A. All pets have an owner

Ax inst(x, pet) => Ey own(y,x)
inst(?x, pet) => own(sk0(?x), ?x)
~inst(?x, pet) V own(sk0(?x), ?x)

================================
~inst(?x, pet) V own(sk0(?x), ?x)
================================

B. Cats and dogs are pets

Ax inst(x, cat) V inst(x, dog) => inst(x, pet)
inst(?x1, cat) V inst(?x1, dog) => inst(?x1, pet)
~[inst(?x1, cat) V inst(?x1, dog)] V inst(?x1, pet)
~inst(?x1, cat) & ~inst(?x1, dog) V inst(?x1, pet)

================================
~inst(?x1, cat) V inst(?x1, pet)
================================

================================
~inst(?x2, dog) V inst(?x2, pet)
================================

C. Pets who have the same owner like each other

Axyz inst(x, pet) & inst(y, pet) & own(z, x) & own(z, y) => likes(x,y) & likes(y,x)
inst(?x3, pet) & inst(?y1,pet) & own(?z1, ?x3) & own(?z1, ?y1) => likes(?x3,?y1) &
likes(?y1, ?x3)
~[inst(?x3, pet) & inst(?y1,pet) & own(?z1, ?x3) & own(?z1, ?y1)]V likes(?x3,?y1) &
likes(?y1, ?x3)
~inst(?x3, pet) V ~inst(?y1,pet) V ~own(?z1, ?x3) V ~own(?z1, ?y1) V likes(?x3,?y1) &
likes(?y1, ?x3)

===
==================
~inst(?x3, pet) V ~inst(?y1,pet) V ~own(?z1, ?x3) V ~own(?z1, ?y1) V likes(?x3, ?y1)
===
================

===
==================
~inst(?x4, pet) V ~inst(?y2,pet) V ~own(?z2, ?x4) V ~own(?z2, ?y2) V likes(?y2, ?x4)
===

==================

D. If someone doesn't like pets, then they don't own any

Ax ~likes(x, pet) => ~own(x, pet)
~likes(?x5, pet) => ~own(?x5, pet)
likes(?x, pet) V ~own(?x, pet)

===============================
likes(?x5, pet) V ~own(?x5, pet)
===============================

E. Pat owns a dog and a cat

own(Pat, dog) & own(Pat, cat)

=============
own(Pat, dog)
=============

=============
own(Pat, cat)
=============

F. Chris doesn't like pets

Ax inst(x, pet) => ~likes(Chris, x)
inst(?x6, pet) => ~likes(Chris, ?x6)
~inst(?x6, pet) V ~likes(Chris, ?x6)

===================================
~inst(?x6, pet) V ~likes(Chris, ?x6)
===================================

F. Fido is a dog

================
inst(Fido, dog)
================

G. Fluffy is a cat

```
==================
inst(Fluffy, cat)
==================
```

Given:

1. ~inst(?x, pet) V own(sk0(?x), ?x)

2. ~inst(?x1, cat) V inst(?x1, pet)

3. ~inst(?x2, dog) V inst(?x2, pet)

4. ~inst(?x3, pet) V ~inst(?y1,pet) V ~own(?z1, ?x3) V ~own(?z1, ?y1) V likes(?x3, ?y1)

5. ~inst(?x4, pet) V ~inst(?y2,pet) V ~own(?z2, ?x4) V ~own(?z2, ?y2) V likes(?y2, ?x4)

6. likes(?x5, pet) V ~own(?x5, pet)

7. own(Pat, dog)

8. own(Pat, cat)

9. ~inst(?x6, pet) V ~likes(Chris, ?x6)

10. inst(Fido, dog)

11. inst(Fluffy, cat)

Prove: Chris doesn't own Fido
~own(Chris, Fido)

12. own(Chris, Fido)	Neg of thm
13. inst(Fido, pet)	10, 3; ?x2=Fido
14. ~inst(?x, pet) V own(sk0(?x), ?x)	12, 13, 1; ?x=Fido; sk0(Fido)=Chris
(6) likes(?x5, pet) V ~own(?x5, pet)	
15. inst(Fido, pet) V likes(?x5, pet)	14, 6; ?x5=sk0(Fido); ?x=Fido

(9) ~inst(?x6, pet) V ~likes(Chris, ?x6)

16. Box   9, 13, 15; ?x6=Fido (By 13, Fido is pet so can
sub pet with Fido in likes(?x5, pet)

Prove: Fluffy has an owner
Ex own(x, Fluffy)
own(sk7, Fluffy)

12. ~own(sk7, Fluffy)   Neg of thm

13. ~inst(Fluffy, pet)   12, 1; ?x=Fluffy, sk7=sk0(Fluffy)

14. ~inst(Fluffy, cat)   13, 2; ?x1=Fluffy

15. Box   14, 11