LAB 2

Aim : Study of rules and unification.

❖ Program 1 :

```
domains
        disease,indication,name=symbol
predicates
        hypothesis(name,disease)
        symptom(name,indication)
clauses
        symptom(parva,fever).
        symptom(parva,rash).
        symptom(parva,headache).
        symptom(parva,runny nose).
        symptom(vidhi,chills).
        symptom(vidhi,fever).
        symptom(vidhi,headache).
        symptom(vivan,runny nose).
        symptom(vivan,rash).
        symptom(vivan,flu).
        hypothesis(Patient,measles):-
                symptom(Patient,fever),
                symptom(Patient,cough),
                symptom(Patient,conjunctivitis),
                symptom(Patient,rash).
        hypothesis(Patient,german_measles):-
                symptom(Patient,fever),
                symptom(Patient,headache),
                symptom(Patient,runny_nose),
                symptom(Patient,rash).
        hypothesis(Patient,flu):-
                symptom(Patient,fever),
                symptom(Patient,headache),
                symptom(Patient,body_ache),
                symptom(Patient,chills).
        hypothesis(Patient,common_cold):-
```

```
        symptom(Patient,headache),
        symptom(Patient,sneezing),
        symptom(Patient,sore_throat),
        symptom(Patient,chills),
        symptom(Patient,runny_nose).
hypothesis(Patient,mumps):-
        symptom(Patient,fever),
        symptom(Patient,swollen_glands).
hypothesis(Patient,chicken_pox):-
        symptom(Patient,fever),
        symptom(Patient,rash),
        symptom(Patient,body_ache),
        symptom(Patient,chills).
```

Question : **Identify patient with any particular disease based on rules and facts given above.**

Goal : hypothesis(Patient,Disease)

Output : Patient=Parva,Disease=german_measels

1 Solution

❖ Program 2 :

```
domains
        name=symbol
predicates
        male(name),
        female(name),
        parent(name,name),
        father(name,name),
        mother(name,name),
        grandmother(name,name),
        grandfather(name,name),
        brother(name,name),
        uncle(name,name),
        aunt(name,name)
        nephew(name,name),
        niece(name,name),
clauses
```

```
female(madri).
female(kunti).
female(draupadi).
female(subhadra).
female(ulupi).
female(citrangada).
male(babhruvahana).
male(irvan).
male(abhimanyu).
male(srutakirti).
male(srutakarma).
male(prativindhya).
male(sutasoma).
male(satatika).
male(yudhisthira).
male(sahadeva).
male(arjuna).
male(pandu).
male(nakula).
male(bhima).
parent(madri,nakula).
parent(madri,sahadeva).
parent(kunti,sahadeva).
parent(kunti,yudhistira).
parent(kunti,arjuna).
parent(kunti,bhima).
parent(pandu,yudhistira).
parent(pandu,bhima).
parent(pandu,arjuna).
parent(pandu,nakula).
parent(pandu,sahadeva).
parent(draupadi,satatika).
parent(draupadi,sutasoma).
parent(draupadi,prativindhya).
parent(draupadi,srutakarma).
parent(draupadi,srutakirti).
parent(nakula,satkriti).
parent(bhima,sutasoma).
parent(yudhistira,prativindhya).
parent(sahadeva,srutakarma).
```

```
parent(arjuna,srutakirti).
parent(arjuna,abhimanyu).
parent(arjuna,iravan).
parent(arjuna,babhruvahana).
parent(subhadra,abhimanyu).
parent(ulupi,irvan).
parent(citranga,babhruvahana).
mother(X,Y):-
        parent(X,Y),
        female(X).
father(X,Y):-
        parent(X,Y),
        male(X).
brother(X,Y):-
        male(X),
        parent(Z,Y),
        parent(Z,X),
        X<>Y.
uncle(X,Y):-
        male(X),
        parent(Z,Y),
        brother(X,Z).
aunt(X,Y):-
        female(X),
        mother(Z,Y),X<>Z,grandmother(W,Y),
        X<>W.
nephew(X,Y):-
        male(X),
        uncle(Y,X).
niece(X,Y):-
        female(X),
        uncle(Y,X).
grandfather(X,Y):-
        parent(Z,Y),
        father(X,Z).
        grandmother(X,y):-
        parent(Z,Y),
        mother(X,Z).
```

1. **Goal : niece(bhima,abhimanyu)**
   Output : No

2. **Goal : mother(kunti,bhima)**
   Output : Yes

3. **Goal : grandmother(kunti,abhimanyu).**
   Output : yes.

4. **Goal : nephew(arjuna,bhima).**
   Output : No

❖ Proram 3 :

```
domains
        subject=symbol
predicates
        takes(symbol,symbol),
        available(subject),
        easy(symbol),
        credits(subject,integer),
        component(subject).
clauses
        easy(hardware).
        easy(graphics).
        takes(marry,compilers).
        takes(X,Y):-easy(Y),available(Y).
        takes(X,Y):-credits(Y,8),component(graphics).
        available(hardware).
        available(db)
```

1. **Does marry take graphics course?**
   Goal : take(marry,graphics)
   Output : Yes

2. **Which course marry takes?**
   Goal : takes(marry,Course)
   Output : Course=compilers
   Course=hardware
   Course=graphics
   3 Solution

3. **Who takes graphics course?**
   Goal : takes(X,graphics)
   Output : No Solution