# Tarea 3

Rodrigo Rivera Paz    Godínez Galicia Luis Alberto    Valdés Galicia Alejandro

November 2015

## 1 Problema I

Haga el juicio de tipo para la función fibonacci y el predicado empty?

**l:list**

---

$\Gamma \vdash l : list$

---

$\Gamma \vdash (emptry?l) : boolean$

$\vdash n : number \quad \vdash 1 : number \quad \vdash n : number \quad \vdash 2 : number$

---

$\Gamma(fib(number \to number))\ \ \Gamma \vdash (-n1) : number\ \ \Gamma(fib(number \to number))\ \ \Gamma \vdash (-n2) : number$

---

$\Gamma \vdash n : number \qquad\qquad \Gamma \vdash n : number\ \ \Gamma \vdash 1 : number \qquad \Gamma \vdash (fib(-n1)) : number\ \ \Gamma \vdash (fib(-n2)) : number$

---

$\Gamma \vdash (zero?n) : boolean\ \ \Gamma \vdash 1 : number\ \ \Gamma \vdash (= n1) : boolean\ \ \Gamma \vdash 1 : number\ \ \Gamma \vdash (+(fib(-n1))(fib(-n2))) : number$

---

$\Gamma[n \leftarrow number] \vdash (cond[(zero?n)1][(= n1)1][(+(fib(-n1))(fib(-n2)))]) : number$

---

**bf**$\Gamma[fib \leftarrow number] \vdash (lambda(n : number)) : number\ \ (cond[(zero?n)1][(= n1)1][(+(fib(-n1))(fib(-n2)))]) : number \to number$

## 2 Problema II

**Considera el siguiente programa: ...**

**{[1]} = {(+ 1 (first(cons true false)))} ={(+ [2] [3])}=number y {[2]}={[3]}**
**{[2]} = number**
**{[3]} = {(first(cons true empty))}=number y {[4]}=list**
**{[4]} = {(cons true empty)}=list y {[true]}=list, {[empty]}=list**
**{[5]} = {true}=boolean ! La operación cons tiene dos argumentos y ambos deben de ser de tipo list, pero a la hora de hacer las restricciones vemos que true es de tipo boolean, lo cual debe mandar un error.**

## 3 Problema III

**Considera la siguiente expresióon con tipos: ...**

$\boldsymbol{\{[1]\} = \{f\}} \rightarrow \{[1]\}$
$\{[2]\} = \{x\} \rightarrow \{[2]\}$
$\{[3]\} = \{y\} \rightarrow \{[3]\}$
$\{[4]\} = \{(cons\ \ x\ \ (f(f\ \ y))\}\ \ con\ \ \{x\} = list\ \ y\ \ \{(f(f\ \ y))\} = list$
$\{[x]\} = number$
$\{[6]\} = \{(f(f\ \ y))\} = list\ \ con\ \ \{f\} = number\ \ y\ \ \{[7]\} = list$
$\{[7]\} = \{(f\ \ y)\} = list\ \ con\ \ \{f\} = number\ \ y\ \ \{y\} = list$
**Gracias a estas restricciones podemos inferir los tipos de los Cn's.**

## 4 Problema IV

**no cambian,por que sirven para revisar que los tipos sean correctos en la funcion**

## 5 Problema V

**-ventajas explisito**
**reutilizacion de codigo, mantenimiento**
**-implícito**
**cambiar su tipo sin hacer un cast**
**El polimorfismo puede hacerse con referencias de superclases abstract, super-clases normales e interfaces.**

**-desventajas -explícito mucho codigo el tipo de la referencia (clase abstracta, clase base o interface) limita los métodos que se pueden utilizar y las variables miembro a las que se pueden acceder.**
**-implisito no saber que tipo es**

## 6 Problema VI

**general**
**-ventajas**
**bibliotecas**
**resuelve diferente tipos de problemas**
**-desventajas**
**el algunos problemas son lentos**
**difícil de abstraer**

**específico**

**-ventajas**
**es fácil de aprender (algunos)**
**rápido**
**seguridad**
**-desventajas**
**solo sirve para una cosa**

**sql**

**fue creado para tener datos en tablas y regresa tablas**
**regresa la tabla nombres en una columna y su calificacion**

**SELECT nombre,calificación**
**FROM estudiante**

**css creado solo para la parte visual de una página**

```
/*CSS sobre selector de identificador*/\\
#header {
       background-color: #ff0000;
       color: #ffffff;
       font-size: 26px;

}
```

**html**
**para dar la solo estructura a la página web**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Ejemplo documento</title>
</head>
<body>
<p>Un párrafo de texto.</p>
</body>
</html>
```