

Universidad Nacional Autónoma de México

Facultad de Ciencias

Lenguajes de Programación

Tarea 3

PROBLEMA I

FIBONACCI

$$\frac{\frac{\frac{n: \text{number}}{\Gamma|-n: \text{number}} \quad \frac{2: \text{number}}{\Gamma|-2: \text{number}}}{\Gamma|-(\leq n \ 2): \text{bool}} \quad \frac{\frac{\frac{n: \text{number}}{\Gamma|-n: \text{number}} \quad \frac{1: \text{number}}{\Gamma|-1: \text{number}}}{\Gamma|-(n \ 1): \text{number}} \quad \frac{\frac{\frac{n: \text{number}}{\Gamma|-n: \text{number}} \quad \frac{2: \text{number}}{\Gamma|-2: \text{number}}}{\Gamma|-(n \ 2): \text{number}} \quad \frac{\Gamma(\text{fib}[number \rightarrow number])|-(n \ 2): \text{number}}{\Gamma|-(\text{fib}(n \ 2)): \text{number}}}{\Gamma|-(+(\text{fib}(n \ 1))(\text{fib}(n \ 2))): \text{number}}}{\Gamma[\text{fib}(number)]|-(\text{cond}[(\leq n \ 2)][\text{else}(+(\text{fib}(n \ 1))(\text{fib}(n \ 2))])]: \text{number} \rightarrow \text{number}}}{\Gamma|-(\text{rec}(f \text{ ivo}: \text{number}(\text{cond}[(\leq n \ 2)1][\text{else}(+(\text{fib}(n \ 1))(\text{fib}(n \ 2))])]))}$$

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

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.

$\llbracket 1 \rrbracket = (+ \llbracket 2 \rrbracket 1 \llbracket 3 \rrbracket (\text{first } \llbracket 4 \rrbracket (\text{cons } \llbracket 5 \rrbracket \text{ true } \llbracket 6 \rrbracket \text{ empty})))$
 $\llbracket 1 \rrbracket = [(+ \llbracket 2 \rrbracket 1 \llbracket 3 \rrbracket (\text{first } \llbracket 4 \rrbracket (\text{cons } \llbracket 5 \rrbracket \text{ true } \llbracket 6 \rrbracket \text{ empty}))) \rightarrow \text{number}]$
 $\llbracket 2 \rrbracket = [1] \rightarrow \text{number}$
 $\llbracket 3 \rrbracket = [(\text{first } \llbracket 4 \rrbracket (\text{cons } \llbracket 5 \rrbracket \text{ true } \llbracket 6 \rrbracket \text{ empty})) \rightarrow \text{number}]$
 $\llbracket 4 \rrbracket = [(\text{cons } \llbracket 5 \rrbracket \text{ true } \llbracket 6 \rrbracket \text{ empty}) \rightarrow \text{nlist}]$
 $\llbracket 5 \rrbracket = [\text{true}] \rightarrow \text{bool}$
 $\llbracket 6 \rrbracket = [\text{empty}] \rightarrow \text{nlist}$

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

PROBLEMA III

$\llbracket 1 \rrbracket = [F] \rightarrow \text{number}$
 $\llbracket 2 \rrbracket = [x] \rightarrow \text{number}$
 $\llbracket 3 \rrbracket = [y] \rightarrow \text{number}$
 $\llbracket 4 \rrbracket = [\text{cons } x \{f \{f y\}\}] \rightarrow \text{list}$ ya que $[x] \rightarrow \text{number}$ y $\{f \{f y\}\} \rightarrow \text{list}$
 $\llbracket 5 \rrbracket = [x] \rightarrow \text{number}$

$\llbracket 6 \rrbracket = [\{f \{f y\}\}] \rightarrow \text{list}$
 $\llbracket 7 \rrbracket = [\{f y\}] \rightarrow \text{nlist}$

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:

- A veces hay muchas cosas innecesarias.
- A veces es menos eficaz para resolver problemas pequeños y específicos.