Cálculo de objetos

Ejercicio 13

Decir si los siguientes pares de términos definen al mismo objeto o no. Justificar.

1. $o_1 \equiv [\text{arg} = \varsigma(\text{x})\text{x.arg}, \text{val} = \varsigma(\text{x})\text{x.arg}]$ y $o_2 \equiv [\text{val} = \varsigma(\text{z})\text{z.arg}, \text{arg} = \varsigma(\text{v})\text{v.arg}]$
   - Definen el mismo objeto: responden los mismos mensajes y los métodos asociados son los iguales (son α-equivalentes).
2. $o_3 \equiv [\text{arg} = \varsigma(\text{x})\text{x.arg}, \text{val} = \varsigma(\text{x})\text{x.arg}]$ y $o_4 \equiv [\text{foo} = \varsigma(\text{z})\text{z.arg}, \text{arg} = \varsigma(\text{v})\text{v.arg}]$
   - No definen el mismo objeto: no responden los mismos mensajes.

Ejercicio 14

Considerar o $\equiv$ [arg = ς(x)x, val = ς(x)x.arg]. Derivar utilizando las reglas de la semántica operacional las reducciones para las siguientes expresiones:

1. o.val

$$\dfrac{\dfrac{}{\text{o} \to \text{o}}[\text{obj}] \quad \dfrac{}{\text{x}\{\text{o/x}\} = \text{o} \to \text{o}}[\text{obj}]}{\dfrac{}{\text{o} \to \text{o}}[\text{obj}] \quad \dfrac{}{\text{x.arg}\{\text{o/x}\} = \text{o.arg} \to \text{o}}[\text{sel}]}[\text{sel}]}{\text{o.val} \to \text{o}}$$

2. o.val.arg

$$\dfrac{\dfrac{\dfrac{}{\text{o} \to \text{o}}[\text{obj}] \quad \dfrac{}{\text{x}\{\text{o/x}\} = \text{o} \to \text{o}}[\text{obj}]}{\dfrac{}{\text{o} \to \text{o}}[\text{obj}] \quad \dfrac{}{\text{x.arg}\{\text{o/x}\} = \text{o.arg} \to \text{o}}[\text{sel}]}[\text{sel}]}{\text{o.val} \to \text{o}} \quad \dfrac{\dfrac{}{\text{o} \to \text{o}}[\text{obj}] \quad \dfrac{}{\text{x}\{\text{o/x}\} = \text{o} \to \text{o}}[\text{obj}]}{\text{o.arg} \to \text{o}}[\text{sel}]}{\text{o.val.arg} \to \text{o}}[\text{sel}]$$

3. (o.arg $\Leftarrow \varsigma$(z)0).val

$$\dfrac{\dfrac{}{\text{o} \to [\text{arg} = \varsigma(\text{x})\text{x}, \text{val} = \varsigma(\text{x})\text{x.arg}]}[\text{obj}]}{\dfrac{}{(\text{o.arg} \Leftarrow \varsigma(\text{z})0) \to \text{o}' \quad \text{o}' \equiv [\text{arg} = \varsigma(\text{z})0, \text{val} = \varsigma(\text{x})\text{x.arg}]}[\text{upd}] \quad \dfrac{\dfrac{}{\text{o}' \to [\text{arg} = \varsigma(\text{z})0, \text{val} = \varsigma(\text{x})\text{x.arg}]}[\text{obj}] \quad \dfrac{}{0\{\text{o}'/\text{z}\} = 0 \to 0}[\text{obj}]}{\text{x.arg}\{\text{o}'/\text{x}\} = \text{o}'.\text{arg} \to 0}[\text{sel}]}{(\text{o.arg} \Leftarrow \varsigma(\text{z})0).\text{val} \to 0}[\text{sel}]$$

PREGUNTA: En 0 → 0 también se usa la regla obj?

Ejercicio 15

Sea o $\equiv$ [a = ς(x)(x.a $\Leftarrow \varsigma$(y)(y.a $\Leftarrow \varsigma$(z)[]))]. Mostrar cómo reduce o.a.a.

$$\dfrac{\dfrac{\dfrac{}{\text{o} \to [\text{a} = \varsigma(\text{x})(\text{x.a} \Leftarrow \varsigma(\text{y})(\text{y.a} \Leftarrow \varsigma(\text{z})[]))]}[\text{obj}]}{\text{o} \to [\text{a} = \varsigma(\text{x})(\text{x.a} \Leftarrow \varsigma(\text{y})(\text{y.a} \Leftarrow \varsigma(\text{z})[]))]}[\text{obj}] \quad \dfrac{}{(\text{x.a} \Leftarrow \varsigma(\text{y})(\text{y.a} \Leftarrow \varsigma(\text{z})[]))\{\text{o/x}\} = (\text{o.a} \Leftarrow \varsigma(\text{y})(\text{y.a} \Leftarrow \varsigma(\text{z})[])) \to \text{o}' \quad \text{o}' \equiv [\text{a} = \varsigma(\text{y})(\text{y.a} \Leftarrow \varsigma(\text{z})[])]}[\text{upd}]}{\text{o.a} \to \text{o}' \quad \text{o}' \equiv [\text{a} = \varsigma(\text{y})(\text{y.a} \Leftarrow \varsigma(\text{z})[])]}[\text{sel}] \quad (1)}{\text{o.a.a} \to [\text{a} = \varsigma(\text{z})[]]}[\text{sel}]$$

$$\dfrac{\dfrac{}{\text{o}' \to [\text{a} = \varsigma(\text{y})(\text{y.a} \Leftarrow \varsigma(\text{z})[])]}[\text{obj}]}{(\text{y.a} \Leftarrow \varsigma(\text{z})[])\{\text{o}'/\text{y}\} = (\text{o}'.\text{a} \Leftarrow \varsigma(\text{z})[]) \to \text{o}'' \quad \text{o}'' \equiv [\text{a} = \varsigma(\text{z})[]]}[\text{upd}]}{(1)}$$

Ejercicio 16

1. Definir true y false como objetos con los siguientes tres métodos: not, if, e ifnot. Notar que tanto if como ifnot deberán retornar una función binaria. Las operaciones deberían satisfacer las siguientes igualdades:

- true.not = false
- false.not = true
- true.if v1 v2 = v1
- true.ifnot v1 v2 = v2
- false.if v1 v2 = v2
- false.ifnot v1 v2 = v1

true ≡ [not = ς(t)[not = t, if = t.ifnot, ifnot = t.if], if = λ(b1)λ(b2)(b1), if = λ(b1)λ(b2)(b2)]

false ≡ true.not

```
PREGUNTA: vale pedirle true.if sin pasarle los parámetros (para obtener la función nada más)?
          se puede definir así: not = sigma(t)((not ← t).if ← t.ifnot).ifnot ← t.if?
```

2. Definir and y or como objetos que se comporten como funciones que esperen dos argumentos (para poder escribir expresiones como and(true)(false), etc).

and ≡ $\lambda(x)\lambda(y)x.if(y)(x)$ PREGUNTA: esto es correcto o hay que escribir [[λx.λy. x.if(y)(false)]]?

or ≡ $\lambda(x)\lambda(y)x.if(x)(y)$

Ejercicio 17

1. Definir el objeto origen que representa el origen de coordenadas en dos dimensiones. Este objeto provee tres operaciones: los observadores x e y y mv tal que origen.mv v w desplaza a origen v unidades a la derecha y w unidades hacia arriba.

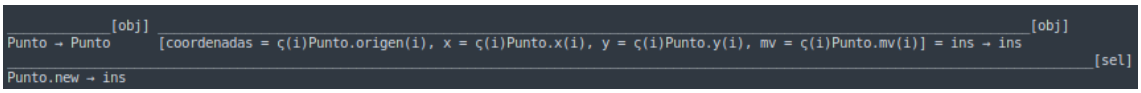
origen ≡ [x = 0, y = 0, mv = ς(o)λ(v)λ(w)(o.x ← o.x+v).y ← o.y+w]

2. Definir una clase Punto, cuyas instancias proveen las operaciones x, y y mv. Considerar que los puntos se crean con sus coordenadas en 0.

Punto = [
  new = ς(c)[coordenadas = ς(i)c.origen(i), x = ς(i)c.x(i), y = ς(i)c.y(i), mv = ς(i)c.mv(i)]
  origen = λ(i) origen,
  x = λ(i) i.coordenadas.x,
  y = λ(i) i.coordenadas.y,
  mv = λ(i)λ(v)λ(w) i.coordenadas ← i.coordenadas.mv(v)(w)
 ]

PREGUNTA: puedo hacer que el método de clase agarre tantos parámetros como quiera?

3. Mostrar como reduce Punto.new.



4. Definir la subclase PuntoColoreado, que permite construir instancias de puntos que tienen asociado un color. Además del método new, que crea puntos blancos, la clase debe contar con un método que cree puntos de un color pasado como parámetro.

PuntoColoreado = [
  new = ς(c)[coordenadas = ς(i)c.origen(i), x = ς(i)c.x(i), y = ς(i)c.y(i), mv = ς(i)c.mv(i), color = ς(i)c.color(i)]
  newConColor = ς(c)λ(color)c.new.color ← color
  origen = Punto.origen,
  x = Punto.x,
  y = Punto.y,
  mv = Punto.mv,
  color = λ(i)blanco
 ]

Ejercicio 18

1. Se desea definir a los números naturales como objetos de manera análoga al ejercicio 3. Estos objetos deben proveer las operaciones esCero y succ que permiten respectivamente testear si el receptor del mensaje es 0 o no, y obtener al sucesor del receptor. Además, todos los números distintos de 0 deben proveer la operación pred.

cero = [esCero = true, succ = ς(n)(n.esCero ← false).pred = n]

2. Definir iguales y menor como objetos que se comporten como funciones que esperen dos argumentos.

iguales = [val = ς(o)λ(x)or(and(o.arg.esCero)(x.esCero))((o.arg.esCero).if(false)((x.esCero).if(false)(o(o.arg.pred)(x.pred)))), arg =ς(o)o.arg]

Ejercicio 19

Definir en el cálculo de objetos, el objeto Vacio que representa el conjunto vacío y sabe responder los siguientes mensajes:

hayElementos(), que devuelve true si el conjunto contiene al menos un elemento.

agregar(x), que devuelve el objeto que agrega x al conjunto.

sacar(x), que devuelve el objeto que saca x del conjunto.

pertenece(x), que indica si x pertenece al conjunto.

Ejemplos:

Vacio.agregar(2).sacar(2).hayElementos() → true;

Vacio.agregar(2).agregar(3).sacar(2).pertenece(3) → true;

Se puede suponer que la operación == está definida para los elementos del conjunto.

vacio = [
hayElementos = false,
agregar = ς(o)λ(x)
(((o.hayElemento := true).pertenece := λ(y)or(y==x)(o.pertenece(y))).sacar := λ(y)(y==x).if(o)(o.sacar(y).agregar(x)),
pertenece=λ(x)false,
sacar=ς(o)λ(x)o
]