State Updates, Satisfaction of Quantified Predicates

CS 536: Science of Programming, Fall 2019

A. Why?

- A predicate is satisfied relative to a state; it is valid if it is satisfied in all states.
- State updates occur when we introduce new variables or change the values of existing variables.

B. Outcomes

At the end of this lecture, you should

- Know what it means to update a state.
- Know what it means for a quantified predicate to be valid of be satisfied in a state.

C. "Updating" States

- To check quantified predicates for satisfaction, we need to look at different states that are related to, but not identical to, our starting state.
- **Example 1**: For $\{y = 1\} \vDash \forall x \in \mathbb{N} . x^2+1 \geq y-1$, we need to know that $\{y = 1, x = \alpha\} \vDash x^2 +1 \geq y-1$ for every natural number α. I.e., we need
    - $\{y = 1, x = 0\} \vDash x^2+1 \geq y-1$
    - $\{y = 1, x = 1\} \vDash x^2+1 \geq y-1$
    - $\{y = 1, x = 2\} \vDash x^2+1 \geq y-1$
    - ….
  - Similarly, for $\{z = 4\} \vDash \exists x \in \mathbb{N} . x \geq z$, we need $\{z = 4, x = \alpha\} \vDash x \geq z$ for some particular natural number α (α = 5 works nicely).
- There is a complicating factor. If the quantified variable already appears in the state, then we need to **replace** its binding with one that gives the value we're interested in checking.
- **Example 2**: We already know $\{z = 4\} \vDash \exists x \in \mathbb{N} . x \geq z$ because $\{z = 4, x = 5\} \vDash x \geq z$. If we start with the state $\{z = 4, x = -15\}$, which already has a binding for x, we find that the new state $\vDash \exists x \in \mathbb{N} . x \geq z$ because once again, $\{z = 4, x = 5\} \vDash x \geq z$ holds.
- In **Example** 2, the x that appears in $\{z = 4, x = 5\}$ is not the same x that appears within $\exists x \in \mathbb{N} . x \geq z$. However, the two x's in "$\{z = 4, x = 5\} \vDash x \geq z$" **are** the same x. Giving the two x's the same name causes the confusion. If we gave the x's different names, there'd be no problem with understanding; let xo be the "outer" x and xi be the "inner" x, then

$$\{z = 4, xo = -15\} \vDash \exists xi \in \mathbb{N} . xi \geq z$$

  because

$$\{z = 4, xo = -15, xi = 5\} \vDash xi \geq z$$

- When we use the same name $\mathtt{x}$, the binding for the outer $\mathtt{x}$ becomes invisible, overridden by the binding for the inner $\mathtt{x}$:

$$\{\mathtt{z} = 4, \text{(outer) } \mathtt{x} = -15\} \vDash \exists\, \mathtt{x} \in \mathbb{N} \,.\, \mathtt{x} \geq \mathtt{z} \text{ because } \{\mathtt{z} = 4, \mathtt{x} = 5\} \vDash \mathtt{x} \geq \mathtt{z}$$

- **Definition**: For any state σ, variable $\mathtt{x}$, and value α, the **update of** σ **at** $\mathtt{x}$ **with** α (written $\sigma[\mathtt{x} \mapsto \alpha]$) is the state that is a copy of σ except that it binds variable $\mathtt{x}$ to value α.
  - Let $\tau = \sigma[\mathtt{x} \mapsto \alpha]$, then $\tau(\mathtt{x}) = \alpha$; if variable $y \not\equiv \mathtt{x}$, then $\tau(y) = \sigma(y)$.
  - Note $\tau(\mathtt{x}) = \alpha$ regardless of whether $\sigma(\mathtt{x})$ is defined or not. If $\sigma(\mathtt{x})$ is defined, its type and exact value are irrelevant.
- Set theoretically,
  - If $\mathtt{x}$ has no binding in σ, then $\sigma[\mathtt{x} \mapsto \alpha]$ is $\sigma \cup \{\mathtt{x} = \alpha\}$: It's like σ but has been extended with $\mathtt{x} = \alpha$.
  - If $\mathtt{x}$ has a binding in σ, say $\sigma = \{\mathtt{x} = \beta\} \cup \sigma_0$ where σ_0 is the rest of σ, then $\sigma[\mathtt{x} \mapsto \alpha]$ is $\sigma_0 \cup \{\mathtt{x} = \alpha\}$. It's like σ but has the binding $\mathtt{x} = \alpha$, not $\mathtt{x} = \beta$. (Having two bindings for $\mathtt{x}$ would be illegal.)
- **Important**: Calling it the "update" of σ is kind of misleading because we're not modifying σ*.
  - Taking $\sigma[\mathtt{x} \mapsto \alpha]$ **does not do** an update in place; if we define $\tau = \sigma[\mathtt{x} \mapsto \alpha]$, then σ is still σ.
  - Conceptually, we aren't modifying σ, we're creating a new state.
- We're not required to give $\sigma[\mathtt{x} \mapsto \alpha]$ a new name; we can write it out explicitly:
  - If $\mathtt{x} \equiv y$, then $\sigma[\mathtt{x} \mapsto \alpha](y) = \alpha$, otherwise (if $\mathtt{x} \not\equiv y$), then $\sigma[\mathtt{x} \mapsto \alpha](y) = \sigma(y)$.
  - (You have to read $\sigma[\mathtt{x} \mapsto \alpha](y)$ left-to-right — we're taking the function $\sigma[\mathtt{x} \mapsto \alpha]$ and applying it to y. I.e., $\sigma[\mathtt{x} \mapsto \alpha](y) = (\sigma[\mathtt{x} \mapsto \alpha])(y)$, where the left pair of parentheses are for grouping and the ones around y are for the function call.)
- **Example 3**: If $\sigma = \{\mathtt{x} = 2, \mathtt{y} = 6\}$, then $\sigma[\mathtt{x} \mapsto 0] = \{\mathtt{x} = 0, \mathtt{y} = 6\}$:
  - $\sigma[\mathtt{x} \mapsto 0](\mathtt{x}) = 0$ (Even though $\sigma(\mathtt{x}) = 2$)
  - $\sigma[\mathtt{x} \mapsto 0](\mathtt{y}) = \sigma(\mathtt{y}) = 6$ (Since we didn't update $\mathtt{y}$)
  - $\sigma[\mathtt{x} \mapsto 0](\mathtt{x+y}) = 0+6 = 6$ (Since the $\mathtt{x}$ in $\mathtt{x+y}$ gets evaluated to 0)
  - $\sigma[\mathtt{x} \mapsto 0] \vDash \mathtt{x}^2 \leq 0$ (Even though our starting $\sigma \nvDash \mathtt{x}^2 \leq 0$)
- The value part of an update has to be a semantic value, not a syntactic one, so $\sigma[\mathtt{x} \mapsto \mathtt{x+1}]$ isn't well-formed.
  - In these notes, it may help to remember that since `x+1` is in `this font`, it's syntactic.
  - On the other hand, "$\sigma[\mathtt{x} \mapsto \sigma(\mathtt{x+1})]$" or "$\sigma[\mathtt{x} \mapsto \alpha \text{ plus one}]$ where $\alpha = \sigma(\mathtt{x})$" do make sense.

Multiple Updates

- We can do a sequence of updates on a state. E.g., $\sigma[\mathtt{x} \mapsto 0][\mathtt{y} \mapsto 8]$ is a doubly updated state. Sequences of updates are read left-to-right, so this is $(\sigma[\mathtt{x} \mapsto 0])[\mathtt{y} \mapsto 8]$.
  - **Example 4**: If $\sigma = \{\mathtt{x} = 2, \mathtt{y} = 6\}$, then $\sigma[\mathtt{x} \mapsto 0][\mathtt{y} \mapsto 8] = \{\mathtt{x} = 0, \mathtt{y} = 6\}[\mathtt{y} \mapsto 8] = \{\mathtt{x} = 0, \mathtt{y} = 8\}$.
- The order of update doesn't matter if you have two different variables.
  - **Example 5**: $\sigma[\mathtt{x} \mapsto 0][\mathtt{y} \mapsto 8] = \sigma[\mathtt{y} \mapsto 8][\mathtt{x} \mapsto 0]$.

* Unfortunately, "update" is the traditional name, and for myself, I can't find any word that's exactly right. We're not always *extending* σ, we're not always *superseding* σ, ….

- If you update the same variable twice, the second update supersedes the first.
  - **Example 6**: $\sigma[\mathtt{x} \mapsto 0][\mathtt{x} \mapsto 17] = \sigma[\mathtt{x} \mapsto 17] \neq \sigma[\mathtt{x} \mapsto 17][\mathtt{x} \mapsto 0] = \sigma[\mathtt{x} \mapsto 0]$
  - Of course, if the second update is identical to the first, nothing happens: $\sigma[\mathtt{x} \mapsto \alpha][\mathtt{x} \mapsto \alpha] = \sigma[\mathtt{x} \mapsto \alpha]$
- If you have to evaluate an expression, be sure to do it in the correct state.
  - Let $\sigma(\mathtt{x}) = 1$ and let $\tau = \sigma[\mathtt{x} \mapsto 2]$, then $\tau[\mathtt{z} \mapsto \sigma(\mathtt{x})+10]$ maps $\mathtt{z}$ to $\sigma(\mathtt{x})+10 = 1+10 = 11$. We can omit τ and also write $\sigma[\mathtt{x} \mapsto 2][\mathtt{z} \mapsto \sigma(\mathtt{x})+10]$, which gives the same state as τ.
  - On the other hand, $\tau[\mathtt{z} \mapsto \tau(\mathtt{x})+10]$ maps $\mathtt{z}$ to $\tau(\mathtt{x})+10 = 2+10 = 12$. Here, if we don't give a name to $\sigma[\mathtt{x} \mapsto 2]$, then we can't write $\tau[\mathtt{z} \mapsto \tau(\mathtt{x})+10]$ so we have to write $\sigma[\mathtt{x} \mapsto 2][\mathtt{z} \mapsto \sigma[\mathtt{x} \mapsto 2](\mathtt{x})+10]$. (This is pretty ugly, so giving $\sigma[\mathtt{x} \mapsto 2]$ a name like τ makes things more readable.)

D. Updating Array Values

- Updating array elements like `b[0]` is a bit more complicated than updating simple variables like `x` and `y`. First, let's extend our notion of updating states to updating general functions.
- **Definition**: If φ is a function on one argument and α and β are valid members of the domain and range of φ respectively, then the **update of φ at α with β**, written $\varphi[\alpha \mapsto \beta]$, is the function defined by $\varphi[\alpha \mapsto \beta](\gamma) = \beta$ if $\gamma = \alpha$ and $\varphi[\alpha \mapsto \beta](\gamma) = \varphi(\gamma)$ if $\gamma \neq \alpha$.
- **Definition**: If σ is a (proper) state for an array $\mathtt{b}$ and α is a valid index value for $\mathtt{b}$, then $\sigma[\mathtt{b}[\alpha] \mapsto \beta]$ means $\sigma[\mathtt{b} \mapsto \gamma[\alpha \mapsto \beta]]$ where $\gamma =$ the function $\sigma(\mathtt{b})$ In words, if σ includes the binding $\mathtt{b} =$ function γ, then the updating σ at $\mathtt{b}[\alpha]$ with β is just like updating σ at $\mathtt{b}$ with an updated version of γ, namely $\gamma[\alpha \mapsto \beta]$.
- **Example 7**: Say $\sigma = \{\mathtt{x} = 3, \mathtt{b} = (2, 4, 6)\}$, then $\sigma[\mathtt{b}[0] \mapsto 8] = \{\mathtt{x} = 3, \mathtt{b} = (8, 4, 6)\}$. Here, $\sigma(\mathtt{b})$ is the function $(2, 4, 6)$ (which means $\{(0, 2), (1, 4), (2, 6)]$), so $\sigma(\mathtt{b})[0 \mapsto 8]$ (the update of function $\sigma(\mathtt{b})$) is the function $(2, 4, 6)[0 \mapsto 8] = (8, 4, 6)$.

E. Satisfaction of Quantified Predicates

- One use of updated states is for describing how assignment works. (We'll see this later.) The other use for updated states is for defining when quantified predicates are satisfied.
- **Definition**: $\sigma \vDash \exists\, \mathtt{x} \in S\,.\, p$ if for one or more **witness** values $\alpha \in S$, it's the case that $\sigma[\mathtt{x} \mapsto \alpha] \vDash p$. Note we're asking a hypothetical question: "If we were to calculate $\sigma[\mathtt{x} \mapsto \alpha]$, would we find that it satisfies p?"
  - **Example 8a**: For any state σ, we can show $\sigma \vDash \exists\, \mathtt{x}\,.\, \mathtt{x}^2 \le 0$ using 0 as the witness: $\sigma[\mathtt{x} \mapsto 0] \vDash \mathtt{x}^2 \le 0$, since $\sigma[\mathtt{x} \mapsto 0](\mathtt{x}^2 \le 0) = \sigma[\mathtt{x} \mapsto 0](\mathtt{x}^2) \le \sigma[\mathtt{x} \mapsto 0](0) = (0^2 \le 0) = \mathrm{T}$.
    - Remember, $\sigma(\mathtt{x})$ is irrelevant, since $\sigma[\mathtt{x} \mapsto \alpha]$ overrides any value for $\sigma(\mathtt{x})$.
  - **Example 8b**: If $\sigma(\mathtt{x})$ is, say 5, it's still the case that $\sigma \vDash \exists\, \mathtt{x}\,.\, \mathtt{x}^2 \le 0$ using 0 as the witness because we $\sigma[\mathtt{x} \mapsto 0] \vDash \mathtt{x}^2 \le 0$, regardless of $\sigma(\mathtt{x}) = 5$.
- If there are many successful witness values, we don't have to specify all of them; we just need one.
  - **Example 12**: If $\sigma(\mathtt{y}) = 3$, then $\sigma \vDash \exists\, \mathtt{x}\,.\, \mathtt{x}^2 \le \mathtt{y}$ with $\mathtt{x} = 0$ or 1 as possible witness values.

- **Definition**: $\sigma \vDash \forall\, \mathtt{x} \in S.\, p$ if for every value $\alpha \in S$, we have $\sigma[\mathtt{x} \mapsto \alpha] \vDash p$. (Again, this is hypothetical: "If for every α, we were to calculate $\sigma[\mathtt{x} \mapsto \alpha]$, would we find that it satisfies p?"
  - **Example 10**: To know $\sigma \vDash \forall\, \mathtt{x} \in \mathbb{Z} \,.\, \mathtt{x}^2 \geq \mathtt{x}$, we need to know $\sigma[\mathtt{x} \mapsto \alpha] \vDash \mathtt{x}^2 \geq \mathtt{x}$ for every $\alpha \in \mathbb{Z}$. Since for every integer α, indeed α^2 is $\geq \alpha$, this does hold. Recall that it doesn't matter what $\sigma(\mathtt{x})$ is, since we're interested in $\sigma[\mathtt{x} \mapsto \alpha]$.
- When asking if σ satisfies $\forall\, \mathtt{x} \in S.\, q$ or $\exists\, \mathtt{x} \in S.\, q$, we don't care about $\sigma(\mathtt{x})$. For a predicate p in general, for the question "Does $\sigma \vDash p$?" only depends on how σ operates on the non-quantified variables of p.
  - **Example 11**: Since the body of $\forall\, \mathtt{x} \in \mathbb{Z} \,.\, \mathtt{x}^2 \geq \mathtt{x}$ uses only the quantified variable $\mathtt{x}$, it doesn't matter what bindings σ has when checking $\sigma \vDash \forall\, \mathtt{x} \in \mathbb{Z} \,.\, \mathtt{x}^2 \geq \mathtt{x}$. Even $\sigma = \varnothing$ works: $\varnothing \vDash \forall\, \mathtt{x} \in \mathbb{Z} \,.\, \mathtt{x}^2 \geq \mathtt{x}$.
- Note with nested quantifiers, the notation does get more complicated.
- **Example 12**: $\sigma \vDash \forall\, \mathtt{x} \geq \mathtt{y}^2 \,.\, \exists\, \mathtt{z} \,.\, \mathtt{z} > \mathtt{x}+\mathtt{y}^2$ iff (for every $\alpha \in \mathbb{Z}$, if $\alpha \geq \sigma(\mathtt{y})^2$, then there is some $\beta \in \mathbb{Z}$ such that $\beta > \alpha + \sigma(\mathtt{y})^2$).

  $\sigma \vDash \forall\, \mathtt{x} > \mathtt{y}^2 \,.\, \exists\, \mathtt{z} \,.\, \mathtt{z} \geq \mathtt{x}+\mathtt{y}^2$

iff $\sigma \vDash \forall\, \mathtt{x} \,.\, \mathtt{x} > \mathtt{y}^2 \rightarrow \exists\, \mathtt{z} \,.\, \mathtt{z} \geq \mathtt{x}+\mathtt{y}^2$	defn bounded $\forall$
iff for every $\alpha \in \mathbb{Z}$, $\sigma[\mathtt{x} \mapsto \alpha] \vDash \mathtt{x} > \mathtt{y}^2 \rightarrow \exists\, \mathtt{z} \,.\, \mathtt{z} \geq \mathtt{x}+\mathtt{y}^2$,	defn $\vDash \forall$

  Now, $\sigma[\mathtt{x} \mapsto \alpha] \vDash \mathtt{x} > \mathtt{y}^2 \rightarrow \exists\, \mathtt{z} \,.\, \mathtt{z} \geq \mathtt{x}+\mathtt{y}^2$

iff $\sigma[\mathtt{x} \mapsto \alpha] \vDash \mathtt{x} > \mathtt{y}^2$ implies $\sigma[\mathtt{x} \mapsto \alpha] \vDash \exists\, \mathtt{z} \,.\, \mathtt{z} \geq \mathtt{x}+\mathtt{y}^2$	defn $\vDash \rightarrow$
iff $\alpha > \gamma^2$ implies $\sigma[\mathtt{x} \mapsto \alpha] \vDash \exists\, \mathtt{z} \,.\, \mathtt{z} \geq \mathtt{x}+\mathtt{y}^2$	where $\gamma = \sigma(\mathtt{y})$
iff $\alpha > \gamma^2$ implies for some β, $\sigma[\mathtt{x} \mapsto \alpha][\mathtt{z} \mapsto \beta] \vDash \mathtt{z} \geq \mathtt{x}+\mathtt{y}^2$	defn $\vDash \exists$
iff $\alpha > \gamma^2$ implies for some β, $\beta \geq \alpha+\gamma^2$	defn $\vDash \geq$

  Taking $\beta = 2\alpha$ for our witness value, we need $\alpha > \gamma^2$ implies for some $2\alpha \geq \alpha+\gamma^2$, which is true.

  Note defining intermediate names like "let $\tau = \sigma[\mathtt{x} \mapsto \alpha][\mathtt{z} \mapsto \beta]$" is allowed, if you prefer that style.

Justifying DeMorgan's Laws for Quantified Predicates

- In general, we want our systems of reasoning to be **sound**: We want the textual transformations that make up logical equivalence to reflect truths about how our semantics work.
- **Example 15**: Here is a check of DeMorgan's law for existentials, which says $\neg \exists\, \mathtt{x} \,.\, p \Leftrightarrow \forall\, \mathtt{x} \,.\, \neg p$. Semantically, we want each of these to be valid if and only if the other is. So we need $\sigma \vDash \neg \exists\, \mathtt{x} \,.\, p$ if and only if $\sigma \vDash \forall\, \mathtt{x} \,.\, \neg p$.

  $\sigma \vDash \neg \exists\, \mathtt{x} \in S \,.\, p$

iff $\sigma \nvDash \exists\, \mathtt{x} \,.\, p$	defn of $\sigma \vDash \neg$predicate
iff for no $\alpha \in S$ do we have $\sigma[\mathtt{x} \mapsto \alpha] \vDash p$	defn of $\sigma \vDash$ existential
iff for every $\alpha \in S$ we have $\sigma[\mathtt{x} \mapsto \alpha] \nvDash p$	equivalence of "no $\vDash$" vs "every $\nvDash$"
iff for every $\alpha \in S$ we have $\sigma[\mathtt{x} \mapsto \alpha] \vDash \neg p$	defn of $\sigma \vDash \neg$predicate
iff $\sigma \vDash \forall\, \mathtt{x} \,.\, \neg p$	defn of $\sigma \vDash$ universal.

- By using this property of $\neg\exists$, we can get a short proof of soundness for the negation of a universal: For all σ,

$\sigma \models \neg\forall \mathtt{x}\,.\,p$	
iff $\sigma \models \neg(\forall \mathtt{x}\,.\,\neg\neg p)$	double $\neg$
iff $\sigma \models \neg(\neg\exists\, \mathtt{x}\,.\,\neg p))$	DeMorgan law ($\neg\,\exists$ vs $\forall\,\neg$)
iff $\sigma \models \exists\, \mathtt{x}\,.\,\neg p$	double $\neg$

Satisfaction, Validity, and State Updates

CS 536: Science of Programming, Fall 2019

A. Why

- A predicate is satisfied or unsatisfied relative to a state.
- A predicate is valid if it is satisfied in all states.
- State updates occur when we introduce new variables or change the values of existing variables.

B. Outcomes

At the end of today, you should

- Know how to check a predicate for satisfaction in a state, how to check a predicate for validity, and know how to update a state.

C. Questions

1. Say u and v stand for variables (possibly the same variable) and α and β are values (possibly equal). When is $\sigma[u \mapsto \alpha][v \mapsto \beta] = \sigma[v \mapsto \beta][u \mapsto \alpha]$? Hint: There are four cases because maybe $x \equiv y$, maybe $\alpha = \beta$.

2. Let $\sigma(\mathtt{b}) = (7, 5, 12, 16)$.
   a. Does $\sigma \vDash \exists\, \mathtt{k}\,.\, \mathtt{0} \le \mathtt{k} \wedge \mathtt{k+1} < \mathtt{size}(\mathtt{b}) \wedge \mathtt{b[k]} < \mathtt{b[k+1]}$? If so, what was your witness values for k?
   b. Does $\sigma \vDash \exists\, \mathtt{k}\,.\, \mathtt{0} \le \mathtt{k-1} \wedge \mathtt{k+1} < \mathtt{size}(\mathtt{b}) \wedge \mathtt{b[k-1]} < \mathtt{b[k]} < \mathtt{b[k+1]}$? If so, what was your witness values for k?
   c. Does $\sigma \vDash \forall\, \mathtt{k}\,.\, \mathtt{b[k]} > \mathtt{0}$?
   d. If $\sigma(\mathtt{k}) = -5$, then does $\sigma \vDash \exists\, \mathtt{k}\,.\, \mathtt{b[k]} > \mathtt{0}$?

3. For each of the situations below, fill in the blanks to describe when the situation holds.

   Fill in $\_\_\_\_\__1$ with "*some*", "*every*", or "*this*"
   Fill in $\_\_\_\_\__2$ with "*some*" or "*every*",
   Fill in $\_\_\_\_\__3$ with "$\sigma(\mathtt{x})$ must *be undefined*", "$\sigma(\mathtt{x})$ must *be defined and* $\sigma \vDash p$", or "*nothing of* $\sigma(\mathtt{x})$",
   Fill in $\_\_\_\_\__4$ with "$\vDash p$" or "$\nvDash p$".

   a. $\sigma \vDash (\exists\, \mathtt{x} \in U\,.\, p)$ iff for $\_\_\_\_\__1$ state σ and $\_\_\_\_\__2$ $\alpha \in U$, $\sigma[\mathtt{x} \mapsto \alpha]$ $\_\_\_\_\__4$
   b. $\sigma \vDash (\forall\, \mathtt{x} \in U\,.\, p)$ iff for $\_\_\_\_\__1$ state σ and $\_\_\_\_\__2$ $\alpha \in U$, $\sigma[\mathtt{x} \mapsto \alpha]$ $\_\_\_\_\__4$
   c. $\sigma \vDash (\exists\, \mathtt{x} \in U\,.\, p)$ requires $\_\_\_\_\__3$.
   d. $\sigma \vDash (\forall\, \mathtt{x} \in U\,.\, p)$ requires $\_\_\_\_\__3$.
   e. $\sigma \nvDash (\exists\, \mathtt{x} \in U\,.\, p)$ iff for $\_\_\_\_\__1$ state σ for $\_\_\_\_\__2$ $\alpha \in U$, $\sigma[\mathtt{x} \mapsto \alpha]$ $\_\_\_\_\__4$ p.
   f. $\sigma \nvDash (\forall\, \mathtt{x} \in U\,.\, p)$ iff for $\_\_\_\_\__1$ state σ for $\_\_\_\_\__2$ $\alpha \in U$, $\sigma[\mathtt{x} \mapsto \alpha]$ $\_\_\_\_\__4$ p.
   g. $\nvDash (\forall\, \mathtt{x} \in U\,.\, p)$ iff for $\_\_\_\_\__2$ state σ, we have σ $\_\_\_\_\__4$ $(\forall\, \mathtt{x} \in U\,.\, p)$.
   h. $\nvDash (\exists\, \mathtt{x} \in U\,.\, p)$ iff for $\_\_\_\_\__2$ state σ, we have σ $\_\_\_\_\__4$ $(\exists\, \mathtt{x} \in U\,.\, p)$.
   i. $\nvDash (\forall\, \mathtt{x} \in U\,.\, p)$ iff for $\_\_\_\_\__2$ state σ, and for $\_\_\_\_\__2$ $\alpha \in U$, we have $\sigma[\mathtt{x} \mapsto \alpha]$ $\_\_\_\_\__4$.

j. $\models (\exists\, \mathtt{x} \in U\ .\ (\forall\, \mathtt{y} \in V\ .\ p))$ iff for ______ $_1$ state σ, for ______ $_2$ $\alpha \in U$, and for ______ $_2$ $\beta \in V$, we have $\sigma[\mathtt{x} \mapsto \alpha][\mathtt{y} \mapsto \beta]$ ______ $_4$.

k. $\not\models (\exists\, \mathtt{x} \in U\ .\ (\forall\, \mathtt{y} \in V\ .\ p))$ iff for ______ $_1$ state σ, for ______ $_2$ $\alpha \in U$, and for ______ $_2$ $\beta \in V$, we have $\sigma[\mathtt{x} \mapsto \alpha][\mathtt{y} \mapsto \beta]$ $[\models \,|\, \models \neg]\ p$.

l. $\models (\forall\, \mathtt{x} \in U\ .\ (\exists\, \mathtt{y} \in V\ .\ p))$ iff for ______ $_1$ state σ, for ______ $_2$ $\alpha \in U$, and for ______ $_2$ $\beta \in V$, we have $\sigma[\mathtt{x} \mapsto \alpha][\mathtt{y} \mapsto \beta]$ $[\models \,|\, \models \neg]\ p$.

m. $\not\models (\forall\, \mathtt{x} \in U\ .\ (\exists\, \mathtt{y} \in V\ .\ p))$ iff for ______ $_1$ state σ, for ______ $_2$ $\alpha \in U$, and for ______ $_2$ $\beta \in V$, we have $\sigma[\mathtt{x} \mapsto \alpha][\mathtt{y} \mapsto \beta]$ ______ $_4$.

4. Let $p_1 \equiv \exists\, \mathtt{y}\,.\, \forall\, \mathtt{x}\,.\, \mathtt{f(x)} > \mathtt{y}$, and let $p_2 \equiv \forall\, \mathtt{x}\,.\, \exists\, \mathtt{y}\,.\, \mathtt{f(x)} > \mathtt{y}$. (As usual, assume a domain of $\mathbb{Z}$.)

   a. Is it the case that (regardless of the definition of `f`), if p_1 is valid then so is p_2? If so, explain why. If not, give a definition of $\mathtt{f(x)}$ and show $\models p_1$ but $\not\models p_2$.

   b. (Repeat part a in the other direction.) Is it the case that (regardless of the definition of `f`), if p_2 is valid then so is p_1? If so, explain why. If not, give a definition of $\mathtt{f(x)}$ and show $\models p_2$ but $\not\models p_1$.

CS 536: Solution to Activity 4 (Satisfaction, Validity, and State Updates)

1. $\sigma[u \mapsto \alpha][v \mapsto \beta] = \sigma[v \mapsto \beta][u \mapsto \alpha]$ iff $u \not\equiv v$ or ($u \equiv v$ and) $\alpha = \beta$.

2. (Quantified statements over arrays) Let $\sigma(\mathtt{b}) = (7, 5, 12, 16)$.
   a. Yes, $\sigma \vDash \exists\, \mathtt{k} \,.\, \mathtt{0} \le \mathtt{k} \wedge \mathtt{k+1} < \mathtt{size(b)} \wedge \mathtt{b[k]} < \mathtt{b[k+1]}$ with 1 and 2 as possible witnesses for $\mathtt{k}$.
   b. Yes, $\sigma \vDash \exists\, \mathtt{k} \,.\, \mathtt{0} \le \mathtt{k-1} \wedge \mathtt{k+1} < \mathtt{size(b)} \wedge \mathtt{b[k-1]} < \mathtt{b[k]} < \mathtt{b[k+1]}$ with 2 as the only witness that works.
   c. Yes, $\sigma \vDash \forall\, \mathtt{k} \,.\, \mathtt{b[k]} > \mathtt{0}$
   d. Yes, if $\sigma(\mathtt{k}) = -5$, we still have $\sigma \vDash \exists\, \mathtt{k} \,.\, \mathtt{b[k]} > \mathtt{0}$, with witnesses 0, 1, 2, 3. The key is that for σ to satisfy the existential with witness call it α, then we need $\sigma[\mathtt{k} \mapsto \alpha] \vDash \mathtt{b[k]} > \mathtt{0}$, which doesn't depend on $\sigma(\mathtt{k})$ because the update of σ uses $\mathtt{k} = \alpha$, not $\mathtt{k} =$ whatever $\sigma(\mathtt{k})$ happens to be. Here's a step-by-step explanation (this is way too much detail for a test):

      $\sigma[\mathtt{k} \mapsto \alpha] \vDash \mathtt{b[k]} > \mathtt{0}$

iff $\sigma[\mathtt{k} \mapsto \alpha](\mathtt{b[k]}) > \sigma[\mathtt{k} \mapsto \alpha](\mathtt{0})$	defn state $\vDash$ relational test
iff $(\sigma[\mathtt{k} \mapsto \alpha](\mathtt{b}))(\sigma[\mathtt{k} \mapsto \alpha](\mathtt{k})) > 0$	the value of $\mathtt{0}$ is zero
iff $(\sigma(\mathtt{b}))(\sigma[\mathtt{k} \mapsto \alpha](\mathtt{k})) > 0$	$\sigma[\mathtt{k} \mapsto \alpha](\mathtt{b}) = \sigma(\mathtt{b})$ because $\mathtt{b} \not\equiv \mathtt{k}$
iff $(\sigma(\mathtt{b}))(\alpha) > 0$	$\sigma[\mathtt{k} \mapsto \alpha](\mathtt{k})) = \alpha$
iff 7, 5, 12, or 16 > 0	depending on $\alpha = 0, 1, 2$, or 3

3. (Validity/invalidity of quantified predicates)
   a. this σ, some α, $\vDash p$
   b. this σ, every α, $\vDash p$
   c. nothing of $\sigma(\mathtt{x})$
   d. nothing of $\sigma(\mathtt{x})$
   e. this σ, every α, $\not\vDash p$
   f. this σ, some α, $\not\vDash p$
   g. some σ, $\not\vDash \forall\, x \in U.\ p$
   h. some σ, every α, $\not\vDash p$
   i. some σ, some α, $\not\vDash p$
   j. every σ, some α, every β, $\vDash p$
   k. some σ, every α, some β, $\not\vDash p$
   l. every σ, every α, some β, $\vDash p$
   m. some σ, some α, every β, $\not\vDash p$

4. ($\exists\forall$ predicates versus $\forall\exists$ predicates, specifically $p_1 \equiv \exists\, \mathtt{y} \,.\, \forall\, \mathtt{x} \,.\, \mathtt{f(x)} > \mathtt{y}$, and $p_2 \equiv \forall\, \mathtt{x} \,.\, \exists\, \mathtt{y} \,.\, \mathtt{f(x)} > \mathtt{y}$)
   a. The relation does hold: $\vDash p_1$ implies $\vDash p_2$. The short explanation is that for each value α for $\mathtt{x}$, we need to find a value β for $\mathtt{y}$ that satisfies the body, but p_1 says that there's a value that works for every α, so we can use that value for β. In more detail, assume p_1 is valid: for every state σ, there is some value β where for every value α, $\sigma[\mathtt{y} \mapsto \beta][\mathtt{x} \mapsto \alpha] \vDash \mathtt{f(x)} > \mathtt{y}$. To show that p_2 is valid, take an arbitrary state τ

with value α for $\mathtt{x}$. We need a witness value for the $\exists\,\mathtt{y}$; using p_1 with τ for σ, we get a β for the $\exists\,\mathtt{y}$ of p_1 and use that as the witness for the $\exists\,\mathtt{y}$ in p_2. So then we need $\tau[\mathtt{x} \mapsto \alpha][\mathtt{y} \mapsto \beta] \vDash \mathtt{f(x)} > \mathtt{y}$. Substituting σ for τ and swapping the order of the updates, we need $\sigma[\mathtt{y} \mapsto \beta][\mathtt{x} \mapsto \alpha] \vDash \mathtt{f(x)} > \mathtt{y}$. But that's exactly what p_1 provided.

b. The relation does not hold: We can have $\vDash p_2$ but $\nvDash p_1$. The easiest example is $\mathtt{f(x)} = \mathtt{x}$, then validity of p_1 would require us to find an integer value for $\mathtt{y}$ that is $>$ every possible integer value of $\mathtt{x}$, and no such value exists.

