Tableau with Quantifiers (or “Qu-Tee”):

1. When we covered sentential logic, we first introduced syntax, then semantics, and then finally proof (Tableau). For predicate logic, we will address proof (Quantifier Tableau, or “Qute”) before we move on to semantics. In part, this will reinforce the idea that the single turnstile (defined in terms of tree closure) is conceptually independent of the double turnstile. However, there are some further virtues as well, which I hope will emerge.

2. The Game of Tableau with quantifier formulas gets a little more involved.

First, here are the rules for existential and universal formulas on the left and right, respectively:

<table>
<thead>
<tr>
<th>∃x φ</th>
<th>( φ(α/x) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>φ (α/x)</td>
<td>for some ( α ) that is new to that path</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>∀x φ</th>
<th>φ(α/x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>φ(α/x)</td>
<td>for every name letter that ever appears on that path (if none have appeared yet, just use any name letter)</td>
</tr>
</tbody>
</table>

These rules are really quite simple; they instruct us to instantiate quantifier formulas on the same side of the path on which the quantifier formula appears. For existentials on the left and universals on the right, the name we use to replace the variable of instantiation must be new to that path(s) being developed. I often call these the “one-shot” rules. I call the other two rules, universal on the left and existential on the right, “every” or “all-shot” rules, because they may be developed for every name that ever appears on one of their subordinate paths, even (and especially) those that may be introduced much later in that path’s development.

3. This feature introduces a whole new wrinkle to the game of tableau. For the presence of the “all-shot” rules can make it very hard to develop a tree’s paths fully. Indeed, unlike tableau limited to the sentential connectives, once we’ve added these quantifier rules, not all initial setups can be fully developed. Here’s a simple example:

\[
\begin{align*}
∀x & \exists y \text{ Fxy} \\
\exists y & \text{ Fay} \\
& \text{ Fab} \\
\exists y & \text{ Fby} \\
& \text{ Fbc} \\
\exists y & \text{ Fcy} \\
& \ldots \text{ And so on.}
\end{align*}
\]
Unfortunately, what this means is that when we play the game of tableau with quantifiers, we will want to instantiate quantifier formulas on the left and the right selectively, with an eye toward closing (or killing!) off branches as quickly as possible. This can be challenging, as evidenced by some of the exercises below.

The fact that tableaus with quantifiers cannot be developed fully will also create some headaches for our eventual proof of completeness (think about why this would be so).

3. But first, let’s consider some exercises:

Construct Tableau proofs to show that all of the following can be closed.

As you go about these exercises, don’t forget to obey restrictions and to try to close off branches as quickly as you can. Also, be sure not to develop or instantiate interior quantifiers and only to develop one quantifier at a time.

1. ∃x(P&Fx) |-- P & ∃xFx  [Remember that |-- asks you to close off two separate tableaus.]

2. ∃x(P vFx) |-- P v ∃xFx

3. ∃x(Fx v Gx) |-- ∃xFx v ∃xGx

4. ∀x∀y(Rxy → ~Ryx) |-- ∀x~Rxx

5. ∀x∀y∀z( (Rxy&Ryz) → ~Rxz) |-- ∀x~Rxx

6. ∀x∀y∀z ( (Rxy&Ryz) → Rxz), ∀x∀y(Rxy → Ryx) |-- ∀x∃y(Rxy v Ryx) → Rxx

7. ∀x(Hx → ∀y(Fy → Gxy)), ∃x(Hx & ∃y ~Gxy) |-- ∀x∃xFx

8. ∀x(Kx → (∃yLxy → ∃zLzx)), ∀x(∃zLzx → Lxx), ~∃xLxx |-- ∀x(Kx → ∀y~Lxy)

9. ∀x(Ox → ∀y(Ry → ~Lxy)), ∀x(Ox → ∃y(Hy&Lxy)), ∃xOx |-- ∃x(Hx&~Rx)

10. ~∀x(Fx → Gx), ∀x(Hx → Gx), ∀x(~(Hx v Gx) → ∃y(Axy v Byx)), ~∃x∃yAxy |-- ∃x∃yByx

11. (∀y)(∀y)(Dy&Kyx) → (∀z)(Az→Hxz), (∀y)(∀y)(Fxy→ ~Hxy), ~(∀x)(Ax→ ~Dx), (∀x)(Cx→ (∃y)Kyxy) |-- (∀x)(Cx → (∃y)(Dy&~Fxy))

12. ∃x(Px & ∀y((Sy&∃z(Pz&Lyz)) → Lyx)), ∀x(Sx → ∃y(Py&Lxy)) |--
\exists x(Px & \forall y(Sy \rightarrow Ly))

13. \forall x \exists y(Fx & Gy) \vdash \exists y \forall x(Fx & Gy)

14. \forall x \exists y(Fx \rightarrow Gy) \vdash \exists y \forall x(Fx \rightarrow Gy)

15. \forall x \exists y((Fx \rightarrow Gy) & (Gy \rightarrow Fx)) \vdash \exists y \forall x(Fx \rightarrow Gy) & \exists y \forall x(Gy \rightarrow Fx)

16. \forall x \exists y((Fx \rightarrow Gy) & (Gy \rightarrow Fx)) \dashv\vdash \exists y \exists z \exists x((Fx \rightarrow Gy) & (Gz \rightarrow Fx))

[Closing the tableau of 16 from left to right is difficult. It might help to consider a modification of the solution to 15.]