Universidad del Valle

Segundo examen parcial
Fundamentos de lenguajes de programación
Duración: 2 horas

Carlos Andres Delgado S, Ing [*]

05 de Diciembre 2015

Nombre:_____
Código:_____

1. **Paso de Parámetros por Referencia [8 pts.]**

   Cuál es el resultado de evaluar la expresión:

```
let a = 1
    b = 2
    c = 3
    p = proc (x, y, z, f)
            begin
                set x = -(y, x);
                set y = (f z x);
                x
            end
    q = proc (m, n)
            begin
                set n = *(m, 5);
                set m = +(m, 6);
                +(m, n)
            end
in
   begin
       set a = (p c b a q);
       +(a, c)
   end
```

   Si el paso de parámetros es *por referencia*? Dibuje el ambiente en el cual se evalua la expresión.

2. **Chequeo de Tipos [10 pts.]**

   Para las siguientes expresiones tipadas, determine su tipo de acuerdo a las reglas de tipamiento para cada variante.

```
let fun1 = proc (int * (int → bool) → (int → bool) g,
                 bool t, bool s, int x, int → bool h)
               if and(t, s) then (g x h)
               else (g *(x, 2) h)
    fun3 = proc (int w)
               proc (int a)
                   >(w, a)
in
   let fun2 = proc (int x, int → bool q)
                  if >(x,3) then proc (int z)
                                     (q z)
                             else (fun3 x)
       fun4 = proc (int k)
                  <(+(k, 5), 9)
   in
      (fun1 fun2 m true j fun4)
```

   En un ambiente de tipos inicial $env_0 = [j \rightarrow int, m \rightarrow bool]$.

3. **Inferencia de Tipos [16 pts.]**

   En clase se describió el proceso formal de inferencia del tipo de una expresión y su implementación. En este ejercicio se pretende comprobar su asimilación del proceso (no de la implementación).

   Para la expresión a continuación y las variables de tipo introducidas describa las ecuaciones de tipo generadas entre ellas. Acto seguido resuelva el sistema para encontrar el tipo de la expresión:

```
let fun1 = proc (j, k, o)
               if o then
                   proc (y)
                       if k then (j y)
                       else (j *(y, 2))
               else proc (w)
                       *(w, 3)
```

[*] carlos.andres.delgado@correounivalle.edu.co

$fun_2 = \textbf{proc}\ (s,t)$
$\quad and(s,\ t)$
$r = false$

in

let $fun_3 = \textbf{proc}\ (a)$
$\quad *(a,\ 2)$

in

$\big((fun_1\ fun_3\ (fun_2\ true\ r)\ true)\ 4\big)$

es: _____

Expresión o Variable ligada	Variable de tipo
fun_1	t_{fun_1}
fun_2	t_{fun_2}
fun_3	t_{fun_3}
r	t_r
j	t_j
k	t_k
o	t_o
s	t_s
t	t_t
a	t_a
y	t_y
w	t_w
if o **then** **proc** (y) **if** k **then** $(j\ y)$ ...	t_1
proc (y) **if** k **then** $(j\ y)$ **else** $(j\ *(y,2))$	t_2
if k **then** $(j\ y)$ **else** $(j\ *(y,2))$	t_3
proc $(w)\ *(w,\ 3)$	t_4
$*(w,\ 3)$	t_5
$(j\ y)$	t_6
$(j\ *(y,2))$	t_7
$*(y,2)$	t_8
$and(s,\ t)$	t_9
proc $(a)\ *(a,2)$	t_{10}
$*(a,2)$	t_{11}
$(fun_2\ true\ r)$	t_{12}
$(fun_1\ fun_3\ (fun_2\ true\ r)\ true)$	t_{13}
$\big((fun_1\ fun_3\ (fun_2\ true\ r)\ true)\ 4\big)$	t_{14}
let fun_1 ...	t_{15}
let fun_3 ...	t_{16}

Expresión	Ecuacion(es) de tipo asociadas
if o **then** **proc** (y) **if** k **then** $(j\ y)$ ...	
proc (y) **if** k **then** $(j\ y)$ **else** $(j\ *(y,2))$	
$if\ k\ then\ (j\ y)$ $else\ (j\ *(y,2))$	
$(j\ y)$	
$(j\ *(y,2))$	
$*(y,2)$	
proc $(w)\ *(w,\ 3)$	
$*(w,\ 3)$	
$and(s,t)$	
proc $(a)\ *(a,\ 2)$	
$*(a,\ 2)$	
$(fun_2\ true\ r)$	
$(fun_1\ fun_3\ (fun_2\ true\ r)$ $true)$	
$\big((fun_1\ fun_3\ (fun_2\ true\ r)$ $true)\ 4\big)$	

4. **Claridad Operativa POO [16 pts.]**

Considere el siguiente programa en nuestro lenguaje OO:

```
class c1 extends object
    field a
    field b
    method initialize () 0
    method setup (k, l)
            begin
            set a=+(k,3);
            set b=l;
            8
            end
    method m1 (n) send self m2 (+(a,n))
    method m2 (n) *(n, -(a,b))
    method m4 (x, y) send self m1 (-(x,y))

class c2 extends c1
    field b
    field c
    method setup (k, l)
            begin
            set b=k;
            set c=-(k, l);
            super setup(+(b,k), -(1,l));
            send self m3(k)
            end
    method m2 (n) super m2(+(n, c))
    method m3 (n) send self m1(*(n,2))
    method m4 (n, m) +(m, super m4(n, m))

class c3 extends c2
    method m2 (n) super m2(n)
    method m4 (n,m) *(+(n,m),b)

let p=proc (o, r, q)
          let r1 = send o setup(r, q)
              in let r2 = send o m4(q, r)
                     r3 = send o m1(r)
                     in +(r1, +(r2,r3 ))
  o1 = new c1()
  o2 = new c2()
  o3 = new c3()
  in let x= (p o1 5 2)
         y= (p o2 4 1)
         z= (p o3 3 0)
         in send o2 m4(x, +(y,z))
```

a) **[10 pts.]** Complete en la siguiente tabla, los valores asociados a las variables indicadas en cada uno de los momentos de evaluación se nalados:

Variable	Valor	Al evaluar la expresión
r_1		+(r_1, +(r_2,r_3)) por efecto de la aplicación (p o_1 5 2)
r_2		+(r_1, +(r_2,r_3)) por efecto de la aplicación (p o_1 5 2)
r_3		+(r_1, +(r_2,r_3)) por efecto de la aplicación (p o_1 5 2)
x		send o_2 m4(x, +(y,z)) del cuerpo del let más interno
r_1		+(r_1, +(r_2,r_3)) por efecto de la aplicación (p o_2 4 1)
r_2		+(r_1, +(r_2,r_3)) por efecto de la aplicación (p o_2 4 1)
r_3		+(r_1, +(r_2,r_3)) por efecto de la aplicación (p o_2 4 1)
y		send o_2 m4(x, +(y,z)) del cuerpo del let más interno
r_1		+(r_1, +(r_2,r_3)) por efecto de la aplicación (p o_3 3 0)
r_2		+(r_1, +(r_2,r_3)) por efecto de la aplicación (p o_3 3 0)
r_3		+(r_1, +(r_2,r_3)) por efecto de la aplicación (p o_3 3 0)
z		send o_2 m4(x, +(y,z)) del cuerpo del let más interno

b) **[6 pts.]** Dibuje el ambiente en el que se evalua el cuerpo del método m4 en el proceso de evaluación de la expresión send o_2 m4(x, +(y,z)) del cuerpo del let más interno. Cuál es el resultado de evaluar esa expresión?