EOPL Quiz 6

Alireza Habibzadeh `99109393`

1

Expression	Type Variable
`successor`	t_s
`let successor = proc(n) -(n, -1) in let x = 5 in (successor x)`	t_0
`proc(n) -(n, -1)`	t_1
`let x = 5 in (successor x)`	t_2
`n`	t_n
`-(n, -1)`	t_3
`x`	t_x
`(successor x)`	t_4

Expression	Equations
`let successor = proc(n) -(n, -1) in let x = 5 in (successor x)`	$t_s = t_1$
	$t_0 = t_2$
`proc(n) -(n, -1)`	$t_1 = t_n \to t_3$
`let x = 5 in (successor x)`	$t_x = int$
	$t_2 = t_4$
`-(n, -1)`	$t_n = int$
	$t_3 = int$
`(successor x)`	$t_x \to t_4 = t_1$

Equations	Substitution
$t_s = t_1$	
$t_0 = t_2$	
$t_1 = t_n \to t_3$	
$t_x = int$	
$t_2 = t_4$	

Equations	Substitution
$t_n = int$	
$t_3 = int$	
$t_x \to t_4 = t_1$	

Equations	Substitution
$t_1 = t_n \to t_3$	$t_0 = t_2$
$t_x = int$	$t_s = t_1$
$t_2 = t_4$	
$t_n = int$	
$t_3 = int$	
$t_x \to t_4 = t_1$	

Equations	Substitution
	$t_0 = t_2$
$t_2 = t_4$	$t_s = t_n \to t_3$
$t_n = int$	$t_1 = t_n \to t_3$
$t_3 = int$	$t_x = int$
$t_x \to t_4 = t_1$	

Equations	Substitution
	$t_0 = t_2$
$t_2 = t_4$	$t_s = int \to int$
	$t_1 = int \to int$
	$t_x = int$
$t_x \to t_4 = t_1$	$t_n = int$
	$t_3 = int$

$$(t_x \to t_4) = t_1 = (int \to int)$$

پس

$$t_x = int, t_4 = int$$

Equations	Substitution
	$t_0 = t_2$
$t_2 = t_4$	$t_s = int \to int$
	$t_1 = int \to int$

Equations	Substitution
	$t_x = int$
$t_x = int$	$t_n = int$
$t_4 = int$	$t_3 = int$

Equations	Substitution
	$t_0 = t_4$
	$t_s = int \to int$
	$t_1 = int \to int$
	$t_x = int$
$t_x = int$	$t_n = int$
$t_4 = int$	$t_3 = int$
	$t_2 = t_4$

Equations	Substitution
	$t_0 = int$
	$t_s = int \to int$
	$t_1 = int \to int$
	$t_x = int$
	$t_n = int$
	$t_3 = int$
	$t_2 = int$
	$t_x = int$
	$t_4 = int$

عبارت از نظر type درست است و type خروجی هم int است.

2

الف

دست‌نویس در انتهای فایل :)

ب

تغییر تابع `type-of` دو تایپ جدید به `cases` اضافه می‌شوند که `pair-exp` و `unpair-exp` هستند. اگر `pair-exp` بود که به سادگی جنس عبارت را به صورت بازگشتی یک pair از جنس (`type-of`) عبارات چپ و راست آن خروجی می‌دهیم.

اگر `unpair-exp` بود باید ابتدا عبارات چپ و راست `p-exp` را بیرون بکشیم (و اگر نمی‌شد و از جنس pair نبود خطا دهیم) سپس `tenv` را با type متغییرهای جدید گسترش دهیم. `((extend-tenv var2 ty2 (extend-tenv var1 ty1 tenv))` و در نهایت جنس body را در این environment جدید پیدا کنیم.

```
#lang eopl

(define type-of
  (λ (exp tenv)
    (cases expression exp
; .
; .
; .
      [pair-exp (exp1 exp2) (pair-type (type-of exp1 tenv) (type-of exp2 tenv))]
      [unpair-exp (var1 var2 p-exp body)
                  (cases type (type-of p-exp tenv)
                    [pair-type (ty1 ty2)
                               (type-of body (extend-tenv var2 ty2 (extend-tenv var1 ty1 tenv)))]
                    [else (eopl:error 'type-of "Not a pair: ~s" exp1)])])))
```

1

99109393 علیرضا حبیب زاده

همان طور که از اسمش هم پیداست pair-exp یک چیزی از جنس pairof ty1 x ty2 زوجی می‌دهد :

$$\frac{\begin{array}{c}(\text{type-of exp1 tenv}) = ty1 \\ (\text{type-of exp2 tenv}) = ty2\end{array}}{(\text{type-of (pair-exp exp1 exp2) tenv}) = \text{pairof } ty1 \times ty2}$$

unpair-exp پس از اینکه کارش با bound کردن متغیرها تمام شد body را ارزیابی می‌کند پس نوع خروجی‌اش هم جنس نوع body است .

$$\frac{\begin{array}{c}(\text{type-of } e_{pair} \text{ tenv}) = (\text{pairof } ty1 \ ty2) \\ (\text{type-of } e_{body} \ [var1 = ty1][var2 = ty2]\text{tenv}) = t_{body}\end{array}}{(\text{type-of (unpair-exp } var1 \ var2 \ e_{pair} \ e_{body}) \text{ tenv}) = t_{body}}$$