ADITHIYA.V
241501010

Experiment 6

IMPLEMENTATION OF UNIFICATION AND RESOLUTION ALGORITHM

Aim:

To implement unification and resolution algorithm using python.

Scenario:

In an AI-based expert system for automated reasoning, the system needs to resolve queries by unifying logical predicates and applying resolution inference. For example, given the knowledge base:

🎬 Rule 1: If John is a human, then John is a mortal →
Human(John) → Mortal(John)
🎬 Fact 1: Human(John)
🎬 Query: Is John mortal?

Procedure:

1. Define the unification function (unify):

🎬 If both terms are identical, return the current substitution (theta).
🎬 If one term is a variable, unify it with the other term.
🎬 If both terms are compound expressions, unify their corresponding parts recursively. 🎬 Otherwise, return None (unification fails).

2. Define the variable unification function (unify_var):

🎬 If the variable already exists in the substitution set, apply unification recursively. 🎬 Otherwise, assign the variable to the given term.

3. Define the resolution function (resolution):

🎬 Iterate through the knowledge base (KB).
🎬 Try to unify the given query with KB clauses.
🎬 If unification succeeds, remove matched parts from KB and

recurse with the remaining parts.
🎬 If the knowledge base is empty after resolution, the query is proven. 🎬 Otherwise, return False (query not proven).

4. Provide a knowledge base with facts and implications.

5. Define a query to resolve (e.g., Mortal(John)).

6. Run the resolution function to check if the query can

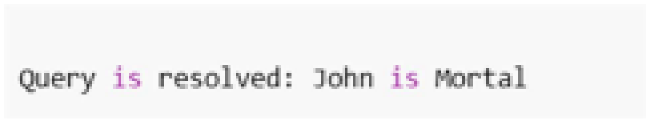
be proven. 7. Print whether the query is resolved.

Program:

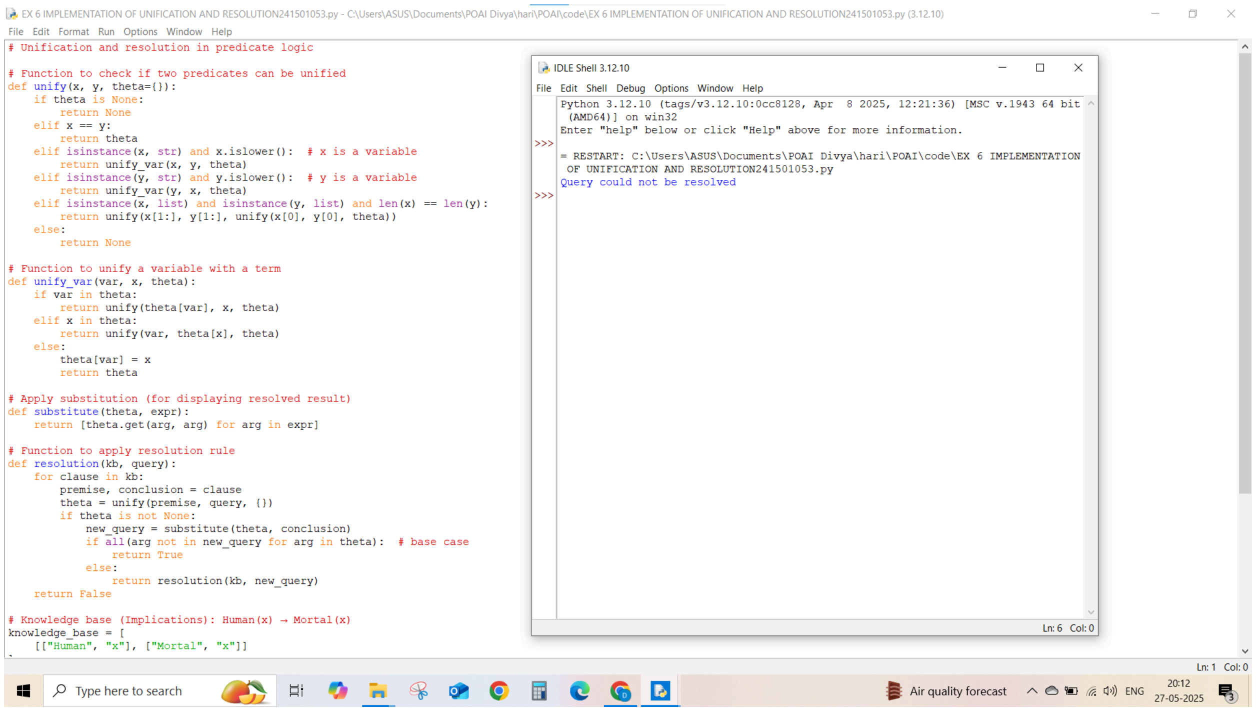
```
import re
# Function to check if two predicates can be unified
def unify(x, y, theta={}):
 if theta is None:
 return None
 elif x == y:
 return theta
 elif isinstance(x, str) and x.islower(): # x is
a variable   return unify_var(x, y, theta)
 elif isinstance(y, str) and y.islower(): # y is
a variable   return unify_var(y, x, theta)
 elif isinstance(x, list) and isinstance(y, list) and
len(x) == len(y):   return unify(x[1:], y[1:], unify(x[0],
y[0], theta))   else:
 return None
# Function to unify a variable with a term
def unify_var(var, x, theta):
 if var in theta:
 return unify(theta[var], x, theta)
 elif x in theta:
 return unify(var, theta[x], theta)
 else:
 theta[var] = x
 return theta
# Function to apply resolution rule
def resolution(kb, query):
 for clause in kb:
 theta = unify(clause[0], query, {})
 if theta is not None:
 new_kb = clause[1:]
 if not new_kb: # If empty, means query is resolved
 return True
 else:
 return resolution(kb, new_kb[0])
 return False
# Knowledge base (Implications)
knowledge_base = [
 [["Human", "John"], ["Mortal", "John"]], # Human(John) →
Mortal(John)  ]
# Fact: Human(John)
fact = ["Human", "John"]

# Query: Mortal(John)?
query = ["Mortal", "John"]
# Apply resolution
if resolution(knowledge_base, query):
 print("Query is resolved: John is Mortal")
else:
 print("Query could not be resolved")
```

Output:



```
Query is resolved: John is Mortal
```



EX 6 IMPLEMENTATION OF UNIFICATION AND RESOLUTION241501053.py - C:\Users\ASUS\Documents\POAI Divya\hari\POAI\code\EX 6 IMPLEMENTATION OF UNIFICATION AND RESOLUTION241501053.py (3.12.10)

```python
# Unification and resolution in predicate logic

# Function to check if two predicates can be unified
def unify(x, y, theta={}):
    if theta is None:
        return None
    elif x == y:
        return theta
    elif isinstance(x, str) and x.islower():  # x is a variable
        return unify_var(x, y, theta)
    elif isinstance(y, str) and y.islower():  # y is a variable
        return unify_var(y, x, theta)
    elif isinstance(x, list) and isinstance(y, list) and len(x) == len(y):
        return unify(x[1:], y[1:], unify(x[0], y[0], theta))
    else:
        return None

# Function to unify a variable with a term
def unify_var(var, x, theta):
    if var in theta:
        return unify(theta[var], x, theta)
    elif x in theta:
        return unify(var, theta[x], theta)
    else:
        theta[var] = x
        return theta

# Apply substitution (for displaying resolved result)
def substitute(theta, expr):
    return [theta.get(arg, arg) for arg in expr]

# Function to apply resolution rule
def resolution(kb, query):
    for clause in kb:
        premise, conclusion = clause
        theta = unify(premise, query, {})
        if theta is not None:
            new_query = substitute(theta, conclusion)
            if all(arg not in new_query for arg in theta):  # base case
                return True
            else:
                return resolution(kb, new_query)
    return False

# Knowledge base (Implications): Human(x) → Mortal(x)
knowledge_base = [
    [["Human", "x"], ["Mortal", "x"]]
```

IDLE Shell 3.12.10

```
Python 3.12.10 (tags/v3.12.10:0cc8128, Apr  8 2025, 12:21:36) [MSC v.1943 64 bit
 (AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
= RESTART: C:\Users\ASUS\Documents\POAI Divya\hari\POAI\code\EX 6 IMPLEMENTATION
 OF UNIFICATION AND RESOLUTION241501053.py
Query could not be resolved
>>>
```

File Edit Format Run Options Window Help
File Edit Shell Debug Options Window Help
Ln: 6 Col: 0
Ln: 1 Col: 0
Type here to search
Air quality forecast
ENG
20:12
27-05-2025