Lista de exc 2

Vincenzo Alberice

32135661


```
{- 1
onde:: Eq a => a -> [a] -> Int
onde _ [] = error "Lista Vazia"
onde n (a:x)
  | n == a = 1
  | otherwise = 1 + onde n x

main = do
  let resultado = onde 3 [1,5,7,2,4,3,9,8]
  print resultado
-}

{- 2
ateh:: Eq a => a -> [a] -> [a]
ateh _ [] = error "Lista Vazia"
ateh elt (a:x)
  | elt == a = [a]
  | otherwise = [a] ++ ateh elt x

main = do
  let resultado = ateh 2 [1,5,7,2,4,3,9,8]
  print resultado
-}
```

```
{- 3
apos:: Eq a => a -> [a] -> [a]
apos _ [] = error "Lista Vazia"
apos elt (a:x)
  | elt == a = x
  | otherwise = apos elt x

main = do
  let resultado = apos 2 [1,5,7,2,4,3,9,8]
  print resultado
-}

{- 4
gera_n 0 = error "Valor invalido"
gera_n n = [1..n]

main = do
  let resultado = gera_n 5
  print resultado
-}

{- 5
gera_m_mult 0 0 = error "Valor invalido"
gera_m_mult n m = [n, n+n..m]

main = do
  let resultado = gera_m_mult 2 10
  print resultado
-}
```

```
{- 6
split [] = [[],[]]
split x = [splitAux1 (div (length x) 2) x, splitAux2 (div (length x) 2) x]

splitAux1:: Eq a => Int -> [a] -> [a]
splitAux1 _ [] = []
splitAux1 meio (a:x)
  | meio == length x = [a]
  | otherwise = [a] ++ splitAux1 meio x

splitAux2:: Eq a => Int -> [a] -> [a]
splitAux2 _ [] = []
splitAux2 meio (a:x)
  | meio == length x = x
  | otherwise = splitAux2 meio x

main = do
  let resultado = split [1,2,3,4,5,6,7]
  print resultado
-}

{- 7
mtam [] [] = True
mtam (a:x) [] = False
mtam [] (a:x) = False
mtam (a:x) (b:y) = mtam x y

main = do
  let resultado = mtam [1,2,3,4,5] [5,4,3,1]
  print resultado
-}
```

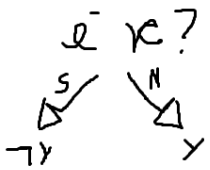
```
{- 8
tri [] = []
tri (a:x) = [a] ++ [a] ++ [a] ++ tri x

main = do
  let resultado = tri [1,2,3]
  print resultado
-}
{- 9
subs:: Eq a => a -> a -> [a] -> [a]
subs _ _ [] = []
subs a b (e:x)
  | e == a = [b] ++ subs a b x
  | otherwise = [e] ++ subs a b x

main = do
  let resultado = subs 1 5 [1, 3, 5, 4, 3, 1, 2, 1, 7, 1]
  print resultado
-}
```

10 -

x	y	xor
T	T	F
T	F	T
F	T	T
F	F	F



$\lambda xy.x(\neg y)(y)$

$(\lambda xy.x(\neg y)(y))\ T\ F$

$T(\neg F)F \rightarrow T\ T\ F$

$(\lambda xy.x)\ T\ F = T$

$(\lambda xy.x(\neg y)(y))\ F\ T$

$F(\neg T)T \rightarrow T\ F\ T$

$(\lambda xy.y)\ F\ T = T$