Prova 21 Lp - respostas

Fernanda Guimarães

1

Obs: no assembly and no typescript. no sql and no css/htlm.

Type/language	Dynamic	Static	Weak	Strong	Compiling	VM	Dyanmic scope	Static	Clojures
C/cpp		✓	✓		✓			✓	cpp
Go		✓		✓	✓			✓	✓
Bash	✓			✓	I	I	✓		✓
Java		✓		✓		✓		✓	✓
C#		✓	✓	✓		✓		✓	✓
Swift		✓		✓		✓		✓	✓
Ruby	✓			✓		✓		✓	✓
PHP	✓		✓			✓		✓	✓
JavaScript	✓		✓			✓		✓	✓
Python	✓			✓		✓		✓	✓

2

1. Não sei.
2. A macro elimina a necessidade do uso de registros de ativação, pois os mesmos são responsáveis por guardar os:
   - return address,
   - parameters,
   - return value.
   - link to caller's activation record.

Com a macro, tudo isso estaria disponível na própria chamada.

3

1. ```
   fun ins (e ,[]) = [e]
     | ins (e, (h::t)) = if h >= e then e::h::t else h::(ins(e, t));
   ```

2. ```
   fun inSortR [] = []
     | inSortR (h::t) = ins (h, inSort t);
   ```

3. ```
   fun inSortF L = foldr ins [] L;
   ```

4. o primeiro y, o segundo x (?) e z.

5. ```
   expr((lambda (name * lambda(name * expr))) * name)
   ```

1. ```
   val x = APP ((LAMBDA ("x", VAR "x")), LAMBDA ("y", APP (VAR "y", VAR "y")))
   ```
   ```

```
val x = (\x.x)(\y(y y))
```

2.
```
datatype expr = VAR of string | LAMBDA of string * expr | APP of expr * expr
fun freevar (VAR v) = [v]
  | freevar (APP (f,s)) = freevar(f) @ freevar(s)
  | freevar (LAMBDA (x, ex)) = diff x (freevar ex);
  (* where diff removes an element from a list *)
```