1

1	∀x(Larger(x,a) → Tet(x))	
2	rm(a) = b	
3	Large(b) ∧ Small(a)	
⋮		
	Tet(rm(a)) ∧ rm(a)≠a	

STRATEGY
for a derivation using
=Elim and AnaCon

2

We plan to build the goal conjunction by ∧Intro. So we aim for the 2 conjuncts.

1	∀x(Larger(x,a) → Tet(x))	
2	rm(a) = b	
3	Large(b) ∧ Small(a)	
⋮		
	Tet(rm(a))	
	rm(a) ≠ a	
	Tet(rm(a)) ∧ rm(a)≠a	∧ Intro

3

To help get ‘Tet(rm(a))’, we notice ‘Tet’ in line 1. So we apply ∀Elim. We aim the antecedent of that conditional so that we will be able to apply →Elim

1	∀x(Larger(x,a) → Tet(x))	
2	rm(a) = b	
3	Large(b) ∧ Small(a)	
⋮		
	Larger(rm(a),a)	
	Larger(rm(a),a) → Tet(rm(a))	∀Elim
	Tet(rm(a))	→Elim
	rm(a) ≠ a	
	Tet(rm(a)) ∧ rm(a)≠a	∧ Intro

4

Because we have ‘rm(a) = b’ on line 2, we can substitute either term for the other. We could do that to reach this new goal if we could use the information in line 3 to get ‘Larger(b,a)’.

1	∀x(Larger(x,a) → Tet(x))	
2	rm(a) = b	
3	Large(b) ∧ Small(a)	
⋮		
	Larger(b,a)	
	Larger(rm(a),a)	
	Larger(rm(a),a) → Tet(rm(a))	∀Elim
	Tet(rm(a))	→Elim
	rm(a) ≠ a	
	Tet(rm(a)) ∧ rm(a)≠a	∧ Intro

5

‘Larger(b,a)’ follows from the meanings of ‘Large’, ‘Small’, and ‘Larger’. Since we are applying AnaCon only to literals (atomic sentences and their negations), we must break apart line 3.

We also notice that we now have all we need to justify line ‘rm(a)≠a’, because nothing is larger than itself.

1	∀x(Larger(x,a) → Tet(x))	
2	rm(a) = b	
3	Large(b) ∧ Small(a)	
4	Large(b)	∧ Elim: 3
5	Small(a)	∧ Elim: 3
6	Larger(b,a)	AnaCon: 4,5
7	Larger(rm(a),a)	=Elim: 2,6
8	Larger(rm(a),a) → Tet(rm(a))	∀Elim: 1
9	Tet(rm(a))	→Elim: 7,8
10	rm(a) ≠ a	AnaCon:7
11	Tet(rm(a)) ∧ rm(a)≠a	∧ Intro: 9,10

If we did not have AnaCon, we would need two additional premises. What general premises would express the relevant meaning relations?

(1) Our goal is a negation, so we assume the sentence it negates, and plan to use ¬Intro. To do this, we need ⊥ in the subproof.

```
1 | ∃xB(x)                 P
2 | ¬∃x(B(x) ↔ C(x))       P
  |-----------------
3 | | ∀xC(x)
  | |-------
: | :
  | | ⊥
  | ¬∀xC(x)                ¬Intro
```

STRATEGY for:
{∃xB(x), ¬∃x(B(x) ↔ C(x))}
/∴ ¬∀xC(x)

(2) We need to apply ∃Elim to use the first premise. So we start a subproof with a boxed constant that does not occur outside the subproof, using that constant in an instance of line 1. Our goal in that subproof, too, will be ⊥, so we can move it to the first subproof by ∃Elim.

```
1 | ∃xB(x)                 P
2 | ¬∃x(B(x) ↔ C(x))       P
  |-----------------
3 | | ∀xC(x)
  | |-------
4 | | | [a] B(a)
  | | |---------
: | : :
  | | | ⊥
  | | ⊥
  | ¬∀xC(x)                ¬Intro
```

(3) ⇒

```
1 | ∃xB(x)                        P
2 | ¬∃x(B(x) ↔ C(x))              P
  |-----------------
3 | | ∀xC(x)
  | |-------
4 | | | [a] B(a)
  | | |---------
: | : :
  | | | ∃x(B(x) ↔ C(x))
  | | | ⊥                         ⊥Intro
  | | ⊥                           ∃Elim
  | ¬∀xC(x)                       ¬Intro
```

(3) We need a sentence and its negation to get ⊥. We already have a negation on line 2, so aim for the sentence that it negates.

(4)

```
1 | ∃xB(x)                        P
2 | ¬∃x(B(x) ↔ C(x))              P
  |-----------------
3 | | ∀xC(x)
  | |-------
4 | | | [a] B(a)
  | | |---------
: | : :
  | | | B(a) ↔ C(a)
  | | | ∃x(B(x) ↔ C(x))           ∃Intro
  | | | ⊥                         ⊥Intro
  | | ⊥                           ∃Elim
  | ¬∀xC(x)                       ¬Intro
```

⇐ (4) To get '∃x(B(x) ↔ C(x))' inside this subproof, aim for an instance. 'B(a) ↔ C(a)' should be the easiest one to get because we already have 'B(a)'.

(5) ⇒

```
1 | ∃xB(x)                          P
2 | ¬∃x(B(x) ↔ C(x))                P
  |-----------------
3 | | ∀xC(x)
  | |-------
4 | | | [a] B(a)
  | | |---------
5 | | | | B(a)
  | | | |-----
: | : : :
  | | | | C(a)
  | | |
  | | | | C(a)
  | | | |-----
: | : : :
  | | | | B(a)
  | | | B(a) ↔ C(a)                 ↔Intro
  | | | ∃x(B(x) ↔ C(x))             ∃Intro
  | | | ⊥                           ⊥Intro
  | | ⊥                             ∃Elim
  | ¬∀xC(x)                         ¬Intro
```

(5) Start 2 subproofs to get 'B(a) ↔ C(a)' by ↔Intro.

(6)

```
1  | ∃xB(x)                          P
2  | ¬∃x(B(x) ↔ C(x))                P
   |-----------------
3  | | ∀xC(x)
   | |-------
4  | | | [a] B(a)
   | | |---------
5  | | | | B(a)
   | | | |-----
6  | | | | C(a)                      ∀Elim: 3
   | | |
7  | | | | C(a)
   | | | |-----
8  | | | | B(a)                      Reit: 4
9  | | | B(a) ↔ C(a)                 ↔Intro: 5-6,7-8
10 | | | ∃x(B(x) ↔ C(x))             ∃Intro: 9
11 | | | ⊥                           ⊥Intro: 9,2
12 | | ⊥                             ∃Elim:1,4-10
13 | ¬∀xC(x)                         ¬Intro: 3-12
```

⇐ (6) We already have all we need to justify all the steps. So complete the proof by filling in justifications.

STRATEGY: {∀x(C(x) ↔ D(x)), C(a), ∃xD(x) → ∀xM(x), ∀x(M(x) → L(x))} /∴ ∀xL(x)

①

1	∀x(C(x) ↔ D(x))	P
2	C(a)	P
3	∃xD(x) → ∀xM(x)	P
4	∀x(M(x) → L(x))	P
	⋮	
	∀xL(x)	

② We can't reach our goal by easy rules like ∧, ∃, or ∀Elim. Next best strategy: build it by ∀Intro. Start a subproof with a boxed constant that's not outside the subproof; get an instance of our goal with the constant in the box. So aim for 'L(b)' in a subproof starting with [b].

1	∀x(C(x) ↔ D(x))	P
2	C(a)	P
3	∃xD(x) → ∀xM(x)	P
4	∀x(M(x) → L(x))	P
	[b]	
	⋮	
	L(b)	
	∀xL(x)	∀Intro

③ The only premise with 'L' is 4, so use the instance of 4 with 'b'.

1	∀x(C(x) ↔ D(x))	P
2	C(a)	P
3	∃xD(x) → ∀xM(x)	P
4	∀x(M(x) → L(x))	P
5	[b]	
	⋮	
	M(b) → L(b)	∀Elim:4
	L(b)	
	∀xL(x)	∀Intro

④ To use this to reach 'L(b)', we also need 'M(b)', so aim for that.

1	∀x(C(x) ↔ D(x))	P
2	C(a)	P
3	∃xD(x) → ∀xM(x)	P
4	∀x(M(x) → L(x))	P
5	[b]	
	⋮	
	M(b)	
	M(b) → L(b)	∀Elim: 4
	L(b)	→Elim
	∀xL(x)	∀Intro

⑤ 'M' is in only two premises. In 4, it's in the antecedent of the conditional to which '∀x' applies, so that's no help. But in 3 it's in the consequent. Also, that consequent is universal, so it would yield 'M(b)' by ∀Elim. So use 3 to get '∀xM(x)'.

1	∀x(C(x) ↔ D(x))	P
2	C(a)	P
3	∃xD(x) → ∀xM(x)	P
4	∀x(M(x) → L(x))	P
5	[b]	
	⋮	
	∀xM(x)	→Elim
	M(b)	∀Elim
	M(b) → L(b)	∀Elim:4
	L(b)	→Elim
	∀xL(x)	∀Intro

⑥ To get '∀xM(x)' from 3, first aim for '∃xD(x)'.

1	∀x(C(x) ↔ D(x))	P
2	C(a)	P
3	∃xD(x) → ∀xM(x)	P
4	∀x(M(x) → L(x))	P
5	[b]	
	⋮	
	∃xD(x)	
	∀xM(x)	→Elim
	M(b)	∀Elim
	M(b) → L(b)	∀Elim:4
	L(b)	→Elim
	∀xL(x)	∀Intro

⑦ Our goal is an existential. Since we can't pull it out of anything, build it by ∃Intro. So we need an instance. Except for 3, the only premise with 'D' is line 1, so we'll use an instance of that to get an instance of '∃xD(x)'. We need a constant we can get after 'C'. Line 2 is 'C(a)', so we use 'a'.

1	∀x(C(x) ↔ D(x))	P
2	C(a)	P
3	∃xD(x) → ∀xM(x)	P
4	∀x(M(x) → L(x))	P
5	[b]	
	⋮	
	D(a)	
	∃xD(x)	
	∀xM(x)	→Elim
	M(b)	∀Elim
	M(b) → L(b)	∀Elim:4
	L(b)	→Elim
	∀xL(x)	∀Intro

⑧ Now all we need to do is fill in the justifi-cations.

1	∀x(C(x) ↔ D(x))	P
2	C(a)	P
3	∃xD(x) → ∀xM(x)	P
4	∀x(M(x) → L(x))	P
5	[b]	
6	C(a) ↔ D(a)	∀Elim: 1
7	D(a)	↔Elim: 2,5
8	∃xD(x)	∃Intro: 7
9	∀xM(x)	→Elim: 3,8
10	M(b)	∀Elim: 9
11	M(b) → L(b)	∀Elim: 4
12	L(b)	→Elim:10,11
13	∀xL(x)	∀Intro: 5-12

STRATEGY for
{∃x(J(x) ∧ K(x)), ¬∀x(K(x) → J(x))}
/∴ ∃xJ(x) ∧ ∃x¬J(x)

(1) Our conclusion is a conjunction. So we try to get each conjunct alone, then join them by ∧Intro.

```
1 | ∃x(J(x) ∧ Kx)        P
2 | ¬∀x(K(x) → J(x))     P
:
: ∃xJ(x)
:
: ∃x¬J(x)
  | ∃xJ(x) ∧ ∃x¬J(x)
```

(2) ⇓ We should be able to use the first premise to get '∃xJ(x)'. We begin a subproof, planning to apply the ∃Elim rule.

```
1 | ∃x(J(x) ∧ Kx)          P
2 | ¬∀x(K(x) → J(x))       P
3 | | [a] J(a) ∧ K(a)
:  :
: ∃xJ(x)
:
: ∃x¬J(x)
| ∃xJ(x) ∧ ∃x¬J(x)
```

(3) ⇒ Within the subproof, we need something without 'a' that we can pull out of the subproof by the ∃Elim rule. We want '∃xJ(x)', so we'll try to get that. We need an instance of '∃xJ(x)', which we can get from 3 by ∧Elim.

```
1 | ∃x(J(x) ∧ Kx)             P
2 | ¬∀x(K(x) → J(x))          P
3 | | [a] J(a) ∧ K(a)
4 | | Ja                 ∧Elim: 3
5 | | ∃xJ(x)             ∃Intro: 4
6 | ∃xJ(x)          ∃Elim: 1,3-5
 :
  | ∃x¬J(x)
  | ∃xJ(x) ∧ ∃x¬J(x)
```

(4) To get '∃x¬J(x)', we must use line 2. The only way to use it is in a contradictory pair. So assume negation of desired statement, and aim for a contradiction with 2.

```
1 | ∃x(J(x) ∧ Kx)             P
2 | ¬∀x(K(x) → J(x))          P
3 | | [a] J(a) ∧ K(a)
4 | | Ja                 ∧Elim: 3
5 | | ∃xJ(x)             ∃Intro: 4
6 | ∃xJ(x)          ∃Elim: 1,3-5

7 | | ¬∃x¬J(x)
 :  :
  | | ∀x(K(x) → J(x))
  | | ⊥                     ⊥Intro
  | ∃x¬J(x)                 ¬Intro
  | ∃xJ(x) ∧ ∃x¬J(x)        ∧Intro
```

(5) ⇓ Plan to use "Intro. Start a subderivation with a new boxed constant constant, and aim for the desired conditional with that constant. Set up subderivation for → Intro.

```
1 | ∃x(J(x) ∧ Kx)               P
2 | ¬∀x(K(x) → J(x))            P
3 | | [a] J(a) ∧ K(a)
4 | | J(a)                ∧Elim: 3
5 | | ∃xJ(x)              ∃Intro: 4
6 | ∃xJ(x)           ∃Elim: 1,3-5
7 | | ¬∃x¬J(x)
8 | | | [b]
 :  :  :
  | | | K(b) → J(b)
  | | ∀x(K(x) → J(x))    ∀Intro
  | | ⊥                  ⊥ Intro
  | ∃x¬J(x)              ¬Intro
  | ∃xJ(x) ∧ ∃x¬J(x)     ∧Intro
```

(6) ⇓

```
1  | ∃x(J(x) ∧ Kx)                      P
2  | ¬∀x(K(x) → J(x))                   P
3  | | [a] J(a) ∧ K(a)
4  | | J(a)                       ∧Elim: 3
5  | | ∃xJ(x)                   ∃ Intro: 4
6  | ∃xJ(x)                  ∃Elim: 1,3-5
7  | | ¬∃x¬J(x)
8  | | | [b]
9  | | | | K(b)
10 | | | | | ¬J(b)
11 | | | | | ∃x¬J(b)            ∃ Intro: 10
12 | | | | | ⊥                 ⊥ Intro: 1017
13 | | | | J(b)               ¬intro: 10-12
14 | | | K(b) → J(b)          →Intro: 9-13
15 | | ∀x(K(x) → J(x))        ∀Intro: 8-14
16 | | ⊥                      ⊥ Intro: 2,15
17 | ∃x¬J(x)                  ¬Intro: 7-16
18 | ∃xJ(x) ∧ ∃x¬J(x)         ∧Intro: 6,17
```

⇐(6) We have no way to get 'J(b)' by use of easy rules. Also, the only way we can use line 7 is in a contradiction. So we assume '¬J(b)'. That justifies '∃x¬J(x)', contradicting line 7.

STRATEGY for:
{∀x[(F(x) v G(x)) → H(x)], ∀x(F(x) ↔ ¬G(x))} /∴ ∀xH(x)

(1)

1	∀x[(F(x) v G(x)) → H(x)]	P
2	∀x(F(x) ↔ ¬ G(x))	P
⋮		
	∀xH(x)	

(2)

1	∀x[(F(x) v G(x)) → H(x)]	P
2	∀x(F(x) ↔ ¬ G(x))	P
3	⎮ [a]	
⋮	⎮ ⋮	
	⎮ H(a)	
	∀xH(x)	∀Intro

(2) The goal is a universal. For ∀Intro, start a subproof with a boxed constant that does not occur outside the subproof. There are no constants outside the subproof, so we can use any constant. In the subproof, aim for an instance of the universal with the constant introduced in the box.

(3) To get our instance, we'll be plugging the constant from our new goal into our universal premises. Since line 1 has the predicate 'H' that we need, begin there.

1	∀x[(F(x) v G(x)) → H(x)]	P
2	∀x(F(x) ↔ ¬ G(x))	P
3	⎮ [a]	
4	⎮ (F(a) v G(a)) → H(a)	
⋮	⎮ ⋮	
	⎮ H(a)	
	∀xH(x)	∀Intro

(4)

1	∀x[(F(x) v G(x)) → H(x)]	P
2	∀x(F(x) ↔ ¬ G(x))	P
3	⎮ [a]	
4	⎮ (F(a) v G(a)) → H(a)	∀Elim:1
⋮	⎮ ⋮	
	⎮ (F(a) v G(a))	
	⎮ H(a)	→Elim
	∀xH(x)	∀ Intro

(4) We can get 'H(a)' with the help of line 3 if we can get 'F(a) v G(a)'.

(5) ⇒ ⇒ ⇒ Neither disjunct is obviously the easy one to get, so we assume the negation of the disjunction.

1	∀x[(F(x) v G(x)) → H(x)]	P
2	∀x(F(x) ↔ ¬ G(x))	P
3	⎮ [a]	
4	⎮ (F(a) v G(a)) → H(a)	∀Elim
5	⎮ ⎮ ¬F(a) v G(a))	
⋮	⎮ ⎮ ⋮	
	⎮ (F(a) v G(a))	
	⎮ H(a)	→Elim
	∀xH(x)	∀ Intro

(6) ⇓ Apply common strategy for getting a contradiction under the negation of a disjunction.. Assume 'G(a)' because we have an easy way to use '¬G(a)' (with line 2), but we have no easy way to use '¬F(a)'. We will use '¬G(a)' to get 'F(a)', then 'F(a) v G(a)'.

1	∀x[(F(x) v G(x)) → H(x)]	P
2	∀x(F(x) ↔ ¬G(x))	P
3	⎮ [a]	
4	⎮ (F(a) v G(a)) → H(a)	∀Elim:1
5	⎮ ⎮ ¬(F(a) v G(a))	
6	⎮ ⎮ ⎮ G(a)	
7	⎮ ⎮ ⎮ F(a) v G(a)	vIntro: 6
8	⎮ ⎮ ⎮ ⊥	⊥Intro: 5,7
9	⎮ ⎮ ¬G(a)	¬Intro: 5-8
⋮	⎮ ⋮	
	⎮ ⎮ F(a)	
	⎮ ⎮ F(a) v G(a)	vIntro
	⎮ ⎮ ⊥	⊥Intro
	⎮ ¬ ¬ (F(a) v G(a))	¬Intro
	⎮ F(a) v G(a)	¬¬Elim
	⎮ H(a)	→Elim
	∀xH(x)	∀ Intro

(7) ⇒ ⇒ ⇒ ⇒ To get 'F(a)', use line 2. When we get 'F(a) v G(a)' again, we can use ⊥ Intro and ¬ Intro to get '¬¬(F(a) v G(a))', then drop the double negation to get 'F(a) v G(a)' where we really want it, outside of any subproof.

1	∀x[(F(x) v G(x)) → H(x)]	P
2	∀x(F(x) ↔ ¬G(x))	P
3	⎮ [a]	
4	⎮ (F(a) v G(a)) → H(a)	∀Elim:1
5	⎮ ⎮ ¬(F(a) v G(a))	
6	⎮ ⎮ ⎮ G(a)	
7	⎮ ⎮ ⎮ F(a) v G(a)	vIntro: 6
8	⎮ ⎮ ⎮ ⊥	⊥Intro: 5,7
9	⎮ ⎮ ¬G(a)	¬Intro: 5-8
10	⎮ ⎮ F(a) ↔ ¬G(a)	∀Elim: 2
11	⎮ ⎮ F(a)	↔Elim: 9,10
12	⎮ ⎮ F(a) v G(a)	vIntro: 11
13	⎮ ⎮ ⊥	⊥Intro: 12,5
14	⎮ ¬ ¬ (F(a) v G(a))	¬Intro: 5-13
15	⎮ F(a) v G(a)	¬¬Elim: 14
16	⎮ H(a)	→Elim: 4,15
17	∀xH(x)	∀Intro: 3-16

STRATEGY for:
{∃xR(x,x) → ∀x(Q(x) ∨ S(x)), ∀x(R(a,x) ∧ ¬Q(x))} /∴ ∀xS(x)

(1)

```
1 | ∃xR(x,x) → ∀x(Q(x) ∨ S(x))
2 | ∀x(R(a,x) ∧ ¬Q(x))
  |______________________
  ⋮
  | ∀xS(x)
```

(2) The only ‘S’ in the premises is in 1. So aim for the antecedent. Get the consequent by →Elim, then try to get ‘∀xS(x)’ from that. ⇒ ⇒ ⇒

```
1 | ∃xR(x,x) → ∀x(Q(x) ∨ S(x))
2 | ∀x(R(a,x) ∧ ¬Q(x))
  |______________________
  ⋮
  | ∃xR(x,x)
  | ∀x(Q(x) ∨ S(x))
  ⋮
  | ∀xS(x)
```

⇐ **(3)** To get ‘∃xR(x,x)’, we’ll get an instance, then apply ∃ Intro. Use line 2 to get the instance ‘R(a,a)’.

(3)

```
1 | ∃xR(x,x) → ∀x(Q(x) ∨ S(x))
2 | ∀x(R(a,x) ∧ ¬Q(x))
  |______________________
3 | R(a,a) ∧ ¬Q(a)            ∀Elim: 2
4 | R(a,a)                    ∧Elim: 3
5 | ∃xR(x,x)                  ∃ Intro: 4
6 | ∀x(Q(x) ∨ S(x))           →Elim: 1,5
  ⋮
  | ∀xS(x)
```

(4) ‘∀xS(x)’ is not in any statement we have yet. Plan to build it by "Intro. Begin a subproof with a boxed constant that is not in any assumption, and aim for an instance of ‘∀xS(x)’ with that constant. Since ‘a’ is in a premise, we must use a different constant.

(4) ⇒

```
1 | ∃xR(x,x) → ∀x(Q(x) ∨ S(x))   P
2 | ∀x(R(a,x) ∧ ¬Q(x))           P
  |______________________
3 | R(a,a) ∧ ¬Q(a)            ∀Elim: 2
4 | R(a,a)                    ∧Elim: 3
5 | ∃xR(x,x)                  ∃ Intro: 4
6 | ∀x(Q(x) ∨ S(x))           →Elim: 1,5
7 | | [b]
  ⋮ ⋮
  | | S(b)
  | ∀xS(x)
```

(5) To get ‘S(b)’, begin by applying ∀Elim to 6:

```
1 | ∃xR(x,x) → ∀x(Q(x) ∨ S(x))   P
2 | ∀x(R(a,x) ∧ ¬Q(x))           P
  |______________________
3 | R(a,a) ∧ ¬Q(a)            ∀Elim: 2
4 | R(a,a)                    ∧Elim: 3
5 | ∃xR(x,x)                  ∃ Intro: 4
6 | ∀x(Q(x) ∨ S(x))           →Elim: 1,5
7 | | [b]
8 | | Q(b) ∨ S(b)
  ⋮ ⋮
  | | S(b)
  | ∀xS(x)
```

(6) Set up for ∨ Elim:
⇓

```
1 | ∃xR(x,x) → ∀x(Q(x) ∨ S(x))   P
2 | ∀x(R(ax) ∧ ¬Q(x))            P
  |______________________
3 | R(a,a) ∧ ¬Q(a)            ∀Elim: 2
4 | R(a,a)                    ∧Elim: 3
5 | ∃xR(x,x)                  ∃ Intro: 4
6 | ∀x(Q(x) ∨ S(x))           →Elim: 1,5
7 | | [b]
8 | | Q(b) ∨ S(b)             ∀Elim: 6
9 | | | Q(b)
  ⋮ ⋮ ⋮
  | | | S(b)
  ⋮ ⋮ ⋮
  | | S(b)
  | ∀xS(x)
```

(7) ⇒ ⇒

Note that we can get ‘¬Q (b)’ by applying ∀Elim to line 2 again. This yields a contradiction, which we can put to work for us by applying ⊥Elim to get ‘S(b)’.

```
 1 | ∃xR(x,x) → ∀x(Q(x) ∨ S(x))   P
 2 | ∀x(R(a,x) ∧ ¬Q(x))           P
   |______________________
 3 | R(a,a) ∧ ¬Q(a)            ∀Elim: 2
 4 | R(a,a)                    ∧Elim: 3
 5 | ∃xR(x,x)                  ∃ Intro: 4
 6 | ∀x(Q(x) ∨ S(x))           →Elim: 1,5
 7 | | [b]
 8 | | Q(b) ∨ S(b)             ∀Elim: 6
 9 | | | Q(b)
10 | | | R(a,b) ∧ ¬Q(b)        ∀Elim: 2
11 | | | ¬Q(b)                 ∧Elim: 10
12 | | | ⊥                     ⊥ Intro: 9,11
13 | | | S(b)                  ⊥ Elim: 12
14 | | | S(b)
15 | | S(b)              ∨Elim: 8,9-13,14-14
16 | ∀xS(x)                    ∀Intro: 7-15
```

STRATEGY: {Cube(a), ¬∃x∃y(x≠y ∧ SameShape(x,y))} /∴ ∀x(Cube(x) → x = a)

①

```
1 | Cube(a)
2 | ¬∃x∃y(x≠y ∧ SameShape(x,y))
  | ---------------------------
  ⋮
  | ∀x(Cube(x) → x = a)
```

② We set up a subderivation to reach our goal by ∀Intro.

```
1 | Cube(a)
2 | ¬∃x∃y(x≠y ∧ SameShape(x,y))
  | ---------------------------
3 | | [b]
  ⋮ ⋮
  | | Cube(b) → b = a
  | ∀x(Cube(x) → x = a)
```

③ We start another subderivation to build the conditional we need.

```
1 | Cube(a)
2 | ¬∃x∃y(x≠y ∧ SameShape(x,y))
  | ---------------------------
3 | | [b]
4 | | | Cube(b)
  | | | -------
  ⋮ ⋮ ⋮
  | | | b = a
  | | Cube(b) → b = a
  | ∀x(Cube(x) → x = a)
```

④ The only way to use line 2 is in a contradiction. So we assume the negation of our current goal, and try to get a contradiction with line 2.

```
1 | Cube(a)
2 | ¬∃x∃y(x≠y ∧ SameShape(x,y))
  | ---------------------------
3 | | [b]
4 | | | Cube(b)
  | | | -------
5 | | | | b ≠ a
  | | | | -----
  ⋮ ⋮ ⋮ ⋮
  | | | | ∃x∃y(x≠y ∧ SameShape(x,y))
  | | | | ⊥
  | | | b = a
  | | Cube(b) → b = a
  | ∀x(Cube(x) → x = a)
```

⑤ We aim for an instance of the existential. We have one conjunct, so we need the other, which we can get right away by AnaCon.

```
1 | Cube(a)
2 | ¬∃x∃y(x≠y ∧ SameShape(x,y))
  | ---------------------------
3 | | [b]
4 | | | Cube(b)
  | | | -------
5 | | | | b ≠ a
  | | | | -----
6 | | | | SameShape(b,a)
7 | | | | b≠a ∧ SameShape(b,a)
  ⋮ ⋮ ⋮ ⋮
  | | | | ∃x∃y(x≠y ∧ SameShape(x,y))
  | | | | ⊥
  | | | b = a
  | | Cube(b) → b = a
  | ∀x(Cube(x) → x = a)
```

⑥ ⇓ We just need to build up the existential sentence, and complete the justifications.

```
 1 | Cube(a)                                        P
 2 | ¬∃x∃y(x≠y ∧ SameShape(x,y))                    P
   | ---------------------------
 3 | | [b]
 4 | | | Cube(b)
   | | | -------
 5 | | | | b ≠ a
   | | | | -----
 6 | | | | SameShape(b,a)                   AnaCon: 1,4
 7 | | | | b≠a ∧ SameShape(b,a)             ∧Intro: 5,6
 8 | | | | ∃y(b=y ∧ SameShape(b,y))         ∃Intro: 7
 9 | | | | ∃x∃y(x≠y ∧ SameShape(x,y))       ∃Intro: 8
10 | | | | ⊥                                ⊥Intro: 2,9
11 | | | ¬b ≠ a                             ¬Intro: 5-10
12 | | | b = a                              ¬Elim: 11
   | | |
13 | | Cube(b) → b = a                      →Intro: 4-13
14 | ∀x(Cube(x) → x = a)                    ∀Intro: 3-13
```

Without AnaCon, we'd need a third premise. You should be able to complete the modified derivation:

```
1 | Cube(a)
2 | ¬∃x∃y(x≠y ∧ SameShape(x,y))
3 | ∀x∀y((Cube(x) ∧ Cube(y)) → SameShape(x,y))
  | ------------------------------------------
  ⋮
  | ∀x(Cube(x) → x = a)
```

Strategy: {∃x∀y(F(x) → G(x,y)))} /∴. ∀xF(x) → ∃xG(x,x)

(1)

1 | ∃x∀y(F(x) → G(x,y)) P
:
:
| ∀xF(x) → ∃xG(x,x)

(2) Since conclusion is a conditional, set up a subproof to build it by →Intro.

1 | ∃x∀y(F(x) → G(x,y)) P
2 | | ∀xF(x)
: :
: :
| | ∃xG(x,x)
| ∀xF(x) → ∃xG(x,x) →Intro

(3)

Now we are aiming for an existential statement. When we have one existential statement and are aiming for another one, a good way to try to reach that goal is by ∃Elim. So set up a subproof for later use of the ∃Elim rule. Aim for the desired existential statement at the end of the subproof. Then ∃Elim will let us move it to the left, out of the subproof.

1 | ∃x∀y(F(x) → G(x,y)) P
2 | | ∀xF(x)
3 | | | [a] ∀y(F(a) → G(a,y))
: : :
: : :
| | | ∃xG(x,x)
| | ∃xG(x,x) ∃Elim
| ∀xF(x) → ∃xG(x,x) →Intro

(4)

To get ‘∃xG(x,x)’, aim for an instance, then use the ∃Intro rule. So you want ‘G’, followed by two occurrences of the same constant. Since you have a ‘G’ with an ‘a’ in line 3, try for ‘G(a,a)’.

1 | ∃x∀y(F(x) → G(x,y)) P
2 | | ∀xF(x)
3 | | | [a] ∀y(F(a) → G(a,y))
: : :
: : :
| | | G(a,a)
| | | ∃xG(x,x)
| | ∃xG(x,x) ∃Elim
| ∀xF(x) → ∃xG(x,x) →Intro

(5)

Notice that if you plug in ‘a’ in place of ‘y’ using the ∀Elim rule on line 3, you will get a conditional that has your current goal as it’s consequent. That is a promising strategy. In this case, it is particularly promising because its antecedent is easy to get by applying ∀Elim to line 2.

1 | ∃x∀y(F(x) → G(x,y)) P
2 | | ∀xF(x)
3 | | | [a] ∀y(F(a) → G(a,y))
4 | | | F(a) → G(a,a) ∀Elim: 3
5 | | | F(a) ∀Elim: 2
6 | | | G(a,a) →Elim: 4,5
7 | | | ∃xG(x,x) ∃ Intro: 6
8 | | ∃xG(x,x) ∃Elim: 1,3-7
9 | ∀xF(x) → ∃xG(x,x) →Intro: 2-8

STRATEGY: {∀x(Gxa ↔ x = b), ¬∃x(Hx ∧ ∃yG(b,y)), Hb} /∴ ¬∃xG(x,a)

①

```
1 | ∀x(Gxa ↔ x = b)
2 | ¬∃x(Hx ∧ ∃yG(b,y))
3 | H(b)
  ⋮
  | ¬∃xG(x,a)
```

②

Our goal is a negation, so we plan for ¬Intro and assume the opposite. The contradiction will be with the only negation in the premises.

```
1 | ∀x(Gxa ↔ x = b)
2 | ¬∃x(Hx ∧ ∃yG(b,y))
3 | H(b)
4 | | ∃xG(x,a)
  ⋮ ⋮
  | | ∃x(H(x) ∧ ∃yG(x,y))
  | | ⊥
  | ¬∃xG(x,a)                 ¬Intro
```

③

To build the existential that is now our goal by ∃Intro, we need an instance. If we use ‘b’, we already have one conjunct on line 3, so we just need to aim for the other conjunct, ‘∃yG(b,y)’.

```
1 | ∀x(Gxa ↔ x = b)
2 | ¬∃x(Hx ∧ ∃yG(b,y))
3 | H(b)
4 | | ∃xG(x,a)
  ⋮ ⋮
  | | ∃yG(b,y)
  | | H(b) ∧ ∃yG(b,y)          ∧Intro
  | | ∃x(H(x) ∧ ∃yG(x,y))      ∃Intro
  | | ⊥                        ⊥Intro
  | ¬∃xG(x,a)                  ¬Intro
```

④

We have an existential sentence at the top, and are aiming for another. So we start a subderivation to apply ∃Elim to the first one to get the second.

```
1 | ∀x(Gxa ↔ x = b)
2 | ¬∃x(Hx ∧ ∃yG(b,y))
3 | H(b)
4 | | ∃xG(x,a)
5 | | | [c] G(c,a)
  ⋮ ⋮ ⋮
  | | | ∃yG(b,y)
  | | ∃yG(b,y)
  | | H(b) ∧ ∃yG(b,y)
  | | ∃x(H(x) ∧ ∃yG(x,y))
  | | ⊥
  | ¬∃xG(x,a)
```

⑤

To build the existential that is now our goal by ∃Intro, we need an instance. We could get ‘G(b,a)’ as our instance if we could replace the ‘c’ in line 4 with ‘b’. We can do that with the help of line 1.

```
 1 | ∀x(Gxa ↔ x = b)
 2 | ¬∃x(Hx ∧ ∃yG(b,y))
 3 | H(b)
 4 | | ∃xG(x,a)
 5 | | | [c] G(c,a)
 6 | | | G(c,a) ↔ c = b          ∀Elim: 1
 7 | | | c = b                   ↔Elim: 5,6
 8 | | | G(b,a)                  =Elim: 5,7
 9 | | | ∃yG(b,y)                ∃Intro: 8
10 | | ∃yG(b,y)                  ∃Elim: 4,5-9
11 | | H(b) ∧ ∃yG(b,y)           ∧Intro: 3,10
12 | | ∃x(H(x) ∧ ∃yG(x,y))       ∃Intro: 11
13 | | ⊥                         ⊥Intro: 2,12
14 | ¬∃xG(x,a)                   ¬Intro: 4-13
```

EXTRA EXERCISES FOR CHAPTER 13, GROUP 2

If the argument is FO-valid, use Fitch to give a proof. Start by opening Proof CStern 130x, where 1≤x≤9, or Proof CStern 13x where x≥10. Use AnaCon only for literals (atomic sentences and their negations). Do not use TautCon.

If the argument is FO-invalid, use Tarski's World to give a counterexample. Start by opening Sentences CStern 130x for H 13.x, where 1≤x≤9, or Sentences CStern 13x where x≥10. Save worlds as World CStern 130x or 13x, corresponding to Sentences CStern 130x

H 13.9

| ∃x(Tet(x) ∧ ∀y[Cube(y) → Adjoins(y,x)]
| ---
| ∀x(Cube(x) → ∃y(Tet(y) ∧ Adjoins(x,y)))

H 13.10

| ∃xCube(x)
| ∀x(Cube(x) → ∃y(Dodec(y) ∧ Adjoins(x,y)))
| ---
| ∃x(Dodec(x) ∧ ∀y[Cube(y) → Adjoins(y,x)])

H 13.11

| ∀x∀y([Cube(x) ∧ Tet(y)] → SameSize(x,y))
| ∃x∃y (Cube(x) ∧ Tet(y))
| ∃x[Dodec(x) ∧ ∀y(Tet(y) → Smaller(x,y))]
| ---
| ∃x∃y (Dodec(x) ∧ Cube(y) ∧ Smaller(x,y))

H 13.12

| ∀x∀y([Cube(x) ∧ Tet(y)] → SameSize(x,y))
| ∃x∃y(Cube(x) ∧ Tet(y))
| ∃x[Dodec(x) ∧ ∀y(Tet(y) → Smaller(x,y))]
| ---
| ∀x∀y[(Dodec(x) ∧ Cube(y)) → Larger(y,x)]

H 13.13

| ∃x(Tet(x) ∧ ∀y[(Tet(y) ∧ y≠x) → Larger(x,y)]
| ∃x∃y(Tet(x) ∧ Tet(y) ∧ y≠x)
| ∀x∀y[(Tet(x) ∧ SameCol(x,y)) → SameSize(x,y)]
| ∀x∃y(SameCol(x,y) ∧ x≠y)
| ---
| ¬∃x∃y(Tet(x) ∧ Tet(y) ∧ x≠y ∧ SameCol(x,y))

H 13.14

| ∃x∃y(Dodec(x) ∧ Tet(y) ∧ Larger(x,y))
| ¬∃x(Tet(x) ∧ ∀y(Cube(y) → Smaller(x,y)))
| ---
| ∃x(Large(x) ∧ Cube(x)) → ∀x(Large(x) → Cube(x))

H 13.15

| ∃x(Dodec(x) ∧ ∀y(Cube(y) → Larger(x,y))
| ∃x(Cube(x) ∧ ∀y(Tet(y) → Larger(x,y)))
| ---
| ∀x((Cube(x) ∨ Tet(x)) → ∃y(Dodec(y) → Larger(y,x)))