NAME:SNEHAN.S

REG NO:241801272

EXP NAME:UNIFICATION AND RESOLUTION

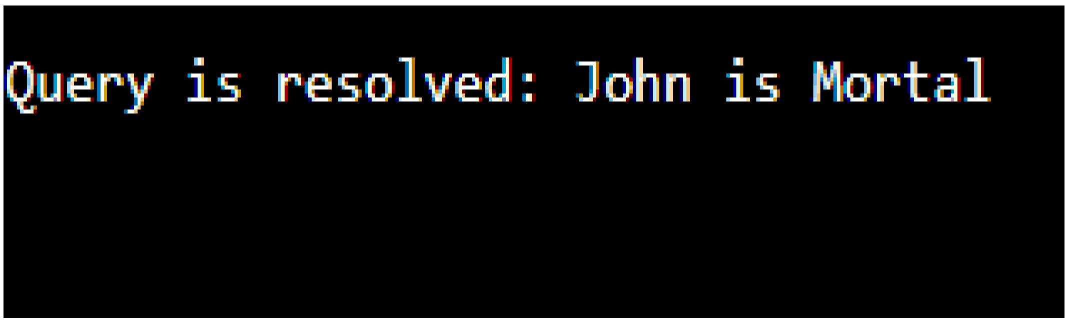
EXP NO:6

PROGRAM:

```
def unify(x, y, theta=None):
    if theta is None:
        theta = {}
    if 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):
        theta = unify(x[0], y[0], theta)
        return unify(x[1:], y[1:], theta)
    else:
        return None


def unify_var(var, x, theta):
    if var in theta:
        return unify(theta[var], x, theta)
    elif isinstance(x, str) and x in theta:
        return unify(var, theta[x], theta)
    else:
        theta[var] = x
        return theta
def negate(literal):
    if literal[0].startswith("¬"):
        return [literal[0][1:], *literal[1:]]
    else:
        return [f"¬{literal[0]}", *literal[1:]]


def substitute(literal, theta):
    return [theta.get(x, x) for x in literal]


def resolve(ci, cj):
    resolvents = []
    for di in ci:
        for dj in cj:
            theta = unify(di, negate(dj))
            if theta is not None:
                new_ci = [substitute(d, theta) for d in ci if d != di]
                new_cj = [substitute(d, theta) for d in cj if d != dj]
  seen = [] for literal in new_ci + new_cj:
          if literal not in seen: seen.append(literal) resolvent = seen
resolvents.append(resolvent) return resolvents

def resolution(kb, query):

    clauses = [clause[:] for clause in kb]

    clauses.append([negate(query)])  # Add the negated query as a clause

    new = set()



    while True:

        pairs = [(clauses[i], clauses[j]) for i in range(len(clauses)) for j in range(i + 1,
len(clauses))]

        for (ci, cj) in pairs:

            resolvents = resolve(ci, cj)

            for r in resolvents:

                if not r:  # Empty clause means query is resolved

                    return True

                r_tuple = tuple(tuple(x) for x in r)

                if r_tuple not in new:

                    new.add(r_tuple)

                    clauses.append(r)

        if all(tuple(tuple(x) for x in clause) in new for clause in clauses):

            return False


# --- Example knowledge base and query ---


# KB: Human(John) → Mortal(John)  is represented as ¬Human(John) ∨ Mortal(John)
```

```
knowledge_base = [
    [["¬Human", "John"], ["Mortal", "John"]],  # Rule: Human(John) → Mortal(John)
    [["Human", "John"]]                  # Fact: Human(John)
]

# Query: Is John Mortal?
query = ["Mortal", "John"]

# Run 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
```