**Forward Chaining Algorithm**

**Aim**

To implement the forward chaining algorithm using Python, in order to determine if a goal can be proven based on a rule-based system and a given set of known facts.

**Procedure**

1. Define the rule base where each rule contains a list of conditions ("if") and a conclusion ("then").
2. Initialize a set of known facts.
3. Accept a goal from the user.
4. Use a loop to:
   * Iterate through all rules.
   * Apply rules whose conditions are all satisfied by current facts.
   * Infer and add new facts based on applied rules.
   * Stop if the goal is inferred or no new inferences are possible.
5. Display the result indicating whether the goal has been proven.

**Program**

rules = [

{"if": ["A", "B"], "then": "C"},

{"if": ["C", "D"], "then": "E"},

{"if": ["E"], "then": "F"},

]

facts = {"A", "B", "D"}

goal = input("Enter the goal to prove (e.g., E): ").strip().upper()

def forward\_chain(rules, facts, goal):

inferred = set()

while True:

applied = False

for rule in rules:

if rule["then"] not in facts:

if all(condition in facts for condition in rule["if"]):

facts.add(rule["then"])

inferred.add(rule["then"])

applied = True

print(f"Inferred: {rule['then']} from {rule['if']}")

if rule["then"] == goal:

return True

if not applied:

break

return goal in facts

if forward\_chain(rules, facts, goal):

print(f"YES, '{goal}' can be proven.")

else:

print(f"NO, '{goal}' cannot be proven with current knowledge.")

**Input**

Enter the goal to prove (e.g., E): F

**Output**

Inferred: C from ['A', 'B']

Inferred: E from ['C', 'D']

Inferred: F from ['E']

YES, 'F' can be proven.

**Result**

The program successfully demonstrates forward chaining by applying applicable rules and inferring new facts step by step. It proves whether the goal can be reached based on the initial facts and the defined rules.