CPS 721 - Q8 (LLM Prolog Coding Report)

Akif Sipahi
Enes Polat
Ekrem Yilmaz

For this bonus question, I chose to use the LLM to solve Q2 from Assignment 1. I used the LLM to generate Prolog code for the given problem.

LLM Used: I used a transformer-based LLM, specifically the T5 model, to generate the Prolog code.

Process: I provided the LLM with the problem statement and the given facts, and asked it to generate the Prolog code for the rules. The LLM generated the code, which I then tested using the provided queries.

What the LLM Got Right: The LLM correctly generated the Prolog code for the rules, including the subtotal, accruedInterest, accruedPenalty, and endOfMonthBalance rules. The LLM also correctly handled the queries, producing the expected results for most of the queries.

What the LLM Got Wrong: The LLM had some issues with the endOfMonthBalancerule, which calculates the end-of-month balance for a person across all banks. The LLM generated code that only considered one bank, rather than both banks. I had to adjust the code to fix this issue.

```
endOfMonthBalance(Name, Bank, Balance) :-
   subtotal(Name, Bank, Subtotal),
   accruedInterest(Name, Bank, Interest),
   accruedPenalty(Name, Bank, Penalty),
   Balance is Subtotal + Interest + Penalty.
```

Adjustments Made: I made several adjustments to the LLM-generated code to improve its accuracy. Specifically, I:
- Modified the endOfMonthBalance rule to consider both banks.
- Added additional queries to test the code more thoroughly.
- Adjusted the code to handle cases where the subtotal is negative.

Testing: I tested the LLM-generated code using the provided queries, as well as additional queries that I created to test the code more thoroughly. The code produced the expected results for most of the queries, but had some issues with the endOfMonthBalance rule.

Conclusion: Overall, the LLM performed well in generating Prolog code for the given problem. However, it required some adjustments to produce accurate results. The LLM's ability to generate code quickly and efficiently was helpful, but it was not perfect and required human oversight to ensure accuracy.