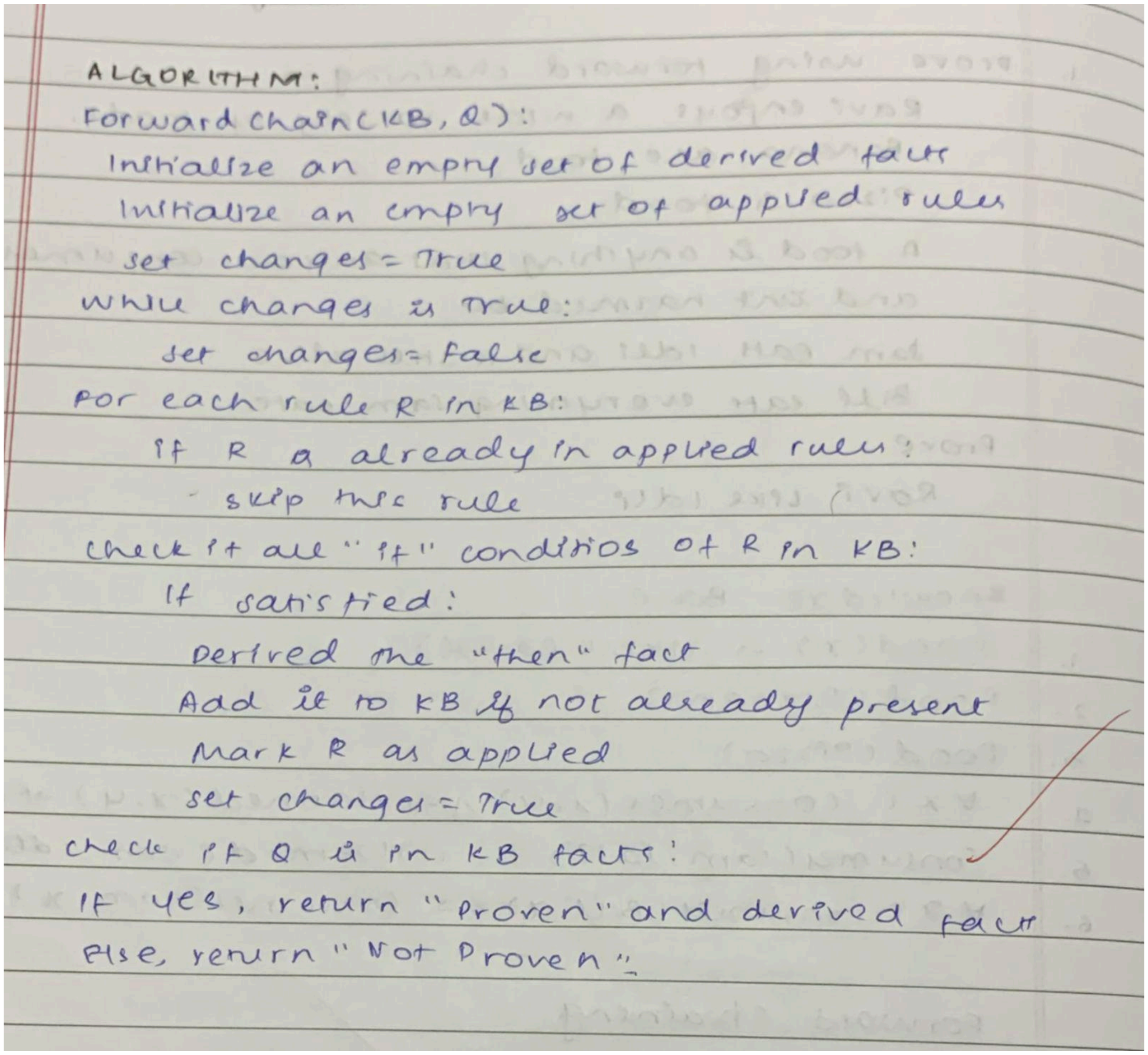
03-12-2024 WEEK-8

KNOWLEDGE BASE USING FORWARD REASONING

ALGORITHM-

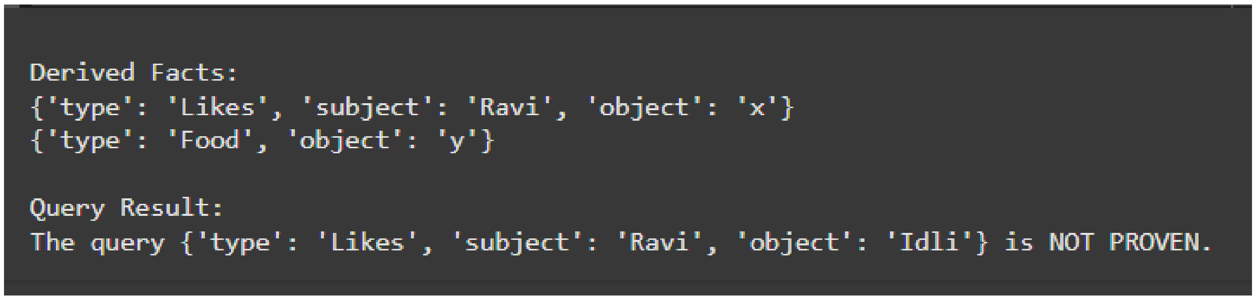


```
ALGORITHM:
Forward Chain(KB, Q):
  Initialize an empty set of derived facts
  Initialize an empty set of applied rules
  set changes = True
while changes is True:
    set changes = False
for each rule R in KB:
    if R a already in applied rules:
      - skip this rule
check if all "if" conditios of R in KB:
    If satisfied:
      Derived the "then" fact
      Add it to KB if not already present
      Mark R as applied
      set changes = True
check if Q is in KB facts:
If yes, return "Proven" and derived facts
Else, return "Not Proven".
```

CODE-

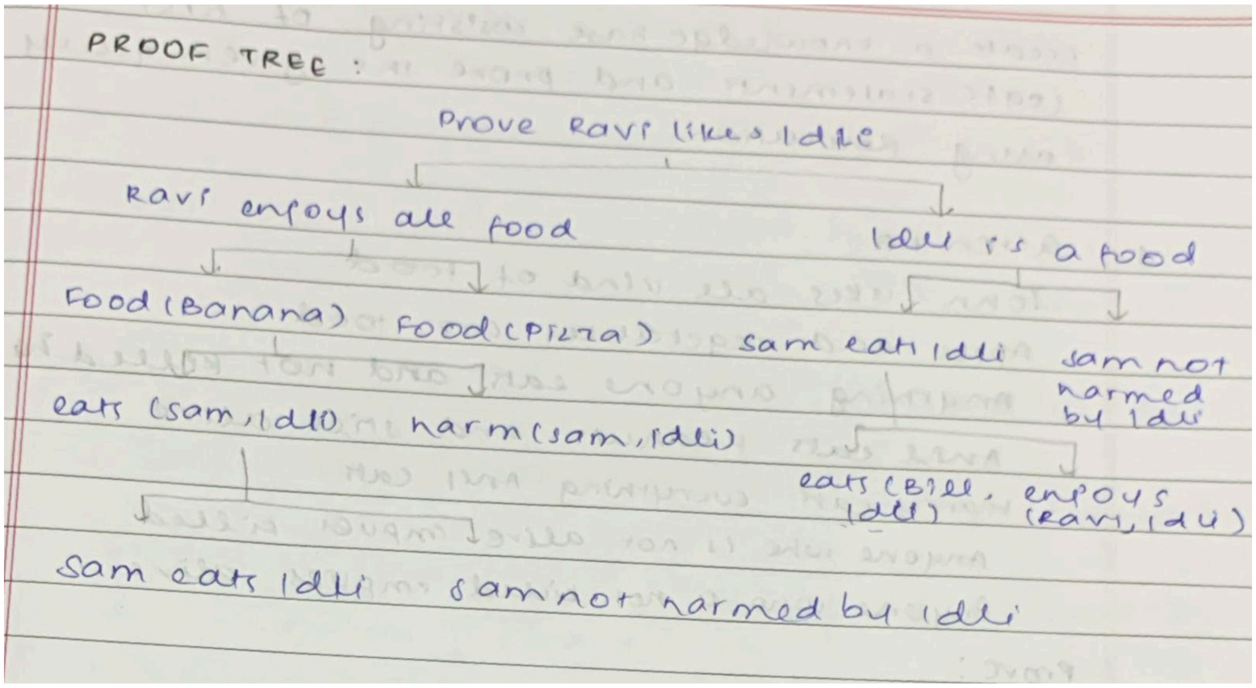
```
knowledge_base = {
  "facts": [
    {"type": "Food", "object": "Banana"},
    {"type": "Food", "object": "Pizza"},
    {"type": "Consumes", "subject": "Sam", "object": "Idli"},
    {"type": "NotHarmed", "subject": "Sam", "object": "Idli"}
  ],
  "rules": [
    {"if": [{"type": "Consumes", "subject": "x", "object": "y"},
         {"type": "NotHarmed", "subject": "x", "object": "y"}],
     "then": {"type": "Food", "object": "y"}},
    {"if": [{"type": "Food", "object": "x"}],
     "then": {"type": "Likes", "subject": "Ravi", "object": "x"}}
  ]
}
def forward_chain(kb, query):
  derived_facts = set()
  applied_rules = set()
  changes = True

  while changes:
    changes = False
    for rule_id, rule in enumerate(kb["rules"]):
      if rule_id in applied_rules:
        continue

      if_conditions = rule["if"]
      satisfied = all(any(all(fact.get(k) == cond.get(k) for k in cond if k != 'object' and k != 
'subject')
                for fact in kb["facts"]) for cond in if_conditions)

      if satisfied:
        applied_rules.add(rule_id)
        derived_fact = rule["then"]
        if not any(all(fact.get(k) == derived_fact.get(k) for k in derived_fact) for fact in 
kb["facts"]):
         kb["facts"].append(derived_fact)
         derived_facts.add(tuple(derived_fact.items()))
         changes = True
    for fact in kb["facts"]:
        if all(fact.get(k) == query.get(k) for k in query):
            return True, derived_facts

    return False, derived_facts

query = {"type": "Likes", "subject": "Ravi", "object": "Idli"}

result, derived_facts = forward_chain(knowledge_base, query)
print("\nDerived Facts:")
for fact in derived_facts:
    print(dict(fact))

print("\nQuery Result:")
if result:
    print(f"The query {query} is PROVEN.")
else:
    print(f"The query {query} is NOT PROVEN.")
```

OUTPUT-



```
Derived Facts:
{'type': 'Likes', 'subject': 'Ravi', 'object': 'x'}
{'type': 'Food', 'object': 'y'}

Query Result:
The query {'type': 'Likes', 'subject': 'Ravi', 'object': 'Idli'} is NOT PROVEN.
```

PROOF TREE-


PROOF TREE :
Prove Ravi likes Idli
Ravi enjoys all food
Idli is a food
Food (Banana)
Food (Pizza)
Sam eats Idli
Sam not harmed by Idli
eats (Sam, Idli)
harm(Sam, Idli)
eats (Bill, Idli)
enjoys (Ravi, Idli)
Sam eats Idli
Sam not harmed by Idli