**Assignment 3**

**Problem Statement:** Perform parsing of family tree using knowledge-base.

**Library:**

1. Python Libraries:
   * collections: Useful for creating data structures like dictionaries and lists to represent the family tree.
   * re (Regular Expressions): For pattern matching and identifying relationships within the family tree data.
2. Prolog/Logical Libraries (Optional):
   * Prolog-style logic programming libraries in Python such as PySWIP or Kanren could be used to model family relations logically.
3. Graph-based Libraries:
   * networkx: To represent the family tree as a graph where nodes are family members and edges represent relationships.
4. Custom Knowledge-base:
   * A custom structure or database containing predefined family relationships (e.g., who is a parent of whom) for efficient querying.

**Theory:**

A family tree is a type of hierarchical data structure where nodes represent family members, and edges represent relationships between them, such as parent-child, sibling, or spouse. A knowledge-base can be used to store facts about the family tree, allowing us to query and infer new facts based on existing data.

Parsing refers to the process of analyzing a structured input (in this case, the family tree data) and breaking it down into components that can be interpreted by a machine. The parsing process identifies:

* Nodes: Family members.
* Edges/Relations: How each family member is related to others.

For example, a knowledge base can store rules such as:

* If A is the parent of B, then B is the child of A.
* If A and B have the same parents, they are siblings.

By parsing the family tree and querying this knowledge base, we can answer complex questions about family relationships.

**Methodology:**

1. Input Data (Family Tree):
   * Start by defining the family tree in a structured format (e.g., a dictionary or a graph where nodes represent people and edges represent relationships).
2. Knowledge-Base Construction:
   * Build a knowledge-base that contains predefined rules and relationships. For instance, "parent(X, Y)" means X is the parent of Y, and "sibling(X, Y)" means X and Y share the same parents.
3. Parsing Algorithm:
   * Use a recursive function or a depth-first search (DFS) approach to traverse the family tree, extract relationships, and populate the knowledge-base.
4. Query System:
   * Implement a querying system to answer questions like:
     + Who is the parent of X?
     + Who are the siblings of Y?
     + What is the relationship between X and Y?
5. Relationship Inference:
   * Using the rules defined in the knowledge-base, infer new relationships. For example, if X is a parent of Y and Y is a parent of Z, then X is the grandparent of Z.
6. Output:
   * The final system should be able to output family relations when queried, such as:
     + "X is the parent of Y."
     + "X is the grandparent of Z."

**Advantages:**

1. Efficient Querying: Once the knowledge-base is built, querying relationships becomes fast and efficient.
2. Clear Representation: The hierarchical structure of the family tree is easy to visualize and understand.
3. Scalability: The system can easily scale to larger family trees by adding more nodes and edges without changing the core logic.
4. Logical Inference: A knowledge-base allows us to infer new relationships from existing facts, which might not be explicitly provided.

**Disadvantages:**

1. Complexity for Large Families: As the family tree grows, the time and space complexity of parsing and querying can increase significantly.
2. Ambiguity in Relationships: In cases of complex family structures (e.g., with adopted children or step-siblings), the relationships may be difficult to model accurately.
3. Knowledge-base Maintenance: Updating the knowledge-base to include new family members or changing relationships can be tedious.

**Conclusion:**

Parsing a family tree using a knowledge-base provides a structured and logical approach to understanding relationships between family members. By defining a set of rules and facts, we can efficiently query and infer new relationships, making it easier to navigate complex family structures. Although challenges arise with scalability and complex family dynamics, the use of a knowledge-base offers a robust solution for managing hierarchical relationships in family trees.