Don't Judge a Book by its Cover: Testing LLMs' Robustness Under Logical Obfuscation

Anonymous ACL submission

Abstract

Tasks such as solving arithmetic equations, evaluating truth tables, and completing syllogisms are handled well by large language models (LLMs) in their standard form, but they often fail when the same problems are posed in logically equivalent yet obfuscated formats. To study this vulnerability, we introduce **Logifus**, a structure-preserving logical obfuscation framework, and, utilizing this, we present **LogiQAte**[1], a first-of-its-kind diagnostic benchmark with 1,108 questions across four reasoning tasks: (i) Obfus FOL (first-order logic entailment under equivalence-preserving rewrites), (ii) Obfus Blood Relation (family-graph entailment under indirect relational chains), (iii) Obfus Number Series (pattern induction under symbolic substitutions), and (iv) Obfus Direction Sense (navigation reasoning under altered directions and reference frames). Across all the tasks, evaluating six state-of-the-art models, we find that obfuscation severely degrades zero-shot performance, with performance dropping on average by 47% for GPT-4o, 27% for GPT-5, and 22% for reasoning model, o4-mini. Our findings reveal that current LLMs parse questions without deep understanding, highlighting the urgency of building models that genuinely comprehend and preserve meaning beyond surface form.

1 Introduction

"A large language model can't do deductive reasoning. It's set up to do pattern recognition and react to those patterns", warned Brent Smolinski, IBM's Vice President and Senior Partner for Global AI Strategy (IBM Think, 2024). Complementing this critique, in 2024, MIT researchers demonstrate that generative models "produce impressive outputs without recovering the underlying world model," crashing when simple detours arise in a New York City navigation task (Vafa et al., 2024). Together, these observations underscore a critical question of whether large language models (LLMs) truly *reason* or simply *memorize* and match patterns learned from their training data.

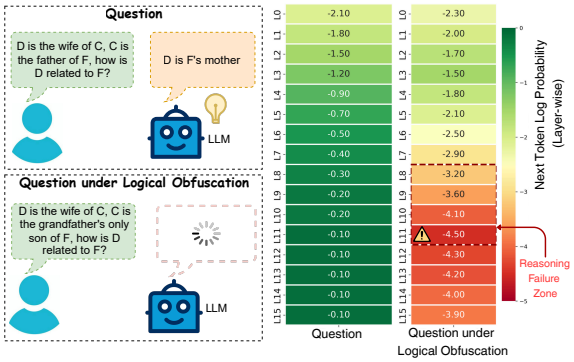

Figure 1: **An Overview of LLMs under Logical Obfuscation with Logifus.** On the left, an obfuscated version of a question, while remaining *logically equivalent*, poses significant challenge to the model. On the right, we track the model's confidence in predicting the correct next word or token, quantified as the layer-wise next-token log probability assigned to that token at each step. Under logical obfuscation, these layer-wise scores drop sharply as the question is processed, indicating a transition into a reasoning-failure zone (cf. Section 5.1). The plot shows this degradation across the first 15 layers of LLaMA 3.1 8B (Touvron et al., 2023).

Logical reasoning, the ability to derive new facts from known ones, solve puzzles, and adapt to novel situations, has anchored AI research since its inception. Early symbolic systems (McCarthy, 1959; Weizenbaum, 1966; Winograd, 1971; Colmerauer and Roussel, 1973; Shortliffe, 1976) pioneered rule-based, logic-driven approaches; more recently, chain-of-thought (CoT) prompting has enabled LLMs to articulate intermediate steps, improving performance on arithmetic, commonsense, and symbolic reasoning tasks (Wei et al., 2022b). Yet, the question of genuine reasoning persists. State-of-the-art systems such as GPT-5 (OpenAI, 2025a), Claude 4 (Anthropic, 2025), and o4 (OpenAI, 2025c) demon-

[1]Dataset available at https://tinyurl.com/nw8487zj

strate impressive capabilities, but it remains unclear whether they construct and verify logical chains or merely exploit memorized patterns. Standard benchmarks (MMLU (Hendrycks et al., 2021), BIG-Bench Hard (Srivastava et al., 2022), ARC (Clark et al., 2018)) typically score only the final answer, allowing high performance via surface recall. Empirically, models often falter when equivalent questions are rephrased or obfuscated, underscoring the gap between pattern matching and robust reasoning (Xie et al., 2025).

To understand reasoning beyond memorization and to check if the model is truly understanding the problem question, we explore *logical obfuscation*, i.e., rewriting a problem into a different surface form while preserving its logical structure. As shown in Fig. 1, in a simple family-relation puzzle, the sentence *"D is the wife of C, C is the father of F"* can be obfuscated as *"D is the wife of C, C is the grandfather's only son of F"*. Both describe the same relationship, yet an LLM that answers "D is F's mother" in the original question struggles when presented with the obfuscated version. Building on this idea, we present **LogiQAte**, the first-of-its-kind benchmark of 1,108 questions, built using **Logifus**, a proposed, structure-preserving obfuscation algorithm that systematically generates logically equivalent but syntactically diverse versions of each question. LogiQAte spans four tasks: **(i) Obfus FOL**, which evaluates first-order logic entailment under equivalence-preserving rewrites (e.g., replacing "If it rains, the ground gets wet" with "The ground does not stay dry whenever it rains"), **(ii) Obfus Blood Relation**, which tests family-graph reasoning under obfuscation with indirect kinship chains and alternative terms (cf. Fig 1), **(iii) Obfus Number Series**, which requires inductive pattern detection in numerical sequences obfuscated via symbolic substitutions and arithmetic transformations (e.g., the base sequence $2, 4, 6, 8,\ ?$ rewritten as $n, n+2, n+4, n+6,\ ?$), and **(iv) Obfus Direction Sense**, where navigation puzzles are rephrased with altered directions and frames of reference (e.g., replacing "turn left and walk 3 steps" with "move westward three paces").

Across six state-of-the-art models, we observe a consistent average performance drop of 28.8% under logical obfuscation, computed across all tasks and averaged over zero-shot, few-shot, and CoT settings (cf. Fig. 2). To understand this brittleness, we examine *why* performance degrades. Memorization tests show detection rates rising from 50% on original to 82% on obfuscated questions, indicating heavy reliance on memorization rather than reasoning. Layer-wise next-token log-probability analysis further reveals a 50–80% collapse in token-level confidence in late layers, critical for reasoning, relative to base questions (cf. Section 5.1). The paper makes the following contributions:

1. We introduce **Logifus**, a novel structure-preserving algorithm for generating logically obfuscated questions across four reasoning tasks.
2. We present **LogiQAte**, a first-of-its-kind diagnostic benchmark comprising 1,108 obfuscated reasoning questions spanning four reasoning tasks (cf. Fig. 3), conducting a comprehensive empirical analysis across nine open- and closed-source (general purpose and reasoning-focused) LLMs.

2 Related Works

Research on evaluating the reasoning capabilities of LLMs spans several paradigms. Natural language inference (NLI) benchmarks such as LogicNLI generate large-scale synthetic triplets (Tian et al., 2021), while FOLIO provides 1,430 human-written examples annotated with first-order logic (Han et al., 2022). More complex deductive datasets, including LogicBench (Parmar et al., 2024), ProofWriter (Tafjord et al., 2021), Multi-LogiEval (Patel et al., 2024), and ProntoQA (Madaan and Yazdanbakhsh, 2022), explore varied inference types but typically evaluate on clean formulations. Machine reading comprehension datasets such as LogiQA (Liu et al., 2020), ReClor (Yu et al., 2020), and LingOly (Bean et al., 2024) similarly target reasoning in QA or linguistic puzzles without testing robustness to equivalent reformulations.

Obfuscation Studies. ObfusQAte proposes multi-tier semantic transformations for factual QA and reports frequent hallucinations and failures under these variants (Ghosh et al., 2025). In code generation, MetamorphASM evaluates resilience to obfuscated assembly and source, finding that large code models can reduce superficial complexity yet often break semantics under heavy obfuscation (Mohseni et al., 2025). Together, these works show that obfuscation is a stringent robustness test, but prior efforts center

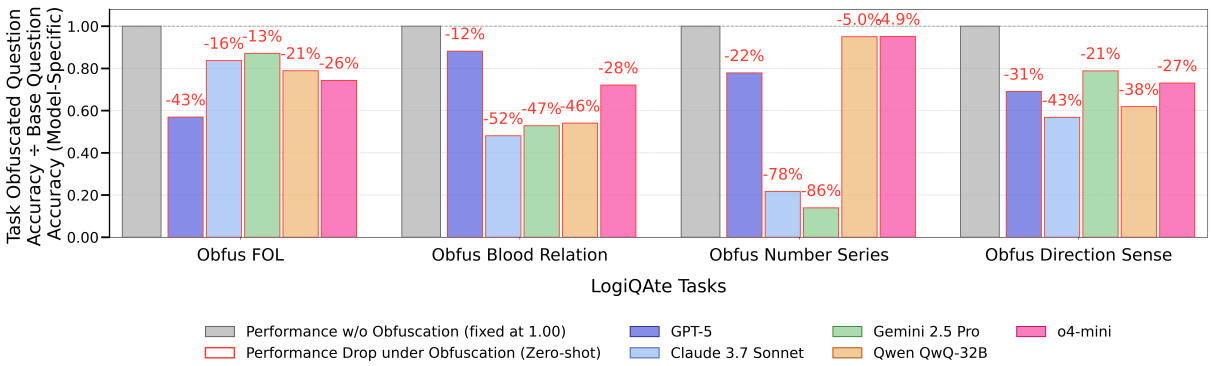

Figure 2: **Logical obfuscation degrades performance across tasks in LogiQAte.** We report relative accuracy for obfuscated questions against base questions (performance fixed at 1.00) across five LLMs under *zero-shot* settings. Across all tasks and settings, we observe consistent degradation, up to -86%, highlighting the brittleness of LLM reasoning under logical obfuscation.

on factual QA or code. To the best of our knowledge, **LogiQAte** is the first benchmark to apply logical obfuscation, via equivalence, preserving rewrites, to assess reasoning robustness across deductive, relational, numerical, directional, and spatial tasks.

3 Proposed Setup

To rigorously evaluate the robustness of LLMs' logical reasoning, we introduce a novel prompting framework called **Logifus**. The central principle of Logifus is the application of controlled, logic-preserving obfuscation. This involves transforming questions and premises to alter their surface-level linguistic or structural representation while keeping the underlying logical structure and, consequently, the correct answer invariant. We define an obfuscation function f as a mapping $f : Q \rightarrow Q'$, where $Q = \{q_1, q_2, \ldots, q_n\}$ denotes a set of questions and Q' is its *logically equivalent* obfuscated counterpart. For any question $q_i \in Q$ and its corresponding *logically equivalent* obfuscated version $q'_i \in Q'$, both share the same correct solution as they are logically equivalent. In other words, if q_i has a valid solution M_i, then q'_i yields the same solution M_i, even though their question's surface forms differ because q'_i is *logically equivalent* to q'_i, compelling models to engage in deeper, more authentic reasoning processes in understanding questions. We report all Logifus algorithms in Appendix A.5.

Leveraging the Logifus framework, we introduce **LogiQAte**, a novel diagnostic benchmark comprising 1,108 questions, designed and instantiated across four complementary reasoning tasks to quantitatively assess a model's sensitivity to logical obfuscations. To guarantee the benchmark's integrity, every base question and its corresponding obfuscated version underwent a stringent validation process involving a two-human-annotator loop. This ensured that the logical equivalence and the original conclusion were preserved, achieving near-perfect agreement across all tasks (Cohen's $\kappa > 0.88$). The task types are visualized in Fig. 3 and described below.

3.1 Obfus FOL

Task Description. First-order logic (FOL) serves as a gold standard for evaluating structured, formal reasoning. Our Obfus First-Order Logic task, comprising 272 questions, applies the Logifus framework to FOL entailment problems to probe a model's ability to handle logically complex rewrites. This is achieved through *FOL Transformation*, where we generate obfuscated variants by rephrasing premises using equivalent logical formulations, such as applying De Morgan's laws or double-negation elimination. For instance, a premise like *"Humans are mammals"* ($\forall x\,(Human(x) \rightarrow Mammal(x))$) is transformed into a convoluted but logically equivalent form like *"It is false that if there exists an entity that is not an animal, then there exists an entity that is a human who is not a mammal"*.

Dataset Construction. The construction of questions for the Obfus FOL task started with entries from the FOLIO (Han et al., 2022) dataset; we first used GPT-4o (OpenAI, 2024) to simplify the natural language premises, creating a clearer, less verbose baseline. To safeguard against potential AI-generated inaccuracies, these simplified premises were cross-verified by two human annotators, achieving a strong inter-annotator agree-

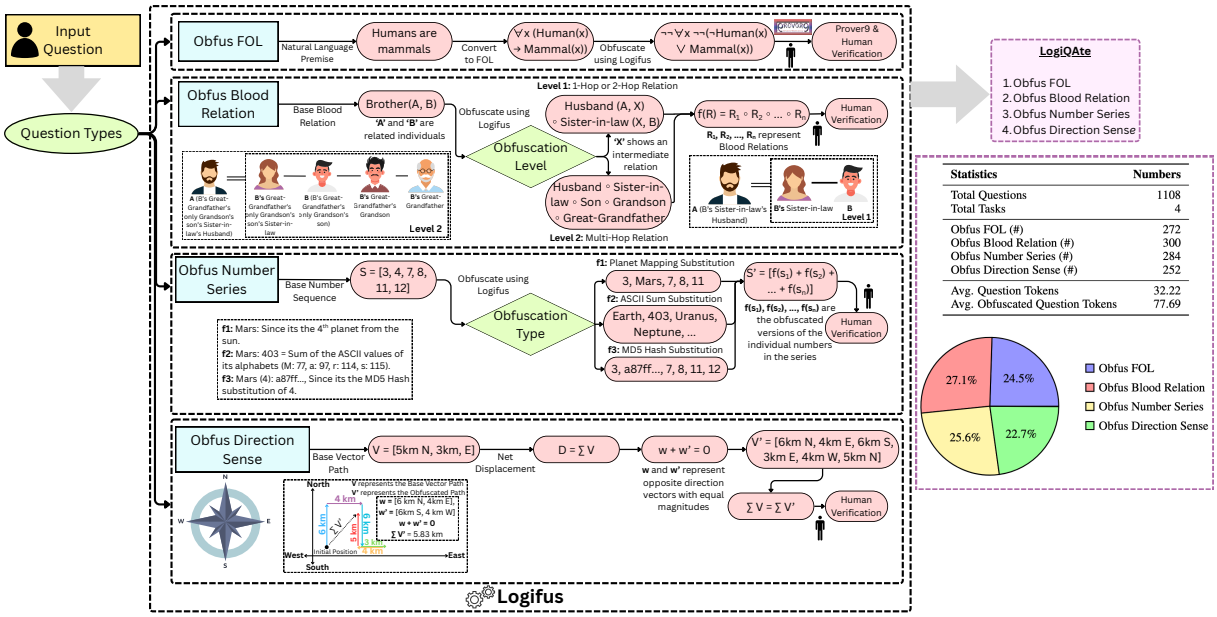

Figure 3: **Overview of LogiQAte and Logifus.** From a user question (left), four task types are generated: Obfus FOL, Obfus Blood Relation, Obfus Number Series, and Obfus Direction Sense. Each task applies a logical equivalence-preserving transformation and undergoes human (along with Prover9 (McCune, 2005–2010) in the case of Obfus FOL) verification. The right panel summarizes the dataset statistics: 1,108 questions across four tasks.

ment (Cohen's $\kappa = 0.95$). For a higher degree of certainty, we employed Prover9 (McCune, 2005–2010), an automated theorem prover, to provide a formal guarantee of logical equivalence (cf. Appendix for procedure). Only after this rigorous baseline validation were the premises obfuscated, and a final manual verification by two annotators confirmed the logical integrity of the obfuscated versions, yielding a high agreement score (Cohen's $\kappa = 0.89$).

FOL Transformation

▶ **Input**: A set of premises $\mathcal{P}$ and a conclusion $\mathcal{C}$ where $\mathcal{P} \models \mathcal{C}$.

▶ **Transformation**: Generate an obfuscated premise set $\mathcal{P}'$ using logical equivalences (e.g., De Morgan's laws) such that $\mathcal{P} \equiv \mathcal{P}'$.

▶ **Output**: An obfuscated problem $(\mathcal{P}', \mathcal{C})$ where the entailment $\mathcal{P}' \models \mathcal{C}$ still holds.

3.2 Obfus Blood Relation

Task Description. Blood-relation puzzles are a classic benchmark for relational reasoning, which involves understanding and reasoning about relationships among entities, people, or objects. Our Obfus Blood Relation task, consisting of 300 questions, evaluates these skills using a technique we term *Relational Obfuscation*. This process involves systematically transforming straightforward kinship relations into indirect, multi-hop descriptions that preserve the underlying family structure, thereby increasing linguistic and cognitive complexity. For example, the direct relation *"A is the brother of B"* is transformed in **Obfuscation Level 1** to a two-step relation like *"A is the sister-in-law's husband of B."* In **Obfuscation Level 2**, the complexity is further increased with a longer reasoning chain, such as *"A is the great-grandfather's only grandson's son's sister-in-law's husband of B."* These levels systematically increase the cognitive load, allowing us to measure a model's breaking point.

Relational Obfuscation

▶ **Input**: A direct kinship statement (e.g., "A is the brother of B").

▶ **Transformation**: Rewrite the statement as an indirect, multi-hop description (e.g., "A is the sister-in-law's husband of B").

▶ **Output**: An obfuscated puzzle with increased reasoning chain length but an unchanged underlying family graph.

Dataset Construction. The construction of questions for the Obfus Blood Relation task, was done by manually curating puzzles from reputable educational platforms such as GeeksforGeeks, IndiaBix and Brilliant. For each puzzle, one key relational premise was transformed using our Level 1 and Level 2 obfuscation strategies. The logical equivalence of every generated obfuscation was manually confirmed by two human annotators, who cross-checked the obfuscation type and logical integrity, achieving a strong inter-annotator

agreement score (Cohen's κ = 90%).

3.3 Obfus Number Series

Task Description. Our task Obfus Number Series, containing 284 questions, evaluates abstract pattern recognition and inductive reasoning, by applying Logifus through a technique, we term as *Pattern-Preserving Obfuscation*, which is defined as disguising the surface representation of numbers without altering the underlying mathematical pattern of the sequence. This type is organized into 3 distinct parts, each derived from the same set of base sequences but transformed using different obfuscation strategies, detailed below:

(i) Type 1: Planet-Name Substitution. We replace each integer in the sequence with a fixed, human-readable token drawn from a small vocabulary of celestial body names (e.g., *Mercury*, *Venus*, *Earth*, *Mars*, ...). The mapping is one-to-one and deterministic: for example, the integer `4` is always rendered as *Mars*. This substitution preserves ordinal relationships but hides the numeric form, forcing the model to treat tokens as abstract symbols that nevertheless encode an underlying numeric mapping. *Example:* the base sequence `[2, 4, 6, 8]` becomes {*Venus*, *Mars*, *Jupiter*, *Saturn*} under a fixed mapping (2→Venus, 4→Mars, 6→Jupiter, 8→Saturn).

(ii) Type 2: ASCII Sum Substitution. After performing the planet-name substitution (Type 1), we optionally convert each planet label to a numeric encoding equal to the sum of its characters' ASCII (or Unicode codepoint) values. For instance, the string "Mars" becomes the integer 403 if the ASCII sum of 'M' + 'a' + 'r' + 's' equals 403. This produces a purely numeric sequence that is semantically detached from the original numbers (the numeric values no longer follow the original progression), but the mapping from original number → planet name → ASCII-sum is deterministic and reversible during dataset construction. This transformation adds a layer of symbol-to-number encoding that is nontrivial for models to invert without recognizing the intermediate symbol mapping. *Example: Mars* → ASCII sum 403, *Venus* → ASCII sum 541, so the visible sequence could read `[541, 403, 857, ...]` even though the latent pattern was an arithmetic progression.

(iii) Type 3: MD5 Hash Substitution. For the strongest surface obfuscation, we replace each original number (or its planet-name token) with its 32-character MD5 hash string (hexadecimal). The MD5 mapping is deterministic but cryptographic in appearance (e.g., the digit '4' might map to `a87ff6...`). Because MD5 outputs look like random character sequences, this substitution makes the visible sequence appear entirely unstructured unless the model (or an auxiliary process) can invert or otherwise reason about the hash mapping. We use MD5 to simulate opaque, high-entropy encodings that a model must either ignore or decode via an external mapping. *Example:* base element `4` → planet token *Mars* → MD5 hash `a87ff6...`; the sequence shown to the model becomes {`d41d8c...`, `a87ff6...`, `0cc175...`, ...}.

Dataset Construction. The Obfus Number Series dataset was built from standard aptitude and reasoning sources (e.g., IndiaBix, Brilliant.org, test-prep materials). A hardcoded python script (a) ingests canonical sequences, (b) applies predefined mappings to generate obfuscated variants, and (c) logs forward and reverse mappings for exact verification and answer reconstruction. Each obfuscated instance was manually reviewed to ensure the original generative rule remained uniquely recoverable under the allowed decoding procedure.

Pattern-Preserving Obfuscation

► **Input:** A numerical sequence $S = \{n_1, n_2, \ldots, n_k\}$ (e.g., an arithmetic or geometric sequence).

► **Transformation:** Apply a deterministic mapping pipeline to each element: (i) map n_i to a planet-name token via a fixed lookup table (Type 1); (ii) optionally convert that token to its ASCII-sum integer (Type 2); (iii) optionally replace the token or ASCII-sum with its MD5 hex string (Type 3). The pipeline may use one, two, or all three steps to produce progressively stronger obfuscation.

► **Output:** An obfuscated sequence $S' = \{f(n_1), f(n_2), \ldots, f(n_k)\}$ that is visibly different from S but preserves a reversible relationship to S for dataset generation and verification.

3.4 Obfus Direction Sense

Task Description. Directional sense puzzles are fundamental for assessing spatial reasoning, such as path tracing, orientation tracking, left–right turn reasoning, and displacement calculation. Our Obfus Direction Sense task, consisting of 252 questions, utilizes a method we call *Spatial Obfusca-*

tion. This technique introduces complexity by embedding irrelevant detours and superfluous conditions within a set of directions, while ensuring the final destination remains identical to the original problem. For instance, a simple path such as *"A person walks 5 km north and then 3 km east"* can be transformed into a longer, more complex version: *"A person walks 6 km north, then 4 km east, 6 km south, 3 km east, 4 km west, and finally 5 km north"*. The additional movements, such as walking 6 km north and later 6 km south, are self-canceling detours that make the route appear more intricate while keeping the final position unchanged.

Spatial Obfuscation

▶ **Input**: A sequence of movement vectors $V = \{v_1, v_2, \ldots, v_n\}$.

▶ **Transformation**: Insert self-canceling vector pairs (w, w^{-1}) into the sequence, where $w + w^{-1} = 0$.

▶ **Output**: An obfuscated sequence $V' = \{v_1, w, \ldots, w^{-1}, v_n\}$ where the net displacement is unchanged: $\sum V = \sum V'$.

Dataset Construction. For Obfus Direction Sense, we collected base directional puzzles from widely used logic and reasoning sources (e.g., IndiaBix, Brilliant, and general aptitude sites). For each base puzzle, a simple Python program generated obfuscated variants by randomly inserting paired detours (e.g., "go X east, then X west," or "go Y north, then Y south") at selected positions, directions, or magnitudes in the walking sequence. These pairs cancel out, preserving the final position and orientation; magnitudes and insertion points were varied to create multiple obfuscations per base. Each base-obfuscated pair was manually reviewed by two human annotators to verify unchanged displacement and absence of ambiguity, yielding strong inter-annotator agreement (Cohen's $\kappa = 0.88$).

4 Evaluation

4.1 Setup

We evaluate nine state-of-the-art models (both open- and closed-source) on LogiQAte, grouped as follows: general-purpose LLMs: GPT-4o (OpenAI, 2024) and Claude 3.7 Sonnet (Anthropic, 2025); reasoning-focused LLMs: GPT-5 (OpenAI, 2025a), o4-mini (OpenAI, 2025b), Gemini 2.5 Pro (Gemini Team, 2025), and Qwen QwQ-32B (Team, 2025). We also evaluate lower-parameter models: GPT-4o-mini (OpenAI, 2025b), Gemma-3-27B-IT (Google DeepMind, 2025), and Llama-4-Maverick-17B (Touvron et al., 2025) on our dataset (cf. Appendix A.2). All models are evaluated across zero-shot, few-shot, and chain-of-thought (CoT) settings with a temperature of 0.0.

Evaluation Metric. Model performance is measured using Exact Match (EM) Accuracy, which computes the percentage of samples where the model's normalized prediction, $\hat{y}_i$, exactly matches the normalized ground truth, y_i: for a task of N evaluation samples, EM Accuracy = $\frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\{\text{norm}(\hat{y}_i) = \text{norm}(y_i)\}$. Normalization involves standardizing case, removing punctuation, and ensuring whitespace uniformity to make comparisons resilient to minor textual variations (e.g., "Sister-in-law" ≡ "sister in law").

5 Results and Discussion

Table 1 and Table 2 (cf. Appendix A.1) summarize the performance of large reasoning and general-purpose models, while Table 3 (cf. Appendix A.2) reports results for lower-parameter models, across all LogiQAte tasks under different prompting settings, and Appendix A.3 reports a qualitative analysis for each LogiQAte task. We provide a breakdown of zero-shot performance on the Obfus Number Series by obfuscation type in Fig. 4. We set the QA level on par with undergraduate (UG) standards, validated by UG students who achieved near-perfect performance, up to 99% accuracy on average across two human evaluators, on both base and obfuscated questions, while even highly capable reasoning models like GPT-5 struggled. This stark contrast casts a shadow on the reasoning ability of current LLMs.

Logical Obfuscation Reduces Performance Across Tasks, Prompt Settings, and Models. In Table 1, average accuracy drops range from 18.70% to 48.70%, with Table 2 showing a similar decline of 40.54%-49.37%. Strikingly, on Obfus FOL, the reasoning-focused GPT-5 falls from 98% to 56% (zero-shot); on Obfus Number Series, GPT-4o drops by 87.3%, Claude 3.7 Sonnet by 78.8%, and Gemini 2.5 Pro by 87.2%. Aggregating across all tasks and prompt settings, reasoning-focused models degrade by roughly 20-30%, while general-purpose models drop by about 45-50% on average.

Task & Question Variant	GPT-5			o4-mini			Qwen QwQ-32B			Claude 3.7 Sonnet		
	Zero-shot	Few-shot	CoT	Zero-shot	Few-shot	CoT	Zero-shot	Few-shot	CoT	Zero-shot	Few-shot	CoT
				(Reasoning Models)						*(No Reasoning)*		
Obfus FOL												
Base	0.98	0.87	0.82	0.92	0.92	0.92	0.77	0.76	0.84	0.82	0.85	0.82
Obfuscation	0.56	0.59	0.61	0.68	0.63	0.67	0.61	0.63	0.66	0.69	0.72	0.68
Obfus Blood Relation												
Base	0.52	0.73	0.72	0.69	0.68	0.74	0.42	0.43	0.48	0.49	0.45	0.59
Obfuscation L1	0.46	0.68	0.67	0.59	0.61	0.60	0.26	0.30	0.29	0.26	0.36	0.29
Obfuscation L2	0.45	0.59	0.55	0.41	0.44	0.45	0.19	0.25	0.24	0.21	0.26	0.24
Obfus Number Series												
Base	0.77	0.64	0.65	0.60	0.64	0.64	0.55	0.59	0.59	0.28	0.35	0.36
Obfuscation	0.60	0.57	0.57	0.57	0.57	0.56	0.53	0.53	0.51	0.06	0.08	0.07
Obfus Direction Sense												
Base	0.64	0.72	0.63	0.62	0.69	0.60	0.59	0.65	0.56	0.29	0.40	0.36
Obfuscation	0.44	0.55	0.50	0.45	0.51	0.48	0.37	0.46	0.36	0.17	0.22	0.21
Average Performance Degradation (%)	-27.00%	-20.20%	-18.70%	-21.40%	-22.70%	-22.00%	-27.50%	-23.30%	-26.10%	-47.50%	-41.60%	-48.70%

Table 1: Performance degradation is observed across four logical obfuscation tasks in LogiQAte, with average drops ranging from 18.70% to 48.70% compared to base question performance. GPT-5, o4-mini, and Qwen QwQ-32B operate under reasoning configuration, while Claude 3.7 Sonnet operates with no reasoning. Each model is evaluated under Zero-shot, Few-shot, and Chain-of-Thought (CoT) prompt settings. Red cells indicate the lowest score for each prompting strategy within each task row.

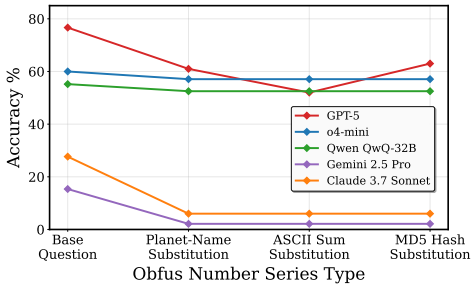

Figure 4: **Obfuscation reduces performance across types in Obfus Number Series.** Accuracy drops range from 20-45% for reasoning-focused models and up to 80-90% for general-purpose models such as Claude 3.7 Sonnet and Gemini 2.5 Pro, which already struggle on base questions. GPT-5 maintains higher accuracy across all obfuscation and base variants.

Task-Level Vulnerabilities Under Logical Obfuscation. Obfus Blood Relation shows a steady decline from base to Obfuscation L1 and L2: Claude 3.7 (CoT) drops from 59% (base) to 24% (L2), while GPT-5 (few-shot) declines from 73% to 59% (–19%). Obfus Number Series is especially punishing for general-purpose models, GPT-4o falls by 87%, Claude 3.7 by 79%, and Gemini 2.5 Pro by 87%, whereas reasoning-focused models sustain comparatively higher accuracy, even on base questions. Obfus Direction Sense stresses state tracking: GPT-5 (few-shot) drops by 24%, Qwen QwQ-32B (few-shot) by 29%, and Claude 3.7 (zero-shot) by 41%. These results highlight how logical obfuscation amplifies the cognitive load across tasks, exposing weaknesses in reasoning consistency.

Prompting Strategies Provide Partial Relief, Not Robustness. Advanced prompting methods like CoT reduce GPT-5's average degradation from 27.00% (zero-shot) to 18.70%, but the effect remains inconsistent; Gemini 2.5 Pro still suffers drops exceeding 40% across all prompt settings (cf. Table 1). Few-shot gains are similarly unstable, indicating models' sensitivity to the surface form of examples. While scale and prompting improve base-question accuracy, they fail to ensure invariance to logically equivalent reformulations.

Lower-Parameter Models Exhibit Greater Brittleness. As shown in Table 3, smaller models incur sharply higher performance losses. GPT-4o-mini's accuracy drops by 48.90-51.30% across prompts, while Gemma-3-27B-IT declines by 39.28-44.86%. Llama-4-Maverick-17B shows notable prompt sensitivity, its 24.75% zero-shot drop worsens to over 42% with few-shot guidance. This fragility is evident in tasks like Obfus Direction Sense, where GPT-4o-mini's accuracy collapses from 21% to 5%. Overall, smaller models remain especially brittle to the state-tracking and symbol-masking challenges introduced by Logifus, even with explicit prompt guidance.

5.1 Mechanistic Analysis of LLMs

To better understand *why* these LLMs fail under logical obfuscation on LogiQAte tasks, we conduct an intrinsic, model-level analysis focusing on two aspects: first, we perform a *memorization sensitivity analysis* to test whether obfuscated inputs trigger higher dependence on memorized patterns from training data, thereby reducing genuine reasoning, and second, we compute *layer-wise next-token log probabilities* to trace how confidence evolves as the model processes a logically equivalent question. This helps identify where within the network reasoning begins to degrade. All experiments are conducted on LLaMA 3.1 8B (Touvron

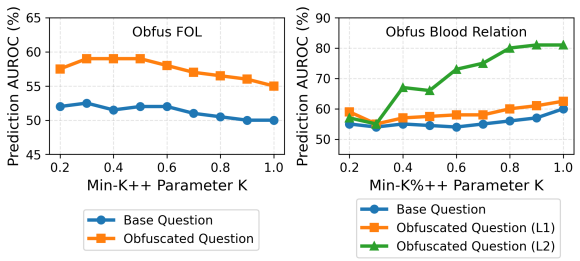

Figure 5: Obfuscation increases prediction AUROC (indicating greater memorization) by 12-15% in Obfus FOL and 20-25% in Obfus Blood Relation across both L1 and L2 levels (L2 being higher). The y-axis shows Prediction AUROC (%), and the x-axis denotes K, the fraction of least-likely tokens used in computing membership scores. Plotting Prediction AUROC against K illustrates how obfuscation amplifies memorization cues, making models rely more on training data patterns than genuine reasoning.

et al., 2023) (due to budget constraints) using a single NVIDIA A100 GPU.

Memorization Sensitivity Analysis. As shown in Fig. 5.1, we apply the Min-K%++ (Zhang et al., 2025) membership inference technique, which detects memorization by checking whether a model assigns unusually high confidence to tokens seen during training. The parameter K controls the fraction of least-likely tokens (by log probability) used when estimating a question's membership score, lower K focuses on the model's most uncertain regions, while higher K aggregates over more of the sequence. By plotting prediction AUROC across varying K, we visualize how confidence patterns differ between base and obfuscated questions. A higher AUROC indicates stronger memorization and greater reliance on training data rather than genuine reasoning.

In Obfus FOL, obfuscated questions consistently yield higher AUROC scores (56-59%) than their base counterparts (49-53%) across K, showing increased reliance on memorized logical patterns under obfuscation. In Obfus Blood Relation, the effect is even stronger: Level 1 obfuscation reaches about 62% AUROC, while Level 2 rises to 82% (both peaking at K=1.0). Base questions remain moderate (55-60%), indicating that relational reasoning tasks involving entity-based relationships make the model more prone to memorization.

Layer-wise Next Token Confidence Analysis. Mid-to-late transformer layers have been shown to capture semantic and relational understanding C (Tenney et al., 2019; Wei et al., 2022a). In Fig. 6, for each transformer layer l and target token x_t, we report the layer-wise next-token log

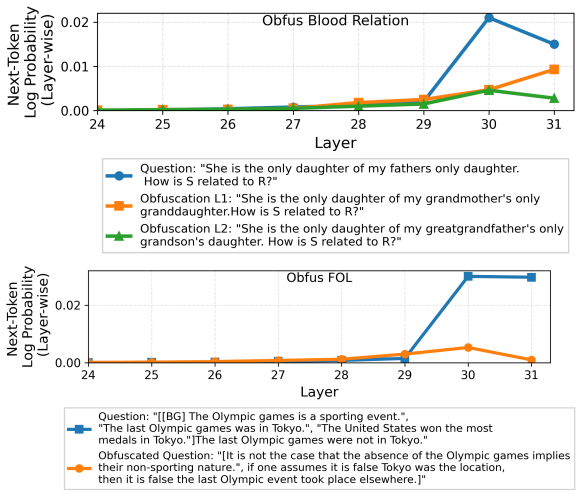

Figure 6: Logical obfuscation reduces late-layer confidence (layers 28-31) across both tasks, showing 50-80% suppression in next-token log probability compared to base questions. This indicates that obfuscation disrupts the model's deep-layer reasoning and semantic consolidation.

probability $\ell_t^{(l)} = \log p_\theta(x_t \mid x_{<t}, h^{(l)})$ and the sequence-average $\bar{\ell}^{(l)} = \frac{1}{T}\sum_{t=1}^{T} \ell_t^{(l)}$ for both base and obfuscated questions to observe how confidence evolves across layers and whether obfuscation affects deep-layer consolidation. In both of the tasks, the late-layer region (layers 28-31) represents where the model typically consolidates semantic and relational reasoning signals before final prediction. In Obfus Blood Relation, we observe that obfuscated variants (L1 and L2) exhibit a reduction of nearly 50-75% in overall confidence magnitude than base questions. In Obfus FOL, the obfuscated questions show over 80% suppression of the confidence rise observed in the base question. These results show that logical obfuscation adversely affects the model's reasoning ability, as it disrupts the most important deep layers that are primarily responsible for integrating semantic cues and producing the correct answer.

6 Conclusion

In this work, we introduce Logifus, a novel structure- and equivalence-preserving logical obfuscation algorithm, and leveraging it, present LogiQAte, the first benchmark designed to rigorously evaluate the reasoning capabilities of LLMs under logical obfuscation across four complementary tasks. Our findings reveal that logical obfuscation consistently weakens model reasoning, indicating that current LLMs rely heavily on surface-level patterns rather than genuine understanding.

7 Limitations and Future Work

While LogiQAte provides a strong foundation for evaluating logical reasoning in large language models, it is currently limited to tasks in English. In future work, we plan to expand the dataset to include multilingual data, with a particular emphasis on low-resource languages. To improve generalizability and diversity, we also aim to extend LogiQAte to additional reasoning categories such as seating arrangements and commonsense inference. Furthermore, developing adaptive logical obfuscation techniques, could strengthen the diagnostic depth and robustness of the framework.

Ethics Statement

All paid models used in this study were accessed through valid subscriptions and in full compliance with the terms of service of their respective providers. No attempts were made to bypass usage restrictions or violate the ethical guidelines established by the model developers.

References

Anthropic. 2025. Claude 4: Safe and robust reasoning in large-scale conversational ai. *arXiv preprint arXiv:2503.23456*.

Anthropic. 2025. Introducing Claude 3.7 Sonnet. https://www.anthropic.com/news/claude-3-7-sonnet.

Andrew M. Bean, Simi Hellsten, Harry Mayne, Jabez Magomere, Ethan A. Chi, Ryan Chi, Scott A. Hale, and Hannah Rose Kirk. 2024. Lingoly: A benchmark of olympiad-level linguistic reasoning puzzles in low-resource and extinct languages. In *Advances in Neural Information Processing Systems*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Alain Colmerauer and Philippe Roussel. 1973. The birth of prolog. *ACM SIGPLAN Notices*.

Google DeepMind Gemini Team. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Shubhra Ghosh, Abhilekh Borah, Aditya Kumar Guru, and Kripabandhu Ghosh. 2025. Obfusqate: A proposed framework to evaluate llm robustness on obfuscated factual question answering. *arXiv preprint arXiv:2508.07321*.

Google DeepMind. 2025. Gemma 3: Next-generation multimodal ai model. https://deepmind.google/models/gemma/gemma-3/. Accessed October 2025.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, et al. 2022. Folio: Natural language reasoning with first-order logic. *arXiv preprint arXiv:2209.00840*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

IBM Think. 2024. How smart is machine intelligence? ai aces games but fails basic reasoning. Interview with Brent Smolinski, IBM VP and Senior Partner.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 3622–3628.

Aman Madaan and Amir Yazdanbakhsh. 2022. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *Advances in Neural Information Processing Systems*.

John McCarthy. 1959. Programs with common sense. *Proceedings of the Symposium on Mechanisation of Thought Processes*.

W. McCune. 2005–2010. Prover9 and mace4. http://www.cs.unm.edu/~mccune/prover9/.

Seyedreza Mohseni, Seyedali Mohammadi, Deepa Tilwani, Yash Saxena, Gerald Ketu Ndawula, Sriram Vema, Edward Raff, and Manas Gaur. 2025. Can llms obfuscate code? a systematic analysis of large language models into assembly code obfuscation. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

OpenAI. 2024. Gpt-4o system card. https://arxiv.org/abs/2410.21276. Accessed: 2025-05-22.

OpenAI. 2025a. Gpt-5: Scaling laws and emergent reasoning abilities. *arXiv preprint arXiv:2505.12345*.

OpenAI. 2025b. o4: An efficient reasoning model. *Technical Report*.

OpenAI. 2025c. o4: Miniaturized reasoning-optimized large language model. *arXiv preprint arXiv:2502.34567*.

Mihir Parmar, Nisarg Patel, Neeraj Varshney, Mutsumi Nakamura, Man Luo, Santosh Mashetty, Arindam Mitra, and Chitta Baral. 2024. Logicbench: Towards systematic evaluation of logical reasoning ability of large language models. *arXiv preprint arXiv:2404.15522*.

Nisarg Patel, Mohith Kulkarni, Neeraj Varshney, Mihir Parmar, Arindam Mitra, and Chitta Baral. 2024. Multi-logieval: Towards evaluating multi-step logical reasoning ability of large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 20856–20879.

Edward H. Shortliffe. 1976. *Computer-Based Medical Consultations: MYCIN*. Elsevier.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Oyvind Tafjord, Bhavana Dalvi Mishra, and Peter Clark. 2021. Proofwriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634.

Qwen Team. 2025. Qwq-32b model card. https://huggingface.co/Qwen/QwQ-32B. Accessed: 2025-05-22.

Ian Tenney, Patrick Xia, Berlin Chen, Alexis Ross, Dahlia Malkin, Slav Petrov, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized embeddings. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Jidong Tian, Yitian Li, Wenqing Chen, Liqiang Xiao, Hao He, and Yaohui Jin. 2021. Diagnosing the first-order logical reasoning ability through logicnli. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3738–3747.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Pedro M. Ferreira, Jean Peyrot, Baptiste W. T. House, Hugo Le Jeune, Etienne Kudelka, Thibault Lacroix, Thomas Lavril, Thibault Laclaur, Mélanie Dessors, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Gabriel Schulz, Colin Raffel, Abhishek Chaudhary, Alexander Bradford, Aravind Neelakantan, Diogo Almeida, Pedro Tabacof, Pierre Mazare, Li Jing, et al. 2025. Llama 4 maverick: Advancements in efficient large language models. https://ai.meta.com/publications/llama-4-maverick. Accessed October 2025.

Keyon Vafa, Justin Y. Chen, Ashesh Rambachan, Jon Kleinberg, and Sendhil Mullainathan. 2024. Evaluating the world model implicit in a generative model. *arXiv preprint arXiv:2406.03689*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, ..., and Qiming Le. 2022a. Emergent abilities of large language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3115–3127.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022b. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Joseph Weizenbaum. 1966. Eliza–a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45.

Terry Winograd. 1971. Procedures as a representation for data in a computer program for understanding natural language. In *MIT Project MAC Technical Report*.

Chulin Xie, Yangsibo Huang, Chiyuan Zhang, Da Yu, Xinyun Chen, Bill Yuchen Lin, Bo Li, Badih Ghazi, and Ravi Kumar. 2025. On memorization of large language models in logical reasoning. In *ICLR 2025*.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. 2020. Reclor: A reading comprehension dataset requiring logical reasoning. In *International Conference on Learning Representations*.

Jingyang Zhang, Jingwei Sun, Eric Yeats, Yang Ouyang, Martin Kuo, Jianyi Zhang, Hao Frank Yang, and Hai Li. 2025. Min-k%++: Improved baseline for pre-training data detection from large language models. In *The Thirteenth International Conference on Learning Representations*.

A Appendix

A.1 Performance on More Models

Table 2 presents a comprehensive performance comparison between GPT-4o and Gemini 2.5 Pro across all four LogiQAte tasks under zero-shot, few-shot, and CoT prompt setting. Both models demonstrate an average performance drop of 45.66% across all prompting configurations under logical obsfuscation. GPT-4o shows particularly severe degradation in Obfus Direction Sense tasks (76% average drop), while Gemini 2.5 Pro, despite being a reasoning model, achieves near-zero accuracy on Obfus Number Series tasks (87% drop). Gemini 2.5's technical report acknowledges that current reasoning architectures still struggle when benchmark patterns are disrupted, exposing persistent model blind spots even in basic logical tasks (Gemini Team, 2025), explaining why even these reasoning-enhanced models experience substantial accuracy collapse.

Task & Question Variant	GPT-4o			Gemini 2.5 Pro		
	Zero-shot	Few-shot	CoT	Zero-shot	Few-shot	CoT
	(No Reasoning)			*(Reasoning Model)*		
Obfus FOL						
Base	0.86	0.90	0.91	0.78	0.86	0.80
Obfuscation	0.74	0.75	0.76	0.68	0.66	0.76
Obfus Blood Relation						
Base	0.57	0.57	0.59	0.51	0.51	0.53
Obfuscation L1	0.37	0.40	0.37	0.35	0.34	0.36
Obfuscation L2	0.27	0.26	0.26	0.19	0.19	0.25
Obfus Number Series						
Base	0.24	0.22	0.33	0.15	0.18	0.21
Obfuscation	0.04	0.03	0.03	0.02	0.02	0.03
Obfus Direction Sense						
Base	0.29	0.29	0.27	0.52	0.69	0.52
Obfuscation	0.14	0.14	0.14	0.41	0.52	0.38
Average Performance Degradation (%)	-47.60%	-47.22%	-49.37%	-42.91%	-46.33%	-40.54%

Table 2: Performance degradation is observed across four logical obfuscation tasks in LogiQAte, with average drops ranging from 40.54% to 49.37% compared to base question performance. GPT-4o operates without reasoning configuration, while Gemini 2.5 Pro is a reasoning model. Each model is evaluated under Zero-shot, Few-shot, and Chain-of-Thought (CoT) prompt settings. Red cells indicate the lowest score for each prompting strategy within each task row.

A.2 Performance on Smaller Models

Table 3 extends the evaluation to three additional lower-parameter non-reasoning models: GPT-4o-mini, Llama-4-Maverick-17B, and Gemma-3-27B-IT. Performance degradation averages 42.77% across all prompt settings, with GPT-4o-mini experiencing the most severe drops, 50.27% on an average. All three of these models collapse nearly completely on Obfus Number Series under logical obfuscation, with accuracy falling to near-zero levels representing drops exceeding 78% from already-weak baselines.

A.3 Qualitative Analysis

To understand the failure modes of LLMs on obfuscated questions, we performed a qualitative analysis of GPT-5's reasoning trace as it justifies its answer. For each LogiQAte task, we selected two representative examples: one where the model produced the correct answer and one where it failed. We prompted the model in the zero-shot setting to predict its answer along with a brief justification of *why*. Table 4 and Table 5 and presents these examples with the model's justifications for each obfuscated question. We manually analyzed each reasoning justification and applied a two-color highlighting scheme, green and red, to identify correct versus incorrect reasoning tokens or traces. In one of the examples in Obfus FOL, the model correctly identifies that *"Legend of Zelda is created by such an entity"* and that Zelda sold over one million copies, but erroneously reverses the direction of implication, treating selling over one million copies as merely necessary for Top 10 inclusion rather than sufficient, incorrectly concluding that *"exceeding one million sales does not suffice to be in the Top 10"*. This represents a failure in parsing complex logical negations and understanding implication directionality. In Obfus Blood Relation, GPT-5 correctly traces *"my great-grandfather's only grandson"* to V's father and simplifies to the *"only son of my father"*, but then makes an unjustified assumption that V is the only son, concluding that the boy being pointed at is the *"son of V"*, an error arising from over-interpreting *"only son"* at the wrong generation level. The Obfus Number Series incorrect example reveals that GPT-5 correctly decodes the MD5 hashes, identifies the pattern (sum of consecutive squares $1^2, 2^2, 3^2, \ldots$), and determines *"the hidden term between 15 and 56 is 31"*, yet outputs 141 by continuing the sequence beyond the given terms ($92 + 7^2 = 141$), answering incorrectly. In Obfus Direction Sense, the error stems from assuming *"they start facing North"* and *"both walk 1 km North together"* despite the problem stating

Task & Question Variant	GPT-4o-mini			Llama-4-Maverick-17B			Gemma-3-27B-IT		
	Zero-shot	Few-shot	CoT	Zero-shot	Few-shot	CoT	Zero-shot	Few-shot	CoT
				(No Reasoning)					
Obfus FOL									
Base	0.73	0.57	0.56	0.65	0.65	0.45	0.82	0.80	0.78
Obfuscation	0.73	0.57	0.56	0.65	0.65	0.45	0.71	0.73	0.62
Obfus Blood Relation									
Base	0.50	0.46	0.48	0.25	0.41	0.21	0.11	0.49	0.46
Obfuscation L1	0.29	0.28	0.28	0.22	0.31	0.17	0.06	0.30	0.27
Obfuscation L2	0.23	0.24	0.20	0.13	0.18	0.08	0.10	0.21	0.25
Obfus Number Series									
Base	0.24	0.22	0.33	0.01	0.02	0.05	0.14	0.16	0.13
Obfuscation	0.04	0.03	0.03	0.01	0.00	0.00	0.03	0.03	0.03
Obfus Direction Sense									
Base	0.21	0.24	0.21	0.14	0.35	0.29	0.31	0.31	0.28
Obfuscation	0.05	0.07	0.08	0.13	0.23	0.22	0.17	0.18	0.18
Average Performance Degradation (%)	-51.30%	-48.90%	-50.62%	-24.75%	-42.64%	-39.11%	-39.28%	-44.86%	-43.45%

Table 3: Performance degradation is observed across four logical obfuscation tasks in LogiQAte, with average drops ranging from 24.75% to 51.30% compared to base question performance. GPT-4o-mini, Llama-4-Maverick-17B, and Gemma-3-27B-IT operate with no reasoning configuration. Each model is evaluated under Zero-shot, Few-shot, and Chain-of-Thought (CoT) prompt settings. Red cells indicate the lowest score for each prompting strategy within each task row.

"in opposite directions". While subsequent turning patterns are correctly identified, this faulty initial condition propagates through all position calculations, yielding 4 km instead of 3 km.

A.4 Human Annotation Procedure and Instructions

To ensure the reliability and validity of *LogiQAte*, all base and obfuscated questions underwent rigorous human verification. Two independent human annotators, both third-year undergraduate Computer Science students with strong backgrounds in logic, discrete mathematics, and reasoning, were recruited for the validation process. Their familiarity with formal logic systems and demonstrated proficiency in analytical reasoning made them well-suited for this task.

Annotation Protocol. The two annotators were provided with detailed task-specific guidelines for each reasoning task:

- **Obfus FOL:** Verify that obfuscated premises remain logically equivalent to base premises and preserve the conclusion's truth value. For example, consider the base premise: *"Humans are mammals"* ($\forall x\ (Human(x) \rightarrow Mammal(x))$). The obfuscated version reads: *"It is false that if there exists an entity that is not an animal, then there exists an entity that is a human who is not a mammal"*. Annotators verified that both versions express the same logical relationship and that any conclusion derived from the base premises holds for the obfuscated premises. They flagged instances where logical transformations altered the truth value of the conclusion.
- **Obfus Blood Relation:** Confirm that multi-hop relational chains preserve the final relationship. For instance, the base statement *"D is the wife of C, C is the father of F"* establishes that D is F's mother. The obfuscated version states: *"D is the wife of C, C is the grandfather's only son of F"*. Annotators traced the obfuscated chain step-by-step: if C is the grandfather's only son of F, then C must be F's father, thus D remains F's mother. They verified that despite the indirect phrasing, the underlying family graph and final answer remain unchanged.
- **Obfus Number Series:** Validate that symbolic substitutions preserve the underlying mathematical pattern. For example, the base sequence $[2, 4, 6, 8]$ follows an arithmetic progression (+2). Under Type 1 obfuscation, this becomes $\{$Venus, Mars, Jupiter, Saturn$\}$ using the planet mapping (Venus=2, Mars=4, Jupiter=6, Saturn=8). Annotators confirmed that (1) the mapping is applied correctly, (2) the arithmetic pattern remains recoverable, and (3) the next term can be uniquely determined by decoding the planet names back to

numbers.

- **Obfus Direction Sense:** Verify that added detours are self-canceling and preserve net displacement. Consider the base path: *"A person walks 5 km north and then 3 km east"* (final position: 5.83 km North-East). The obfuscated version states: *"A person walks 6 km north, then 4 km east, 6 km south, 3 km east, 4 km west, and finally 5 km north"*. Annotators computed each segment: the 6 km north and 6 km south cancel out, as do the 4 km east and 4 km west, leaving the net movement of 5 km north and 3 km east, identical to the base path. They verified that all inserted movements form canceling pairs and the final displacement remains 5.83 km North-East.

Both annotators worked independently, and disagreements were resolved through discussion until consensus was reached. For Obfus FOL, the automated theorem prover, Prover9 (McCune, 2005–2010) was used as an additional verification step prior to human review.

Inter-Annotator Agreement. We measured inter-annotator reliability using Cohen's kappa ($\kappa = \frac{p_o - p_e}{1 - p_e}$), which accounts for chance agreement between the two human annotators. The results demonstrated strong consistency across all tasks: Obfus FOL achieved $\kappa = 0.95$ for base question validation and $\kappa = 0.89$ for obfuscated questions; Obfus Blood Relation obtained $\kappa = 0.90$; and Obfus Direction Sense reached $\kappa = 0.88$. Overall, the benchmark achieved near-perfect agreement across all tasks ($\kappa > 0.88$), confirming that logical equivalence and answer preservation were reliably maintained throughout the obfuscation process. This high agreement underscores both the clarity of our annotation guidelines and the logical soundness of the Logifus framework.

Table 4: Qualitative Analysis of GPT-5 on LogiQAte: Reasoning traces showing correct answers and reasoning steps in green and incorrect ones (answers and reasoning steps) in red across four LogiQAte tasks. We observe that, even when GPT-5 arrives at an incorrect answer, portions of its reasoning are often correct; the model typically makes localized errors at specific inference steps rather than failing completely.

Task	Base Question	Obfus Question	Obfus Answer	Reasoning
Obfus FOL: Given premises and conclusion, determine if conclusion logically follows it. Output TRUE or FALSE with brief justification.	If people perform in school talent shows often, then they attend and are very engaged with school events. People either perform in school talent shows often or are inactive and disinterested members of their community. If people chaperone high school dances, then they are not students who attend the school. All people who are inactive and disinterested members of their community chaperone high school dances. All young children and teenagers who wish to further their academic careers and educational opportunities are students who attend the school. Bonnie either both attends and is very engaged with school events and is a student who attends the school, or she neither attends and is very engaged with school events nor is a student who attends the school. **Ground Truth:** TRUE	It is not the case that not all people when they are not engaged with school events do not perform in school talent shows often. If it is not the case that all people are engaged with school events, then it must be that not everyone is an inactive and disinterested member of their community. Only if people are not students who attend the school will they chaperone high school dances, according to not all other cases. When individuals do not chaperone high school dances, it necessarily follows that they are not inactive and disinterested members of their community. There is no one for whom it is not the case that, unless they wish to further their academic careers and educational opportunities, they are not students who attend the school. Bonnie is either both attending and engaged with school events and attends the school, or it is not the case that she does either.	**FALSE**	The conclusion does not follow. The antecedent is equivalent to Bonnie chaperones or performs often. If she performs, then by 'It is not the case that not all people when they are not engaged ... do not perform' we get she is engaged (so P→E). By 'Bonnie is either both ... engaged ... and attends the school, or ... neither' we have E↔S, hence S. From 'Only if people are not students ... will they chaperone' we infer S→ ¬C, and from 'When individuals do not chaperone ... they are not inactive' we get ¬I. Thus performance implies ¬I(B), contradicting the conclusion's requirement that she be inactive. In the chaperone case, we still lack any premise making her inactive or a young child/teen; and S→W is the only link to wishing. Therefore the stated conditional is not entailed.
	A Japanese game company created the game the Legend of Zelda. All games in the Top 10 list are made by Japanese game companies. If a game sells more than one million copies, then it will be selected into the Top 10 list. The Legend of Zelda sold more than one million copies. **Ground Truth:** TRUE	It is not the case that the absence of a Japanese entity responsible for video games results in the Legend of Zelda being unlinked with creation by such an entity. The state where a Japanese creator is absent implies that it's not possible for a game created and present in the Top 10 list to exist. For every game, the condition of surpassing sales of a million copies is a necessity for it not to be excluded from the Top 10 list. It is not the case that the Legend of Zelda didn't exceed selling more than one million copies.	**FALSE**	From the denial of 'if no Japanese entity exists then the Legend of Zelda is not created by one,' it follows that the absence of a Japanese creator holds and, nonetheless, the Legend of Zelda is created by such an entity. Given that absence, the second premise says that no game that is created and in the Top 10 can exist; in particular, the Legend of Zelda cannot be in the Top 10. The sales premises add only a necessary condition: any Top 10 game must have sold over one million copies, and indeed Zelda did, but exceeding one million sales does not suffice to be in the Top 10. Hence the conclusion that the Legend of Zelda is in the Top 10 does not follow and is in fact contradicted.
Obfus Blood Relation: Given a relationship description, determine the final relationship with brief justification.	Pointing towards a boy V said He is the son of only son of my grandfather, How is that boy related to V? **Ground Truth:** BROTHER	Pointing towards a boy V said He is the son of only son of my father's father, How is that boy related to V?	**BROTHER**	My father's father" is V's grandfather. The only son" of V's grandfather must be V's father (since the grandfather has exactly one son). So the phrase becomes: He is the son of my father." A male child of V's father is V's brother. It is not V himself because V is pointing to another boy as he," and standard reasoning assumes the speaker is not indicating themselves. Therefore, the boy pointed out is V's father's son other than V, making him V's brother.
	Pointing towards a boy V said He is the son of only son of my grandfather, How is that boy related to V? **Ground Truth:** BROTHER	Pointing towards a boy V said He is the son of only son of my greatgrandfather's only grandson, How is that boy related to V?	**Son**	My great-grandfather's only grandson" is two generations below and thus in the parent's generation; with only one male grandchild, that person is V's father. The phrase becomes: the only son of my father," which identifies the single male child of V's father. Since there is exactly one such son, that son is V himself (V is the only son of his father). Therefore, the boy pointed at is "the son of [V]," i.e., V's son. So, the boy is V's son.

Table 5: (Continued) Qualitative Analysis of GPT-5 on LogiQAte: Reasoning traces showing correct answers and reasoning steps in green and incorrect ones (answers and reasoning steps) in red across four LogiQAte tasks. We observe that, even when GPT-5 arrives at an incorrect answer, portions of its reasoning are often correct; the model typically makes localized errors at specific inference steps rather than failing completely.

Task	Base Question	Obfus Question	Obfus Answer	Reasoning
Obfus Number Series: Given a number series with planets representing digits or hashes representing single digits, find the missing term. Planet Mapping: Sun:0, Mercury:1, Venus:2, ..., Neptune:8 Hash Mapping: cfcd... :0, c4ca... :1, c81e... :2, ..., 45c4... :9	2048, 516, 1032, 262, ?, 135, 270 **Ground Truth:** 524	2048, Jupiter Mercury Saturn, Mercury Sun Earth Venus, Venus Saturn Venus, ?, Mercury Earth Jupiter, Venus Uranus Sun	**524**	Using the code Sun=0, Mercury=1, Venus=2, Earth=3, Mars=4, Jupiter=5, Saturn=6, Uranus=7, Neptune=8, Pluto=9, the planets translate to digits: Jupiter Mercury Saturn → 516, Mercury Sun Earth Venus → 1032, Venus Saturn Venus → 262, Mercury Earth Jupiter → 135, Venus Uranus Sun → 270. So the numeric sequence is 2048, 516, 1032, 262, ?, 135, 270. The terms follow an alternating rule: (1) divide by 4 and add 4; (2) multiply by 2. Check: 2048/4+4=516; 516×2=1032; 1032/4+4=262; then 262×2=524; 524/4+4=135; 135×2=270. Hence the missing term is 524.
	1, 2, 6, 15, ..., 56, 92 **Ground Truth:** 31	1, c81e728d9d4c2f636f067f89cc14862c, 1679091c5a880faf6fb5e6087eb1b2dc, 15, ..., 56, 92	**141**	Decode the hashes as MD5 of single digits: c81e728d9d4c2f636f067f89cc14862c = 2 and 1679091c5a880faf6fb5e6087eb1b2dc = 6. So the visible sequence is 1, 2, 6, 15, ..., 56, 92. The differences form 1, 4, 9, 16, 25, 36, i.e., consecutive squares 1^2 through 6^2. Equivalently, $a_n = a_{n-1} + n^2$ with $a_1 = 1$. Thus the hidden term between 15 and 56 is 31 (15 + 16). Continuing, the next increment is $7^2 = 49$, giving 92 + 49 = 141. MD5('2') and MD5('6') are standard, widely known values, confirming the intended decoding. Summing consecutive squares accumulates as 1, 1+4, 1+4+9, ... producing 1, 2, 6, 15, 31, 56, 92. Each step adds the next square in order.
Obfus Direction Sense: Given a navigation path with conditional turns, determine the final position with a brief justification.	From point A, a traveler moves 5 km North. From there, he proceeds 3 km East. Relative to point A, what is the final direction and distance of the traveler? **Ground Truth:** 5.83 km away, North-East	A man walks 6 km North. If he sees a shop, he must turn right; if not, he must turn left. He sees a shop, so he turns right and goes 4 km. Then, as per his rule, he again turns right and walks 6 km. Finally, since he sees a park, he turns left and walks 3 km. He finds a wall there and comes back for 4 km. At that spot, he looks again. This time he does not see a shop, so he turns right and walks 5 km. Where is he now?	**5.83 km away, North-East**	Step-by-step (taking right/left relative to current heading): Start at (0,0). Walk 6 km North to (0,6). He sees a shop, so turn right (facing North → right is East) and go 4 km to (4,6). As per the rule, he again turns right (East → right is South) and walks 6 km to (4,0). Seeing a park, he turns left (South → left is East) and goes 3 km to (7,0). Finding a wall, he comes back 4 km along the same path (West) to (3,0). There he does not see a shop, but as stated he turns right (West → right is North) and walks 5 km to (3,5). Net displacement: 3 km East and 5 km North; distance = $\sqrt{3^2 + 5^2} \approx 5.83$ km, direction North-East.
	Shahid and Rohit start from the same point in opposite directions. After each 1 km, Shahid always turns left, Rohit always turns right. When do they first meet? **Ground Truth:** 3 km	Shahid and Rohit meet at sunrise at the old banyan tree. After each 1 km, Shahid always turns left, Rohit always turns right. Both walk 1 km. They glance at the fields. They turn, and walk 1 km. A child waves from a bicycle. Turn again and walk 1 km. They pass a vendor selling sugarcane. Turn and walk 1 km. A dog barks; they both stop to shoo it away (no walking during the stop). Turn and walk 1 km. They reach a small pond; Shahid tosses a pebble in, Rohit counts ripples. Turn and walk 1 km. They pass an old woman knitting by the roadside. Turn and walk 1 km. A milkman slows them to ask directions (no extra distance). Turn and walk 1 km. They cross a narrow bridge; Rohit hears church bells. After how many 1-km moves (i.e., after how many kilometers walked by each) do Shahid and Rohit first meet?	**4 km**	Assume they start facing North. Move 1: both walk 1 km North together; they haven't separated yet. Thereafter Shahid always turns left and Rohit always turns right after each kilometer. Shahid's sequence of headings cycles N → W → S → E → N, while Rohit's is N → E → S → W → N. Positions after each move from the origin (0,0): after 1 km both at (0,1). After 2 km: Shahid at (-1,1), Rohit at (1,1). After 3 km: Shahid (-1,0), Rohit (1,0). After 4 km: both return to (0,0), the starting point. This is their first meeting after they have turned in opposite directions. The intermediate incidents add no distance, so the first re-meeting is after 4 km.

A.5 Logifus Algorithms

We list the algorithms (prompts and hardcoded programs) for all four reasoning tasks in Logifus, thereby yielding LogiQAte after logical obfuscation. Each algorithm is processed through GPT-4o with a generation temperature of 0.0 to ensure deterministic outputs. A subsequent review process involves manual verification by two independent human reviewers. For Obfus FOL, each generated instance is first validated using the Prover9 (McCune, 2005–2010) theorem prover to confirm logical equivalence, followed by verification by the same two human reviewers to ensure correctness.

Prompt used for Simplifying FOLIO Questions in Obfus FOL

Task: Transform a FOLIO dataset row into a simplified logical reasoning problem with a ”True” or ”False” label.
Input Structure: Premises NL: premises nl
Premises FOL: premises fol
Conclusion NL: conclusion nl
Conclusion FOL: conclusion fol
Label: label

Selection Rules:
1. Only process rows where the label is ”True” or ”False” (exclude any ”Uncertain” cases)
2. Select any two premises from Premises NL that form a logical relationship with each other
3. Choose any one statement from the remaining Premises NL or use the original Conclusion NL as your new conclusion
4. Extract the corresponding formulas from Premises FOL and Conclusion FOL for your selections

Simplification Guidelines:
1. Rewrite each chosen premise and conclusion to be brief, clear, shorter than the original, and easy to understand
2. Preserve all domain-specific terms and naming conventions
3. Focus on making the logical relationship easy to understand
4. Model the style after simple examples like: ”Tanya is older than Eric. Cliff is older than Tanya.”

Output Format:
1. Two simplified natural language premises labeled as ”Premise 1:” and ”Premise 2:”
2. One simplified natural language conclusion labeled as ”Conclusion:”
3. The question: ”If the first two statements are true, is the third statement true or false?”
4. A composite FOL representation: ([Premise1 FOL]) ([Premise2 FOL]) $\rightarrow$ ([Conclusion FOL])
5. The final label: ”True” or ”False”

Label Determination:
- The output label should reflect whether the chosen conclusion logically follows from the two premises
- Aim to preserve the original logical relationship implied by the dataset when possible

Produce only the requested output with no additional explanations.

Logical Obfuscation Prompt used in Obfus FOL

You are an expert in logical obfuscation. Your task is to take the input dataset row and create a complex, logically obfuscated version of the premises while preserving the semantic meaning and ground truth relationship with the conclusion.

Input: Premises (Natural Language): premises; Premises (FOL): premises_fol; Conclusion (Natural Language): conclusion; Conclusion (FOL): conclusion_fol; Label: label

Core Requirements:
1. Obfuscate ONLY the premises - do NOT modify the conclusion.
2. Preserve the ”True” or ”False” label relationship between the obfuscated premises and the original conclusion.
3. Maintain ALL domain-specific vocabulary (words like ”employee,” ”customer,” ”meeting,” etc.).
4. Keep ALL proper names and entity references (like ”James,” ”Bonnie,” etc.) exactly as they appear.
5. Transform ONLY the logical structure and connectors using highly nested and compound logical transformations.

Logical Obfuscation Prompt used in Obfus FOL - Continued

Logical Transformation Methods (apply at least FOUR per premise, chaining and nesting them together):

- Contraposition ("if P then Q" → "if not Q then not P"); Double negation ("P" → "not not P").
- Logical equivalences ("P or Q" → "not(not P and not Q)"); Conditional to disjunction ("if P then Q" → "not P or Q").
- De Morgan's laws ("not (P and Q)" → "not P or not Q"); Quantifier replacement ("x P(x)" → "¬x ¬P(x)").
- Biconditional expansion ("P Q" → "(P → Q) (Q → P)"); Distribution ("P (Q R)" → "(P Q) (P R)").
- Absorption ("P (P Q)" → "P"); Conditional as conjunction ("P → Q" → "¬(P ¬Q)").
- Negation Normal Form ("¬(P → Q)" → "P ¬Q"); Quantifier commutation ("x y P(x,y)" → "y x P(x,y)").
- Redundant tautology ("P" → "P (Q ¬Q)"); Nested compound transformations; Introduce intermediate statements.

FOL Transformation Guidelines: Maintain all predicate names (e.g., Meeting, LunchInCompany) and constants (e.g., bonnie) exactly as they appear. Do NOT rename or replace any predicates or constants with symbols. Only modify the logical connectives (→, , , ¬, ,) and their arrangement. Ensure strict logical equivalence between original and obfuscated FOL. Use multiple nested logical equivalences per premise to strongly obscure the original logical structure.

Output Format:

Obfuscated Premises (Natural Language): ["⟨Obfuscated NL premise 1⟩", "⟨Obfuscated NL premise 2⟩", . . .]
Obfuscated Premises (FOL): ["⟨Obfuscated FOL premise 1⟩", "⟨Obfuscated FOL premise 2⟩", . . .]
Conclusion (Natural Language): conclusion; Conclusion (FOL): conclusion_fol; Label: label

Remember: The obfuscation should deeply mask the logical reasoning path, making it significantly less obvious while strictly maintaining logical equivalence and the original truth-value relationship with the conclusion. Produce ONLY the exact output format specified above with no introductory text, explanations, reasoning, or comments before or after the requested output.

Logical Obfuscation Algorithm used in Obfus Number Series (Hardcoded)

Input: Base numerical sequence $S = \{n_1, n_2, \ldots, n_k\}$ with underlying pattern

Planet Mapping: $\mathcal{P}$: Sun → 0, Mercury → 1, Venus → 2, Earth → 3, Mars → 4, Jupiter → 5, Saturn → 6, Uranus → 7, Neptune → 8, Pluto → 9.

Output: Three obfuscated sequences S_1, S_2, S_3 with preserved pattern

Type 1: Planet-Name Substitution

- Initialize $S_1 \leftarrow \emptyset$
- For each $n_i \in S$, add $\mathcal{P}[n_i]$ to S_1
- **Return:** $S_1 = \{\mathcal{P}[n_1], \mathcal{P}[n_2], \ldots, \mathcal{P}[n_k]\}$
- *Example:* $[2, 4, 6, 8] \rightarrow \{\text{Venus}, \text{Mars}, \text{Jupiter}, \text{Saturn}\}$

Type 2: ASCII Sum Substitution

- Initialize $S_2 \leftarrow \emptyset$
- For each $n_i \in S$: planet $\leftarrow \mathcal{P}[n_i]$; ascii_sum $\leftarrow \sum_{c \in \text{planet}} \text{ASCII}(c)$; $S_2 \leftarrow S_2 \cup \{\text{ascii_sum}\}$
- **Return:** $S_2 = \{f_{\text{ASCII}}(\mathcal{P}[n_1]), \ldots, f_{\text{ASCII}}(\mathcal{P}[n_k])\}$
- *Example:* Mars → $77 + 97 + 114 + 115 = 403$

951

Logical Obfuscation Algorithm used in Obfus Number Series (Hardcoded) - Continued

Type 3: MD5 Hash Substitution

- Initialize $S_3 \leftarrow \emptyset$
- For each $n_i \in S$: hash $\leftarrow$ MD5(str(n_i)); $S_3 \leftarrow S_3 \cup \{\text{hash}\}$
- **Return:** $S_3 = \{\text{MD5}(\text{str}(n_1)), \ldots, \text{MD5}(\text{str}(n_k))\}$
- *Example:* $4 \rightarrow$ `a87ff679a2f3e71d9181a67b7542122c`

Dataset Construction:

- Collect base sequences from standard aptitude sources
- For each base sequence S with answer a: Generate S_1, S_2, S_3; verify recoverability; add $(S, a), (S_1, a), (S_2, a), (S_3, a)$ to $\mathcal{D}$
- **Return:** Dataset $\mathcal{D}$ containing base and obfuscated sequence pairs

Logical Obfuscation Prompt used in Obfus Blood Relation (Level 1 and Level 2)

You are a blood relation puzzle assistant.

Below are the substitution rules you must follow when modifying the puzzle:

Level 1 Substitutions:
Father → mother's husband, grandfather's only son
Mother → Father's wife, grandfather's only daughter-in-law
Brother → sister-in-law's husband
Sister → Brother-in-law's wife
niece → Brother's daughter, sister-in-law's daughter, Father's granddaughter
nephew → Brother's son, sister-in-law's son, Father's grandson
Aunt → Father's sister, Grandfather's daughter, mother's sister-in-law
Uncle → Father's brother, Grandfather's son, mother's brother-in-law
Maternal Uncle → mother's brother, Father's brother-in-law, maternal grandmother's son
Paternal Uncle → Father's brother, cousin's father
Maternal Aunt → mother's sister
Sister-in-law → Brother's wife, niece's mother
Brother-in-law → Sister's husband
Grand Father → father's father, mother's father-in-law
Grand Mother → Father's mother, mother's mother-in-law
Daughter-in-law → son's wife
Son-in-law → Daughter's husband
cousin sister → uncle's daughter
cousin brother → aunt's son
son → Father's grandson
daughter → mother's granddaughter
grandson → son's son
granddaughter → son's daughter

Logical Obfuscation Prompt used in Obfus Blood Relation (Level 1 and Level 2) - Continued

Level 2 Substitutions:
Father → greatgrandfather's only grandson
Mother → greatgrandfather's only grandson's wife
Brother → greatgrandfather's only grandson's daughterinlaw's husband
Sister → greatgrandfather's only grandson's daughter
niece → Father's granddaughter
nephew → Father's grandson
Aunt → Greatgrandfather's granddaughter, Grandfather's daughter
Uncle → greatgrandfather's only grand daughter inlaw's brother inlaw
Maternal Uncle → maternal grandmother's son
Maternal Aunt → greatgrandfather's only grand daughter inlaw's sister
Sister inlaw → greatgrandfather's only grandson's daughterinlaw
Brother inlaw → greatgrandfather's only grandson's daughter's husband
Grand Father → greatgrandfather's only son
Grand Mother → greatgrandfather's only daughter inlaw
Daughter inlaw → father inlaw's son's wife
Son inlaw → Daughter's husband
cousin sister → uncle's daughter
cousin brother → aunt's son
son → Father's grandson
daughter → mother's granddaughter
grandson → son's son
granddaughter → son's daughter

Your task:
1. Carefully read the puzzle below.
2. Identify exactly one blood relation word that appears in the puzzle and is listed above.
3. Choose one of the allowed substitutions for that word.
4. Replace that relation word in the puzzle with the selected substitution — only once.
5. Print the modified puzzle, do not add header and footer or any other explanation.

Puzzle:
""" {puzzle} """

Logical Obfuscation Prompt used in Obfus Direction Sense

Direction sense : You are a direction-sense puzzle maker.

Below are the transformation rules you must follow when modifying the puzzle:

1. Direction Substitution Rules:
- North → facing 12 o'clock
- South → facing 6 o'clock
- East → facing 3 o'clock
- West → facing 9 o'clock

2. Rotation Substitution Rules:
- "turn right" → "rotate 1 quarter-turn clockwise"
- "turn left" → "rotate 1 quarter-turn counterclockwise"
- "turn back" → "rotate 2 quarter-turns"
- "faces opposite direction" → "rotates 2 quarter-turns"

3. Apply Zigzag Cancellation Rule:
- Add an equal distance in opposite directions (e.g., 5 km North and 5 km South), so that it must cancel out the back-and-forth movement.

Your task:
1. Read the base puzzle below.
2. Apply the above transformation and simplification rules.
3. Ensure that the logical movement pattern and final displacement remain the same.
4. Output only the modified (obfuscated) puzzle text — no explanation, no headers or footers.

Puzzle:
""" {puzzle} """