Task 7: Logical Inference

This document describes the backward chaining system implemented for Task 7 of the LLM+Logic project. The goal of the task was to build a simple logical inference engine for First Order Logic (FOL) using the backward chaining approach. The implementation was done from scratch in Python.

The system uses a minimal representation of knowledge bases. Facts and rules are provided as strings. Each rule has a body (the premises) and a head (the conclusion). The program includes functions for parsing predicates, performing unification, applying substitutions, and recursively attempting to prove a goal based on known facts and applicable rules.

For example, given facts such as:

```
Unset
parent(john, mary)
parent(mary, alice)
```

and rules like:

```
Unset
parent(x, y) => ancestor(x, y)
ancestor(x, y) and parent(y, z) => ancestor(x, z)
```

the system is able to correctly infer that:

```
Unset
ancestor(john, alice)
```

is true.

The main function, `bc_ask`, is goal-driven. It first checks whether a query matches any known facts. If not, it looks for rules whose head matches the query and then recursively attempts to prove each condition in the rule's body. Unification is handled for simple variable matching, and substitutions are applied consistently.

Testing was performed using a set of manually defined cases stored in `test_cases.txt`. These include both direct and chained inferences, as well as negative cases where a query

cannot be proven. The system behaves as expected and returns accurate results in all tested cases.

The project structure includes `backward_chaining.py` for the logic engine, `test_cases.txt` for sample input, and this write-up located in `tutorial/task7_writeup.pdf`. All materials are available at: https://github.com/suvalavala/llm-logic-task7

This task helped reinforce core ideas behind symbolic reasoning, recursion, and simple rule engines. While basic, the current implementation can serve as a foundation for extending to more complex logic systems or integration with language models in future tasks.