Funções Anônimas

São simplesmente funções sem nome, também conhecidas por "Abstrações Lambda" (Lambda Abstractions). O lambda (λ), nesse caso, é representado por "\". No exemplo `(\x -> x + 1) 8`, a saída é `8+1` que é igual a `9`, sendo `8` o parâmetro da função. Esse tipo de função é muito útil quando precisamos usar `map`, `foldr` ou `foldl`, como no exemplo abaixo.

```
strEx = "14 JAN;Amazon;40.32;23 JAN;Uber;14.84;" :: String
```

Opção 1 - Utilizando padrões:

```
clear' :: Char -> Char
clear' x | x == ';' = ' '
         | otherwise = x

map clear' strEx
```

Opção 2 - Utilizando case:

```
clear :: Char -> Char
clear x = case x of
             ';' -> ' '
             _   -> x

map clear strEx
```

Opção 3 - Utilizando uma função anônima:

```
map (\x -> case x of ';' -> ' '; _ -> x) strEx
```

Um exemplo de duas funções semelhantes:

```
menorQue20 :: Int -> Bool
menorQue20 x = x < 20

(\x -> x < 20)
```