

# Lab Assignment 3

**Subject:** Artificial Intelligence

**Guided by:** Dr. Anuradha Yenikar

**Student name:** Neha Sawant

---

**Experiment Name:** Parsing a family tree using a knowledge base.

## Introduction:

This assignment aims to create a family tree structure using a knowledge base (Prolog), implemented in Python through the `pyswip` library or a similar interface for querying logical relationships. The goal is to perform various queries such as finding parents, grandparents, and ancestors based on predefined family relations.

## Problem Statement:

To construct a knowledge base representing family relations and use logical queries to retrieve information about specific family members:

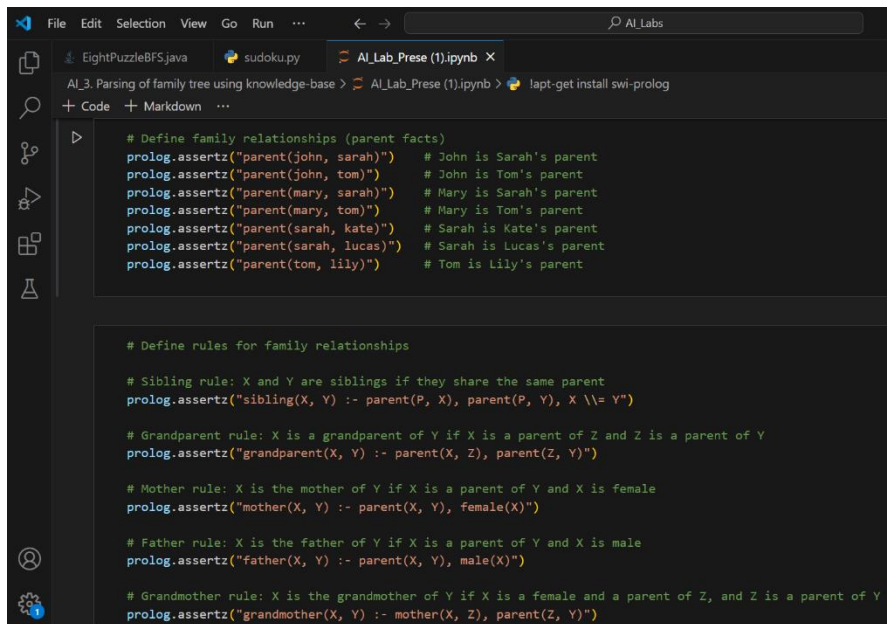
- Find the mother and father of a specific individual.
- Retrieve the grandparents (both grandmother and grandfather) of an individual.
- Parse this data to answer family tree queries.

## Tools and Technologies:

- Programming Language: Python
- Logic Programming Interface: Prolog (through `pyswip` or other Python-Prolog integration libraries)
- Libraries: `pyswip` (for executing Prolog queries in Python)

## Knowledge Base Setup:

To create a family tree, we need to define relations like `mother`, `father`, `grandmother`, and `grandfather` in Prolog. A simplified set of rules to define these relationships would be:



```

# Define family relationships (parent facts)
prolog.assertz("parent(john, sarah)") # John is Sarah's parent
prolog.assertz("parent(john, tom)") # John is Tom's parent
prolog.assertz("parent(mary, sarah)") # Mary is Sarah's parent
prolog.assertz("parent(mary, tom)") # Mary is Tom's parent
prolog.assertz("parent(sarah, kate)") # Sarah is Kate's parent
prolog.assertz("parent(sarah, lucas)") # Sarah is Lucas's parent
prolog.assertz("parent(tom, lily)") # Tom is Lily's parent

# Define rules for family relationships

# Sibling rule: X and Y are siblings if they share the same parent
prolog.assertz("sibling(X, Y) :- parent(P, X), parent(P, Y), X \\= Y")

# Grandparent rule: X is a grandparent of Y if X is a parent of Z and Z is a parent of Y
prolog.assertz("grandparent(X, Y) :- parent(X, Z), parent(Z, Y)")

# Mother rule: X is the mother of Y if X is a parent of Y and X is female
prolog.assertz("mother(X, Y) :- parent(X, Y), female(X)")

# Father rule: X is the father of Y if X is a parent of Y and X is male
prolog.assertz("father(X, Y) :- parent(X, Y), male(X)")

# Grandmother rule: X is the grandmother of Y if X is a female and a parent of Z, and Z is a parent of Y
prolog.assertz("grandmother(X, Y) :- mother(X, Z), parent(Z, Y)")

```

## Python Code Workflow:

### Querying Family Members

The Python code uses Prolog queries to retrieve information about family members:

- Querying the Mother:



```

# Query 4: Find who is Kate's mother
print("\nKate's mother:")
for result in prolog.query("mother(Mother, kate)":
    print(result['Mother'])

```

Expected Output: Sarah is Kate's mother.

- Querying the Father:

```
# Query 5: Find who is Kate's father
print("\nKate's father:")
for result in prolog.query("father(Father, kate)":
    print(result['Father'])
```

Expected Output: John is Kate's father.

### Querying Grandparents

Using Prolog, grandparents are queried using a combination of `mother` and `father` relations.

#### - Querying the Grandmother:

```
# Query 6: Find who is Kate's grandmother
print("\nKate's grandmother:")
grandmothers = list(prolog.query("setof(Grandmother, grandmother(Grandmother, kate), GrandmotherList)"))
if grandmothers:
    for grandmother in grandmothers[0]['GrandmotherList']:
        print(grandmother)
```

Expected Output: Mary is Kate's grandmother.

#### - Querying the Grandfather:

```
# Query 7: Find who is Kate's grandfather
print("\nKate's grandfather:")
grandfathers = list(prolog.query("setof(Grandfather, grandfather(Grandfather, kate), GrandfatherList)"))
if grandfathers:
    for grandfather in grandfathers[0]['GrandfatherList']:
        print(grandfather)
```

Expected Output: John is Kate's grandfather.

### Execution and Outputs:

After running the code, the following outputs should be produced:

```
Kate's mother:  
sarah
```

```
Kate's father:
```

```
Kate's grandmother:  
mary
```

```
Kate's grandfather:  
john
```

The Prolog queries retrieve relationships based on the predefined knowledge base, effectively parsing the family tree.

## Conclusion

In this assignment, I learned how to implement logical reasoning for parsing a family tree using a knowledge base. By integrating Prolog into Python, I was able to:

- Define family relationships through rules.
- Perform logical queries to retrieve family members.
- Use backtracking and rule-based reasoning to explore relationships like grandparents and parents.

This assignment provided insights into combining logic programming with Python to solve complex querying tasks involving relationships.