↪ Stylistic variants:

Everything is F.

Each thing is F.

⋀x Fx

All things are F.

For each x, x is F.

Something is F.

⋁x Fx At least one thing is F.

There is a F.

Some F is G.

Some F’s are G’s.

⋁x (Fx ∧ Gx) At least one F is G.

There is a F who/which is G.

\*\*A (certain) F is G. \*\*

⋁x (Fx ∧ ~Gx) Some F isn’t G.

All F’s are G.

Each F is G.

Every F is G.

Everything that is F is G.

Anything that is F is G.

⋀x (Fx → Gx) A (generic) F is G.

\*\*Any F is G. \*\*

If anything is F, it is G.

Anyone who is F is G.

Whoever/Whatever is F is G.

The only F’s are G.

Only F’s are G’s.

⋀x (Gx → Fx)

~⋁x (~Fx ∧ Gx)

None but F’s are G’s.

Nothing other/but F’s are G’s.

⋀x(Gx ↔ Fx)

All and only F’s are G’s.

⋀x(Gx → Fx) ∧ ⋀x(Fx → Gx)

~⋁x (Fx ∧ Gx) No F is G.

⋀x (Fx → ~Gx) F’s are not G.

⋀x (Fx → Gx) ∧ ⋀x (Ix → Gx)

All F’s and I are G.

⋀x (Fx ∨ Ix → Gx)

⋀x (Ix → Fx ∧ Gx) All I’s are F’s and G’s.

⋀x (Ix → ~Fx ∨ Gx)

No I is F unless he/she/it is G.

No I’s are F’s unless they are G’s.

~⋁x (Ix ∧ Fx ∧ ~Gx)

⋀x (Ix → (~Fx ∨ GA)

⋀x (Ix → ~Fx) ∨ GA

No I is F unless A is G.

~⋁x (Ix ∧ Fx ∧ ~GA)

~⋁x (Ix ∧ Fx) ∧ ~GA

⋀x (Hx → ( Gx → Fx ∨ Ix)) Among H’s, only F’s and I’s are G.

⋀x (Ix ∧ Gx → Fx) I’s who are G’s are F’s.

⋀x (Ix → Fx) ∧ ⋀x (Ix → Gx)

I’s, who are G’s, are F’s.

⋀x (Ix → Gx ∧ Fx)

⋀x (Ix → Gx) → ⋀x (Fx → Gx) If every I is G, then any F is G.

⋁x (Fx ∧ Gx) → ⋀x (Ix → Gx) If only F’s are G’s, then every I is G.

GA → ⋀x (Fx → Gx) If A is G, then any F is G.

⋁x (Fx ∧ Gx) → GA If any F is G, then A is G.

⋀x (Fx ∧ Gx → Hx)

If any F is G, then he/she/it is H.

⋀x (Fx → (Gx → Hx)

↪ New form of derivation: **U**niversal **D**erivation - **UD** (K&M, p. 143)

n. ~~Show~~ ⋀α Φα Assertion **n+1**, UD

n+1. ~~Show~~ Φα

Then follow the appropriate strategy.

- If Φα is a conditional, do CD;

- If Φα is a negation, ID;

- If Φα is a disjunction, derive the corresponding conditional;

- If Φα is a conjunction, derive each of the conjuncts;

- If Φα is a biconditional, derive both directions of the biconditional.

↪ New Rules of Inference (K&M, p. 141)

- proper substitution (K&M, p. 139): Φβ comes from proper substitution of β for α if Φβ is just like Φα except for having free occurrences of β whenever Φα has free occurrences of α.

UI n. ⋀α Φα

Φβ n, UI/β

\*\* Where Φβ comes from Φα by proper substitution of the term β for the variable α in Φα \*\*

EG n. Φβ

⋁α Φα n, EG

\*\* Where Φβ comes from Φα by proper substitution of the term β for the variable α in Φα \*\*

EI n. ⋁α Φα

Φβ n, EI/β

\* Where Φβ comes from Φα by proper substitution of the term β for the variable α in Φα; AND

\* β is a variable; AND

\* β is a new variable, i.e., doesn’t occur anywhere in the derivation.

- Examples:

\*\* In order to follow a derivation, you have to read carefully the annotation. Make sure you know which lines and inference rules are being used to justify a line \*\*

Deriv 3.001: ⋀x (Fx ∧ Gx) ∴ ⋀xGx

Once I show Gx, I can box and cancel line 1.

1. ~~Show~~ ⋀xGx 2, UD

Remember: UD is not an inference rule ‼

2. ~~Show~~ Gx 4, DD

3. Fx ∧ Gx Pr UI/x

4. Gx 3, S

Deriv 3.002: ⋁xFx ∴ ⋁x(Gx⟶Fx)

Hint: EI as soon as you can ‼

1. ~~Show~~ ⋁x(Gx⟶Fx) 4 DD

2. Fa Pr EI/a

3. Ga⟶Fa 2 RT2

4. ⋁x(Gx⟶Fx) 3 EG

Deriv 3.004: ⋀x(Fx⟶Gx). ⋀x(Gx⟶Hx) ∴ FA⟶⋁x(Gx ∧ Hx)

1. ~~Show~~ FA⟶⋁x(Gx ∧ Hx) 3, CD

2. FA Ass CD

3. ~~Show~~ ⋁x(Gx ∧ Hx) 10, 11 ID

4. ~⋁x(Gx ∧ Hx) Ass ID

5. ⋀x ~(Gx ∧ Hx) 4, QN

6. FA⟶GA Pr1, UI/A

7. GA 2, 6 MP

8. GA⟶HA Pr2, UI/A

9. HA 6, 7 MP

10. ~(GA ∧ HA) 5 UI/A

11. GA ∧ HA 7,9 ADJ

Deriv 3.014: ⋀x(Fx⟶Gx). ⋁x((Fx ∧ Hx) ∨ (Fx ∧ Jx))⟶∼⋀x(Fx⟶Gx) ∴

⋀x(Fx⟶∼Jx)

Remember: to use UD, you have to show the formula that follows the quantifier, in this case, Fx⟶∼Jx. For this reason I introduced a new show line.

1. ~~Show~~ ⋀x(Fx⟶∼Jx) 2 UD

2. ~~Show~~ Fx⟶∼Jx 4 CD

3. Fx Ass CD

Just our usual strategy to show a negation, i.e., assume the thing that is being negated.

4. ~~Show~~ ∼Jx 10, 11 ID

5. Jx Ass ID

6. ~⋁x((Fx ∧ Hx) ∨ (Fx ∧ Jx)) Pr1 DN, Pr2 MT

7. ⋀x∼((Fx ∧ Hx) ∨ (Fx ∧ Jx)) 6 QN

Remember: no restrictions upon the term you instantiate by UI ‼

8. ∼((Fx ∧ Hx) ∨ (Fx ∧ Jx)) 7 UI/x

9. ∼(Fx ∧ Hx) ∧ ~(Fx ∧ Jx) 8 DM

10. ~(Fx ∧ Jx) 9 S

11. Fx ∧ Jx 3,5 ADJ

Deriv 3.715: ⋀x⋁y(Fx ∨ ∼Gy). ⋁x⋀y(Gy ∨ Hx) ∴ ∼⋁xHx⟶⋀xFx

1. ~~Show~~ ∼⋁xHx⟶⋀xFx 3 CD

2. ∼⋁xHx Ass CD

Pr1 UI/x: ⋁y(Fx ∨ ∼Gy), then EI/a.

3. ~~Show~~ ⋀xFx 4 UD

Just our usual strategy to show an atomic sentence, i.e., assume the negation.

4. ~~Show~~ Fx 10, 12 Id

5. ~Fx Ass ID

6. Fx ∨ ∼Ga Pr1 UI/x, EI/a

Remember the restriction: NEW VARIABLE ‼

7. ∼Ga 5, 6 MTP

8. ⋀y(Gy ∨ Hb) Pr2 EI/b

9. Ga ∨ Hb 8 UI/a

I decided to UI to *a* because I have ~Ga on line 7.

10. Hb 7, 9 MTP

11. ⋀x~Hx 2 QN

12. ~Hb 11 UI/b

↪ Exercises to practice:

\* For each of the following expression, state whether or not it is a well formed formula. If an expression is a symbolic formula, give the tree of formation. (Examples: K&M, p.121)

Pars 3.002:⋁x∼(Fx) Pars 3.011: ⋁x(E⟶Fx)

Pars 3.012: ⋀A(FA⟶∼GA) Pars 3.017: ⋀a(Hx⟷Gy)

Pars 3.026: ∼⋀x∼⋁yFx ∧ ∼Gy Pars 3.027: ⋀x(FGx⟶Gy)

Pars 3.028: ⋁xFx ∧ ⋁xGx⟶⋁x(Fx ∧ Gx) Pars 3.030: ⋁x(P⟶⋀x∼Qx)

\* Determine which inference rule, if any, the following arguments instantiate:

Recog 3.001: ⋀x(Fx⟶Gy) Recog 3.002: Gx

Fx⟶Gy ⋀xGx

Recog 3.004: ⋁yGy Recog 3.006: ⋁xGy

GA Gz

Recog 3.007: ⋀x⋁y(Fx⟶Gy∨Hx) Recog 3.011: FA⟶GA

⋁y(FA⟶Gy∨HA) ⋁y(Fy⟶Gy)

Recog 3.018: (⋀xFx⟶⋁y(Hy∨Hx)) Recog 3.020: ⋁z(FA∧Gz)⟶⋁xHx∨GA

FB⟶⋁y(Hy∨HB) ⋁x(⋁z(FA∧Gz)⟶⋁xHx∨Gx)

Recog 3.028: ⋁xFx⟶FA∨⋀xGx Recog 3.030: ⋀x(Fx⟶⋁y(FB∧Gy))

⋁x(⋁xFx⟶Fx∨⋀xGx) ⋀x(Fx⟶⋁z⋁y(Fz∧Gy))

↪ Do as many derivations as you can on the software ‼