Lenguajes de Programación 2020-1
Facultad de Ciencias UNAM
Ejercicio Semanal 12

Sandra del Mar Soto Corderi
Edgar Quiroz Castañeda

November 13, 2019

1. Evalúe el resultado de la siguiente expresión

   ```
   (letcc k in 10 * (continue k 3)) + 2
   ```

   En en caso donde las expresiones antes de evaluar ya son valores y no forman parte de una expresión `continue`, se saltará directamente a la siguiente evaluación.

$$
\begin{aligned}
& \square \succ \texttt{(letcc[Nat](k.10 * continue(k, 3))) + 2} \\
\to_{\mathcal{K}} \; & \texttt{sum(\_ + 2)} \succ \texttt{letcc[Nat](k.10 * continue(k, 3))} \\
\to_{\mathcal{K}} \; & \texttt{sum(\_ + 2)} \succ \texttt{(10 * continue(k, 3))[k := cont(sum(\_ + 2))]} \\
\to_{\beta} \; & \texttt{sum(\_ + 2)} \succ \texttt{10 * continue(cont(sum(\_ + 2)), 3)} \\
\to_{\mathcal{K}}^{\star} \; & \texttt{mul(10, \_),sum(\_ + 2)} \succ \texttt{continue(cont(sum(\_ + 2)), 3)} \\
\to_{\mathcal{K}} \; & \texttt{continue(\_, 3), mul(10, \_),sum(\_ + 2)} \succ \texttt{cont(sum(\_ + 2))} \\
\to_{\mathcal{K}} \; & \texttt{continue(\_, 3), mul(10, \_),sum(\_ + 2)} \prec \texttt{cont(sum(\_ + 2))} \\
\to_{\mathcal{K}} \; & \texttt{continue(cont(sum(\_ + 2)), \_), mul(10, \_),sum(\_ + 2)} \succ 3 \\
\to_{\mathcal{K}} \; & \texttt{continue(cont(sum(\_ + 2)), \_), mul(10, \_),sum(\_ + 2)} \prec 3 \\
\to_{\mathcal{K}} \; & \texttt{sum(\_ + 2)} \prec 3 \\
\to_{\mathcal{K}}^{\star} \; & \square \prec 5
\end{aligned}
$$

2. Transforme la siguiente función a CPS

```haskell
-- función filter
filter _ [] = []
filter f (x:xs)
    | f x = x:xs'
    | otherwise xs'
  where xs' = filter f xs
```

   Hay que agregar un acumulador de aplicaciones de funciones así como alterar las llamadas recursivas para que siempre sean lo último que sea llamado.

```haskell
-- función filter en CPS
cpsfilter _ [] k = k []
cpsfilter f (x:xs) k
    | f x k = cpsfilter f xs (\v -> k (x:v))
    | otherwise = cpsfilter f xs k
```

   y utilice su definición para calcluar `cpsfilter cpseven [0..10]` paso a paso.