Types, Expressions, and States

CS 536: Science of Programming, Fall 2021

1. Which of the following expressions are legal or illegal according to the syntax we’re using? Assume $x$, $y$, and $z$ are integer variables and $b$ is an array name.
   a. $(if \ x > \ y \ then \ x \ else \ y)$
   b. $(if \ x < \ y \ then \ -1 \ else \ (if \ x = \ y \ then \ 0 \ else \ 1))$
   c. $(if \ y = 0 \ then \ f \ else \ g)(17)$
   d. $b[0][1]$ /* What type must $b$ have for this to be legal? */
   e. $b$ /* Remember we’re given that $b$ is an array */

2. Which of the following are legal ways to write out a state? (And if not, why not?)
   a. $\{ x = 5, y = 2 \}$
   b. $\{ x = \text{five}, y = \text{one plus one} \}$
   c. $\{ x = 5, y = x \text{ minus } 3 \}$
   d. $\{ x = 5, y = \alpha - 3 \}$ where $\alpha = 5$
   e. $\{ x = 5, y = (\text{the value of $x$ in this environment minus } 3) \}$
   f. $\{ \}$

3. Let $e_4 \equiv x = y+1 \land y = z^2 - 3 \land z = 6$. Write out the textual definition of a state $\sigma_4$ in which $e_4$ evaluates to true. Use only bindings that map variables to constants. $\sigma_4 = \{ x = 34, y = 33, z = 6 \}$

4. Which of the following states are well-formed and also proper for the expression $b[i] + 0 * y$? If ill-formed, why? If taking the value might cause a runtime error, why?
   a. $\{ i = 0, b = (3, 4, 8), y = 3, z = 5 \}$
   b. $\{ i = 0, b = (6), y = 5 \}$
   c. $\{ i = 0, b = 6, y = 5 \}$
   d. $\{ i = 1, b = (3, 4, 8) \}$
   e. $\{ i = 1, i = 2, y = 0, b = (2, 6) \}$
   f. $\{ i = 5, b = (1, 2), y = 4 \}$
CS 536 Solution to Practice 3 (Types, Expressions, and States)

1. (Legal and illegal expressions)
   a. legal
   b. legal
   c. illegal because the conditional expression can’t yield a function/operator
   d. $b[0][1]$ is legal ($b$ must be a 2-dimensional array)
   e. $b$ (all by itself) is illegal, since $b$ we’ve assumed is an array

2. (Legal ways to represent states)
   a. \{ $x = 5, y = 2$ \} is legal
   b. \{ $x = \text{five}, y = \text{one plus one}$ \} is legal because “five” and “one” etc. refer to semantic objects.
   c. \{ $x = 5, y = x \text{ minus } 3$ \} is illegal: To be legal, “$x \text{ minus } 3$” has to be a value, so “$x$” has to be a value (it has to be the name of a mathematical object like 5). But the binding $x = 5$ tells us “$x$” is a variable that can appear in an expression, so “$x$” is a syntactic object. It can’t be syntactic and semantic at the same time.

   Also, in this nicely word-processed document, “$x$” is presented in this font, so we know it’s supposed to be a syntactic object. On paper, you see the difference between “$x$” and “$x$”. Even so, if someone wrote \{ $x = 5, y = x \text{ minus } 1$ \} on the blackboard, it would have to be illegal because of using $X$ in two incompatible ways.

   d. \{ $x = 5, y = \alpha - 3$ \} where $\alpha = 5$ — is legal. We infer that symbols $x$ and $y$ are syntactic objects and $\alpha$ is the name of the semantic object 5.
   e. \{ $x = 5, y = (\text{the value of } x \text{ in this environment, minus } 3)$ \} is legal. Since “the value of $x$ in this environment” is just another name (albeit complicated) for the mathematical object 5, it’s legal to use here.
   f. \{ \} is legal, since it’s just another way to write $\emptyset$, the empty state.

3. $\sigma_4 = \{ z = 6, y = 33, x = 34 \}$

4. (Proper states)
   a. (Well-formed and) Proper: The extra binding for $z$ isn’t a problem
   b. (Well-formed and) Proper: The value of $b$ is an array of length 1.
   c. (Well-formed but) Improper: The value of $b$ can’t be an integer.
d. (Well-formed but) Improper: We need a binding for $y$ even though we're multiplying it by zero. [So our semantics uses eager evaluation, not lazy evaluation.]

e. Ill-formed: We have two bindings for $i$.

f. (Well-formed and) Proper but causes a runtime error, since $b$ has size 2.