Lista de Exercícios Avaliativa 2

Danilo Pimentel de Carvalho Costa, 2016058077

Exercício 1

Escopo estático define valores fixos para variáveis no escopo. Analisando um programa antes de sua execução, é possível dizer o valor de uma variável para aquele escopo independentemente de redefinições subsequentes em escopos derivados deste. Escopo dinâmico, contrário ao escopo estático, calcula valores para variáveis à medida que estas são utilizadas pelo programa, não sendo possível precisar os valores para as variáveis do escopo analisando o programa antes de sua execução.

Exemplo 1

```
let
    val x = 5
in
    let
        val y = 2 * x
    in
        let
            val x = 3
        in
            x + y
        end
    end
end;
```

Se escopo estático for implementado, a expressão acima resulta em 13, pois o valor de y não é influenciado pela redefinição do valor de x no escopo criado pelo terceiro bloco let, sendo 10. Caso escopo dinâmico for implementado, a expressão resultaria em 9, pois o valor de y seria calculado somente quando usado dentro do terceiro bloco let, sendo 6.

Exemplo 2

```
val x = 5;
val y = 2 * x;
let
    val x = 3
in
    x + y
end;
```

A mesma explicação acima é válida para o código sem a definição dos dois primeiros blocos let, definindo as variáveis x e y no escopo global. Caso escopo estático for implementado, a expressão resulta em 13, pois o valor de y dependerá do valor de x definido no escopo global. Caso escopo dinâmico for implementado, a expressão resulta em 9, pois o valor de y utilizará o valor 3 para x, definido no escopo criado pelo bloco let.

Exercício 4

Uma closure é composta pelo nome da função, nome do argumento da função, corpo da função, e estado do ambiente no qual a função foi declarada. O nome da função é necessário para que outras expressões possam invocá-la. O nome do argumento da função é necessário para que seu valor possa ser utilizado no corpo da função. O corpo da função é necessário para definir as operações feitas pela função. O estado do ambiente da função é necessário para guardar os valores das variáveis definidas no escopo em que a função foi criada, possibilitando seu uso posteriormente na invocação da função sem que redefinições de tais variáveis afetem o resultado da função.

Exemplo 1

```
val x = 3;
fun f y = x * y;
```

Exemplo 2

```
val x = 3;
val z = 2;
fun f _ = x * z;
```