Truth in Predicate Logic

Consider the sentence "All turtles are green." Let "T*x*" mean "*x* is a turtle," and "G*x*" mean "*x* is green." In PL, we symbolize the sentence: (x)(Tx ⊃ Gx).
We might want to know if the sentence is true or false. In SL we could tell if any sentence was true or false based on the truth-values of the atomic sentences, which we could list out. We don't exactly have atomic sentences in PL, so what do we need to make a list of in order to determine the truth of our sentence? We need an *interpretation*.

Obviously we need to know all the things that are turtles and all the things that are green. Let's assume, for convenience sake, that the only turtles are Leonardo, Michelangelo, Donatello, and Raphael, and that the only things that are Leonardo, Michelangelo, Donatello, Raphael. We can list these facts by providing two sets (a mathematical name for a collection of objects).

T*x*: {l, m, d, r}
G*x*: {l, m, d, r}[1]

That would help us for (x)(Tx ⊃ Gx), but it wouldn't help us for other sentences like (x)Tx. (x)Tx says "Everything is a turtle. To know if this sentence is true or false, it may not be enough to know what all the turtles are and what all the green things are. We also need to know what all the things are. The list of all the things is called the domain, which we can also list in a set. If our domain is just {l, m, d, r}, then (x)Tx is true. If our domain includes Splinter who is not a turtle then (x)Tx is false.

So an interpretation includes the domain (a list of all the objects),[2] the predicates we're interested in, and the list of objects that *satisfy* those predicates.[3] You may also need a dictionary to know what the interpretation is saying. It may seem weird giving a domain, but notice that many times we often (and usually!) restrict the domain in our own conversations. If I "Jim drank all the beers." I've usually restricted the domain to things at the party, not everything in the universe. We take myself to mean that Jim drank all the beers in the party, not that he drank all the beers in the universe. This is a kind of domain restriction.

---

[1] We use braces {...} around sets and commas in between the elements. The order doesn't matter. So {m, d} is the same set as {d, m}.
[2] For the purposes of this class, we will not consider empty domains.
[3] An object *x* satisfies a predicate *F* if and only if *Fx* is true.

Here's an interpretation:
𝓓: {a, d, l, m, r, s}
T*x*: {d, l, m, r}
G*x*: {d, l, m, r}
C*x*: {a, l, m, r}
R*x*: {r, s}
P*x*: {m}

Here's a dictionary:
a ≡ April; d ≡ Donatello; m ≡ Michelangelo; r ≡ Raphael; s ≡ Splinter
T*x* ≡ *x* is a turtle. G*x* ≡ *x* is green. C*x* ≡ *x* is cool. R*x* ≡ *x* is rude. P*x* ≡ *x* is a party dude.

Now we can determine the truth value for any PL sentences using these predicates for this domain.

Exercise 1: Determine whether the following sentences are true or false.

a. ~(x)Px
b. (∃x)(Px · Cx)
c. (x)(Tx ⊃ Cx)
d. (∃x)(Tx · Gx)
e. (∃x)(Tx ⊃ ~Gx)
f. (x)((Cx · Rx) ⊃ Px)
g. (∃x)((Tx · (Cx · Rx))
h. (∃x)(~Tx · Cx)
i. (x)(T ⊃ (Cx v ~Rx))
j. (∃x)(~Tx · Cx)
k. (x)(Px ⊃ Cx)

Exercise 2: Create an interpretation where the following sentence is true (remember you can use a different domain that the one above, but there must be at least one object in the domain).
(x)(Tx ⊃ Px)

Exercise 3: Create and interpretation where the following three sentences are true.
(x)(Tx ⊃ Cx)
~(∃x)(Cx · ~Tx)
(x)(Gx ⊃ (Tx v ~Cx))

Validity in Predicate Logic

Now that we're armed with interpretations, we can tackle validity in PL. The first thing to notice is that truth tables aren't going to be of any use. In SL there were only 2 ways the world could be for an atomic sentence *p*; 4 ways the world could be for atomic sentences *p* and *q*, etc:

p	*q*
T	T
T	F
F	T
F	F

Those possible ways the world could be were represented on our truth tables. Once we listed those truth assignments, we could determine the truth value for any sentence give any assignment. We could procedurally list every way the world could be, and then check each of these ways for a counter-example for any argument. If there was no counter-example, the argument was valid.

An argument logic in PL is valid when it's impossible for there to be any interpretation in which the premises are true, and the conclusion is false. A counter-example in PL, then, consists of an interpretation in which all the premises of an argument are true, but the conclusion is false.

EXAMPLE
Consider the argument: (x)(Gx ⊃ Hx); (∃x)(Gx · Fx), Fa // Fa · Ha
And counterexample is the interpretation 𝓓:{a, b} Gx: {b} Fx: {a, b} Hx: {b}

If there were something like a truth table for PL arguments, the interpretations would be the lines of the table, and we would check each interpretation for a counterexample. However, for any sentence (and any argument), there are infinitely many interpretations because there are infinitely many possible domains.

This means that we'll be using the proof method we used for SL with some added rules to prove validity. To prove invalidity, we'll have to creatively come up with an interpretation that serves as a counter-example. We'll practice both of these methods in the future.