No dogs are intelligent

$\forall x\ (dog(x) \rightarrow \neg Intelligent(x))$ or
$\neg \exists x\ (dog(x) \wedge Intelligent(x))$

Every number is either negative or has a square root

$\forall x\ \neg(negative(x) \leftrightarrow sqroot(x))$ or

$\neg \exists x\ (negative(x) \leftrightarrow sqroot(x))$ or

$\forall x((negative(x) \wedge \neg sqroot(x)) \vee (\neg negative(x) \wedge sqroot(x))$

There is a barber in town who shaves all men in town who do not shave themselves.

$\exists x\ (Barber(x) \wedge InTown(x) \wedge$
$\forall y\ (Man(y) \wedge InTown(y) \wedge \neg Shave(y,y) \Rightarrow Shave(x,y)))$

Unification

1. Unification means making expression looks identical.
2. Can be done with the process of substitution.

Example :
p(x,F(y)) ———— 1
p(a,F(g(z)) ———- 2

1 and 2 are identical if x is replaced by a and y with g(z)

p(a,F(y)) [a/x, g(z)/y]

Unification condition:

1. Predicate symbol must be same.(p)
2. No. Of arguments in both expression must be identical.
3. If two similar variables present in same expression , then unification **fails**.

Algorithm :
Unify (A1,A2)

1. If a1 or a2 is variable/constant
a. if a1 and a2 are identical
return nil
b. else if a1 occurs in a2

return fail
else return a2/a1
c. Check for a2 in a1
fails if a2 occurs in ai
else return a1/a2

2. If predicate not same
3. If different arguments
4. Else subs to NIL
5. LOOP
6. Return subs

Example 1:

Q(a,g(x,a),f(y))
Q(a,g(f(b),a),x)

Substitute x with f(b) —-> [f(b)/x]

Q(a,g(f(b),a),f(y))
Q(a,g(f(b),a),f(b)

Substitute [b/y]

Q(a,g(f(b), f(b))
Q(a,g(f(b),f(b))

Unified Successfully !

Example 2 :

Prime (11)
Prime(y)

Substitute y with 11 [y/11]

Example 3 :

p (X, X) and p (Z, f(Z))
Substitute [x/z]
p(z,z)
p(z,f(z))
Unification failed.

Example 4 :

knows(Richard, x)
knows(Richard, John)
Substitue [john/x]

Example 5 :

p(f(a), g(Y))
p(X, X)
Substitue [f(a)/x]