



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

### **Tarea 4**

#### INTEGRANTES

**Torres Valencia Kevin Jair - 318331818**  
**Aguilera Moreno Adrián - 421005200**  
**Rivera Silva Marco Antonio - 318183583**

#### PROFESORA

**Karla Ramírez Pulido**

#### AYUDANTES

**Alan Alexis Martínez López**  
**Manuel Ignacio Castillo López**  
**Alejandra Cervera Taboada**

#### ASIGNATURA

**Lenguajes de Programación**

23 de octubre de 2022

1. Currifica cada uno de los siguientes términos.

■  $\lambda abc.abc.$

**Solución:**  $\lambda a.\lambda b.\lambda c.abc$

■  $\lambda abc.\lambda cde.acbdce.$

**Solución:**  $\lambda a.\lambda b.\lambda c.\lambda c.\lambda d.\lambda e.acbdce$

■  $(\lambda d.(\lambda de.e)(\lambda fc.c))(\lambda ab.b).$

**Solución:**  $(\lambda d.(\lambda d.\lambda e.e)(\lambda f.\lambda c.c))(\lambda a.\lambda b.b)$

2. Aplica  $\alpha$ -conversiones en cada expresión para cambiar los términos de las variables de ligado.

a)  $\lambda_a.\lambda_b.((\lambda_a.b) (\lambda_b.a))$

**Solución:**

$$\lambda_a.\lambda_b.((\lambda_a.b) (\lambda_b.a)) \rightarrow_\alpha \lambda_x.\lambda_y.((\lambda_w.y) (\lambda_z.x))$$

b)  $\lambda a.(a(\lambda b.(\lambda a.a b)a))$

**Solución:**

$$\lambda a.(a(\lambda b.(\lambda a.a b)a)) \rightarrow_\alpha \lambda_x.(x(\lambda_y.(\lambda_w.w y)x))$$

c)  $\lambda x.(\lambda y.x \lambda y.(\lambda x.x y))$

**Solución:**

$$\lambda x.(\lambda y.x \lambda y.(\lambda x.x y)) \rightarrow_\alpha \lambda_a.(\lambda_b.a \lambda_c.(\lambda_e.e c))$$

3. Aplica  $\beta$ -reducciones a las siguientes expresiones para llegar a una Forma Normal, en caso de que no se pueda justifica. Además indica en cada paso el reducto y el redex.

$$l =_{def} \lambda a.a$$

$$K =_{def} \lambda a.\lambda b.a$$

$$S =_{def} \lambda a.\lambda b.\lambda a.ac(bc)$$

$$\Omega =_{def} (\lambda a.aa)(\lambda a.aa)$$

a)  $\lambda a.aK\Omega$

**Solución:**

$$\begin{aligned} \lambda a.aK\Omega &\equiv \lambda a.a(\lambda a.\lambda b.a)(\underbrace{(\lambda a.aa)(\lambda a.aa)}_{Redex}) \\ &\rightarrow_\beta \lambda a.a(\lambda a.a)(\underbrace{(\lambda a.aa)(\lambda a.aa)}_{Reducto}). \end{aligned}$$

Esta  $\lambda$ -función diverge, como se puede observar en la  $\beta$ -reducción anterior.

b)  $(\lambda a.a(ll))c$

**Solución:**

$$\begin{aligned}
 (\lambda a.a(ll))c &\equiv \lambda a.a(\underbrace{(\lambda a.a)(\lambda a.a)}_{\text{Redex}})c \\
 &\rightarrow_{\beta} \lambda a.a(\underbrace{\lambda a.a}_{\text{Reducto y Redex}})c \\
 &\rightarrow_{\beta} \underbrace{\lambda a.a}_{\text{Redex}} \underbrace{c}_{\text{Reducto}} \\
 &\rightarrow_{\beta} \underbrace{c}_{\text{Reducto}} .
 \end{aligned}$$

c)  $(\lambda d.\lambda e.(\lambda f.f(\lambda a.ad))e)b(\lambda c.\lambda b.cb)$

**Solución:**

$$\begin{aligned}
 (\lambda d.\lambda e.(\underbrace{\lambda f.f(\lambda a.ad)}_{\text{Redex}}))e)b(\lambda c.\lambda b.cb) &\rightarrow_{\beta} (\lambda d.\lambda e.(\underbrace{\lambda a.a d}_{\text{Redex}})e)b(\lambda c.\lambda b.cb) \\
 &\rightarrow_{\beta} (\lambda d.\lambda e.(\underbrace{\lambda a.a}_{\text{Reducto}} d)e)b(\lambda c.\lambda b.cb) \\
 &\rightarrow_{\beta} (\lambda d.\lambda e.(\underbrace{e}_{\text{Reducto}}))b(\lambda c.\lambda b.cb) \\
 &\rightarrow_{\beta} (\lambda d.(\underbrace{d}_{\text{Reducto}}))b(\lambda c.\lambda b.cb) \\
 &\rightarrow_{\beta} \underbrace{b}_{\text{Reducto}} ((\underbrace{\lambda c.\lambda b.cb}_{\text{Redex}})b) \\
 &\rightarrow_{\beta} b \underbrace{\lambda b.b}_{\text{Reducto}} .
 \end{aligned}$$

4. Realiza la representación de los booleanos en el cálculo  $\lambda$  según la representación de los Numerales de Church.

a) Define la función disyunción  $\leftrightarrow$  (equivalencia) sobre los booleanos.

Sabemos que a partir de las leyes de equivalencia de la lógica proposicional, tenemos que:

$$a \leftrightarrow b \equiv (a \rightarrow b) \wedge (b \rightarrow a) \qquad a \rightarrow b \equiv \neg a \vee b \qquad b \rightarrow a \equiv \neg b \vee a$$

Por lo que finalmente tenemos que:  $a \leftrightarrow b \equiv (\neg a \vee b) \wedge (\neg b \vee a)$ .

– Como en clase se vio que *and* queda definido de la siguiente forma:

$$\wedge =_{def} \lambda a. \lambda b. ((ab)F).$$

Definimos las siguientes funciones como:

$$\vee =_{def} \lambda a. \lambda b. ((aT)b).$$

$$\neg =_{def} \lambda a. aFT.$$

$$\rightarrow =_{def} (\lambda a. \lambda b. \vee (\neg a)b).$$

$$\leftrightarrow =_{def} (\lambda a. \lambda b. \wedge (\rightarrow ab)(\rightarrow ba))$$

b) Define la función *xor* (disyunción exclusiva) sobre los booleanos.

$$xor =_{def} \lambda a. \lambda b. (a (bFT) (bTF))$$

**5.** Observa la siguiente expresión en el lenguaje programación Racket.

```
(let ([sum (λ (n) (if (zero ? n) 0 (+ n (sum (sub1 n))))))
  (sum 5))
```

a) Ejecútala y explica el por qué del resultado.

b) Ejecútala modificándola usando Combinador de Punto Fijo Y y Combinador de Punto Fijo Z. Explica el resultado en ambos casos.