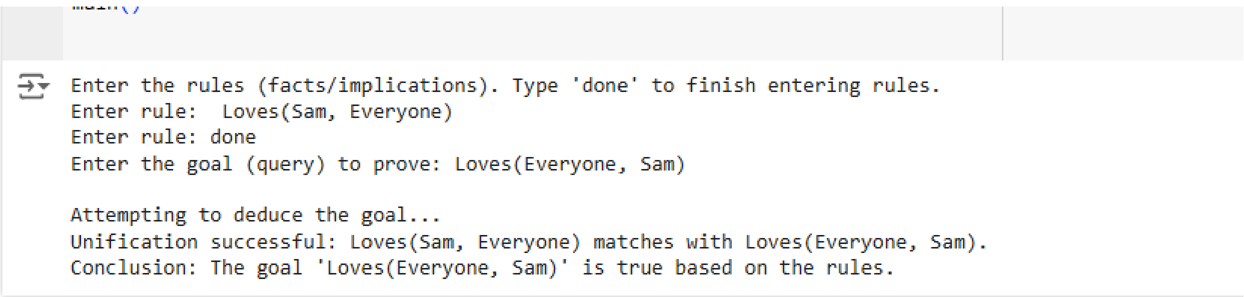
Lab 8 Fol

code:-

```
import re

# Define a simple function for extracting predicates from sentences
def extract_predicate(sentence):
    # Regular expression to find patterns like Predicate(Argument)
    pattern = r"([A-Za-z]+)\((\w+)\)"
    match = re.search(pattern, sentence)

    if match:
        predicate = match.group(1)
        subject = match.group(2)
        return predicate, subject
    return None, None

# Function for unification
def unify(fact, query):
    # Check if the fact and query are the same
    if fact == query:
        return True
    # Extract predicate and subject from fact and query
    fact_predicate, fact_subject = extract_predicate(fact)
    query_predicate, query_subject = extract_predicate(query)
    # If predicates match, unify the subjects
    if fact_predicate == query_predicate:
        if fact_subject == query_subject:
            return True
        else:
            # Here, we could handle variable substitution (unification)
            return False
    return False

# Function to deduce the goal using given rules
def deduct(rules, goal):
    # Try to find unification for the goal from the rules
    for rule in rules:
        if unify(rule, goal):
            print(f"Unification successful: {rule} matches with {goal}.")
            return True
    return False

# Main function to handle user input
def main():
    # Step 1: Get the rules (facts/implications) from the user
    print("Enter the rules (facts/implications). Type 'done' to finish entering rules.")
    rules = []
    while True:
        rule_input = input("Enter rule: ")
        if rule_input.lower() == 'done':
            break
        else:
            rules.append(rule_input.strip())

    # Step 2: Get the goal (query) from the user
    goal_input = input("Enter the goal (query) to prove: ").strip()

    # Step 3: Try to deduce the goal using the given rules
    print("\nAttempting to deduce the goal...")
    if deduct(rules, goal_input):
        print(f"Conclusion: The goal '{goal_input}' is true based on the rules.")
    else:
        print(f"Conclusion: The goal '{goal_input}' cannot be proven with the provided rules.")

# Run the program
main()
```

output:-



```
main()
```

```
Enter the rules (facts/implications). Type 'done' to finish entering rules.
Enter rule:  Loves(Sam, Everyone)
Enter rule: done
Enter the goal (query) to prove: Loves(Everyone, Sam)

Attempting to deduce the goal...
Unification successful: Loves(Sam, Everyone) matches with Loves(Everyone, Sam).
Conclusion: The goal 'Loves(Everyone, Sam)' is true based on the rules.
```

observation book

LAB-8.

FOL

Key elements of FOL:-

Predicates:- Represent's relationships between objects.

Terms:- Objects or variables in the domain being described.

Quantifers:-

$\forall$ - Universal quantifier

$\exists$ - Existential quantifer.

$\wedge$(AND), ($\vee$)OR, NOT, $\neg$ $\rightarrow$ implies, $\leftrightarrow$ Equivalent - logical conectives.

Example of FOL in AI:

If all humans are mortal, and Socrates is a human, then So

Now we will prove for

If someone loves everyone, then everyone is loved by that someone.

1. Premise:- There exists a person x who loves everyone y.

$\exists x(\forall y \text{ loves}(x,y))$

2. Conclusion: $\forall y (\exists x \text{ loves}(x,y))$

(For every person y, there exist someone x who loves y.)

1) Solution steps:-

1. Assume the Premise:

From the premise $\exists x (\forall y \text{ loves}(x,y))$, we known there exists a specific individual a such that: $\forall y$ loves(a

. Universal Instantiation:

From $\forall y$ loves (a,y) instantiate for a specific individual b:

loves (a, b)

This holds for any b because a loves ev

3. Existential Generalization:

Since loves (a, b) is true for any individual b, we can say for any person b: ∃x Loves (x, b) (because a is the one who loves b).

4. Universal Generalization:

Since this holds for any individual b, we generalize:

$\forall y (\exists x \; Loves(x, y))$.

Conclusion:-

By applying universal instantiation, existential generalization, and universal generalization, we deduce: "If someone loves everyone, then everyone is loved by that Someone".

Output:-

```
Enter the rules (facts/implications).
    Type 'done' to finish rules
Enter rule: loves (Sam, Everyone)
Enter rule: Done
Enter your goal (query) to prove : loves(Every
  Sam).
Attempting to deduce the goal...
Unification successful : loves ( Sam, Everyone)
          matches with loves (Everyone, Sam).
Conclusion : The goal 'Love (Everyone, Sam'
    is true based on the rules
```

19/11/24