Ejercicio Semanal 2

Emiliano Galeana Araujo

Facultad de Ciencias, UNAM

Fecha de entrega: 06 de Septiembre de 2018

1. Descripción del programa

En este semañal vimos el lenguaje EAB. Importamos `Data.List` para operaciones con listas.

1.1. Lenguaje EAB

```
type Identifier = String

data Exp = V Identifier | I Int | B Bool
       | Add Exp Exp | Mul Exp Exp | Succ Exp | Pred Exp
       | Not Exp | And Exp Exp | Or Exp Exp
       | Lt Exp Exp | Gt Exp Exp | Eq Exp Exp
       | If Exp Exp Exp
       | Let Identifier Exp Exp deriving (Eq)
```

Un ejemplo de una EAB

```
(Let "x" (I 1) (V "x"))
```

Las siguientes son funciones recursivas para el tipo del lenguaje de EAB.

```
-- | frVars. Funcion que obtiene el conjunto de variables libres de una expresion.
frVars :: Exp → [Identifier]

-- | subst. Funcion que aplica la substitucion a la expression dada en caso de ser
--          posible.
```

```
subst :: Exp → Substitution → Exp

-- | alphaEq. Fencing que determina si dos expresiones son alfa-equivalentes.
alphaEq :: Exp → Exp → Bool
```

2. Entrada y ejecución

El programa es interpretado por `GCHI` de la siguiente forma

```
~:ghci EjerSem02.hs
```

2.1. EAB

Ya en el programa, los siguientes son ejemplos de ejecución del lenguaje de EAB..

```
*EjerSem02> frVars (Add (V "x") (I 5))
[``x'']

*EjerSem02> subst (Add (V "x") (I 5)) ("x", I 10)
Add(N[10], N[5])

*EjerSem02> alphaEq (Let "x" (I 1) (V "x")) (Let "y" (I 1) (V "y"))
True
```

3. Conclusiones

El ejercicio estuvo bien, solo la parte de la alpha-equivalencia me confundió un poco, creía que Add(V ẍ)(I 3) y Add(V ÿ)(I 3) eran alpha-equivalentes, pero después de estar leyendo un rato, entendí que solo lo son cuando las variables ligadas son distintas, y que esas variables no son ligadas, así que no deberían ser alpha-equivalentes, luego la escribí, pero si ponía números diferentes me daba `True` y eso no debería pasar, y ahí fue cuando hice ya los casos base, y como `Let` es el único que liga variables entonces es el único caso exhaustivo.

Referencias

[1] Archivero, curso de Lenguajes de Programación 2019-1