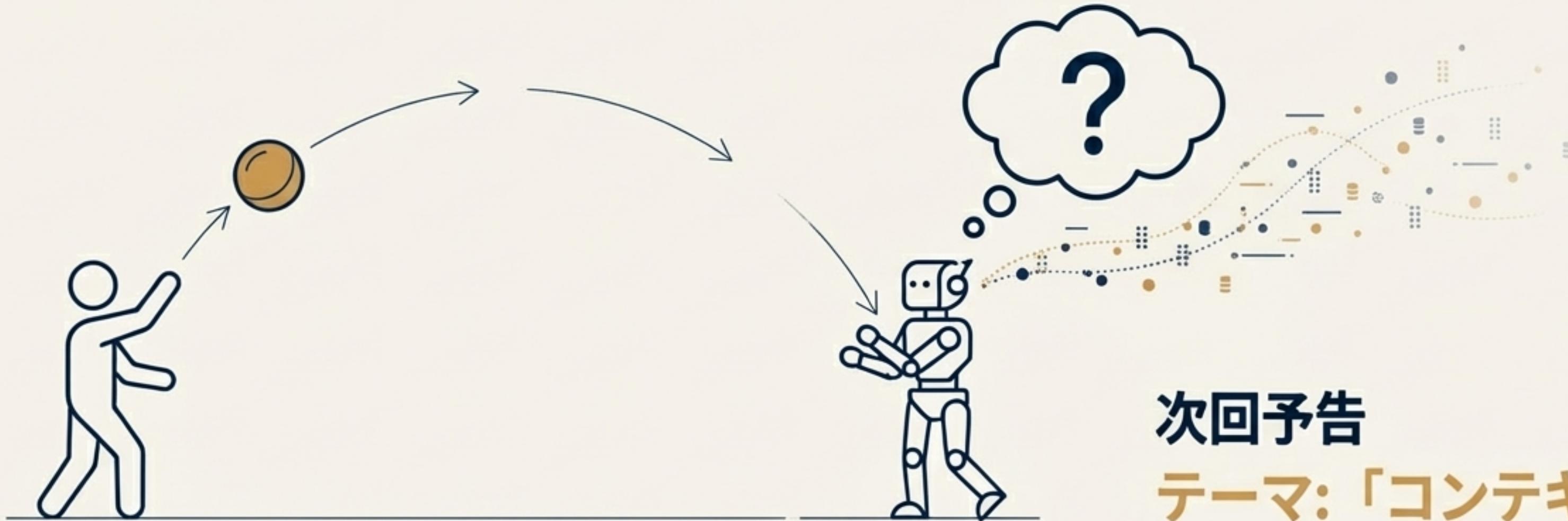
【まとめ】プロンプトは「対話」の始まり



- ・生成AI活用とは、60点の回答に対し、**対話（ラリー）**を通じて精度を上げていくプロセスである。
- ・今回学んだ「構造化」は、その対話をスムーズに行うための**共通言語（基準点）**である。

どんなに完璧なプロンプトを作っても、
一発で100点の回答が来るとは限らない。

【Next Step】対話の「記憶」をどう管理するか？



「では、どう対話（ラリー）を進めればいいのか？」

対話を続けると、AIは前の会話を忘れていく…（記憶の限界）

次回予告

テーマ: 「コンテキストエンジニアリング」

内容: 対話の履歴を管理し、AIの記憶を維持する技術。