# Function to unify two expressions def unify(x, y, theta=None):

if theta is None:

theta = {} if x == y:

return theta

elif isinstance(x, str) and x.islower(): # variable return unify\_var(x, y, theta)

elif isinstance(y, str) and y.islower(): # variable return unify\_var(y, x, theta)

elif isinstance(x, list) and isinstance(y, list) and len(x) == len(y): for xi, yi in zip(x, y):

theta = unify(xi, yi, theta) if theta is None:

return None return theta

else:

return None

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

# Resolution-like function for simple implication def resolution(kb, query):

for clause in kb:

premise, conclusion = clause

theta = unify(conclusion, query, {}) if theta is not None:

# Check if all premises unify

all\_premises\_true = all(unify(p, fact, theta) is not None for fact in facts for p in [premise]) if all\_premises\_true:

return True return False

# Knowledge base: Implication - Human(John) → Mortal(John) knowledge\_base = [

[["Human", "x"], ["Mortal", "x"]] # generalized implication

]

# Known facts

facts = [["Human", "John"]]

# Query

query = ["Mortal", "John"]

# Check resolution

if resolution(knowledge\_base, query): print("Query is resolved: John is Mortal")

else:

print("Query could not be resolved")

