Universidad Nacional Autónoma de México

Facultad de Ciencias

Lenguajes de Programación

Tarea 3

PROBLEMA I

FIBONACCI

$$\dfrac{\dfrac{\dfrac{n:number}{\Gamma\text{|-}n:number}\quad\dfrac{2:number}{\Gamma\text{|-}2:number}}{\Gamma\text{|-}(<=n\ \ 2):bool}\qquad\dfrac{\dfrac{\dfrac{\dfrac{n:number}{\Gamma\text{|-}n:number}\quad\dfrac{1:number}{\Gamma\text{|-}1:number}}{\Gamma\text{|-}(-n\ \ 1):number}}{\Gamma(fib[number\to number])\text{|-}(-n\ \ 1):number}}{\Gamma\text{|-}(fib(-n\ \ 1)):number}\qquad\dfrac{\dfrac{\dfrac{\dfrac{n:number}{\Gamma\text{|-}n:number}\quad\dfrac{2:number}{\Gamma\text{|-}2:number}}{\Gamma\text{|-}(-n\ \ 2):number}}{\Gamma(fib[number\to number])\text{|-}(-n\ \ 2):number}}{\Gamma\text{|-}(fib(-n\ \ 2)):number}}{\dfrac{\Gamma\text{|-}(+(fib(-n\ \ 1))(fib(-n\ \ 2))):number}{\Gamma[fib(number)]\text{|-}(cond[(<=n\ \ 2)][else(+(fib(-n\ \ 1))(fib(-n\ \ 2)))]):number\to number}}$$

$$\Gamma\text{|-}(rec(fibo:number(cond[(<=n\ \ 2)1][else(+(fib(-n\ \ 1))(fib(-n\ \ 2)))])))$$

EMPTY

$$\dfrac{\Gamma\text{|-}l:List}{\Gamma\text{|-}(Empty\ ?l):Bool}$$

PROBLEMA II

Tenemos la función suma (+ exp1 exp2) donde exp1 y exp2 son de tipo number entonces con (first (cons true empty)) debemos devolver number y con (cons true empty) devolvemos un nlist.

(1) (+ (2)1 (3)(first (4)(cons (5)true (6)empty)))
- $[\![1]\!]$ = (+(2)(3))
- $[\![2]\!]$ = $[\![1]\!]$ → number
- $[\![3]\!]$ = (f irst(4)(cons(5)true(6)empty))) → number
- $[\![4]\!]$ = (cons(5)true(6)empty)) → nlist
- $[\![5]\!]$ = $[\![true]\!]$ → bool
- $[\![6]\!]$ = $[\![empty]\!]$ → nlist

En $[\![5]\!]$ vemos que se está intentando construir una lista de números con un tipo bool lo cual causaría un error

PROBLEMA III

- $[\![1]\!]$ = [F] → number
- $[\![2]\!]$ = [x] → number
- $[\![3]\!]$ = [y] → number
- $[\![4]\!]$ = [cos x {f {f y}}] → list ya que [x]→number y [{f {f y}}]→list
- $[\![5]\!]$ = [x]→number

$[\![6]\!]$ = [{f {f y}}] →list
$[\![7]\!]$ = [{f y}]→nlist

Por lo tanto C_1 , C_3 , C_5 son tipo number y C_2 , C_4 , C_6 son nlist.

PROBLEMA IV

No importa si es perezoso o glotón Los juicios de tipo no cambian ya que el chequeo de tipos no se realiza en tiempo de ejecución, lo hace el compilador o interprete, entonces la evaluación no afecta el chequeo de tipos.

PROBLEMA V

Polimorfismo Explícito

Ventajas:

- Es más rápido en tiempo de compilación.
- Podemos usar el mismo código para varios propósitos.

Desventajas

- Escribir el código fuente es más tardado.
- una variable no puede ser utilizada para distintos tipos lo que significar más variables y por ende más espacio en memoria ocupado.

Polimorfismo Implícito

Ventajas

- No hay necesidad de construir un tipo en específico ya que siempre que usemos variables estaremos trabajando con tipos genéricos.
- El código fuente es mucho menos verboso.

Desventajas

- Los errores de tipo se dan en tiempo de ejecución.
- El programa es más propenso a errores de semántica.

PROBLEMA VI

DSL

Ventajas

- Fácil para resolver problemas específicos ya que es más "liviano" que un lenguaje de propósito general.
- Resuelven el problema sin redundancias.

Desventajas:

- Los problemas más grandes requieren el uso de un lenguaje de propósito general

Lenguajes de propósito general

Ventajas

- DSL dentro del mismo lenguaje de propósito general.
- Mayor control sobre el programa.

Desventajas:

- Aveces hay muchas cosas inecesarias.
- Aveces es menos eficaz para resolver problemas pequeños y específicos.