Lista de exercícios para serem feitos durante a aula do dia 19/03/2020

[Q1] Crie uma função que, dada uma string x e um inteiro y, adiciona y espaços entre os caracteres da string.

```
addEspOnStr "Camila" 2
"C a m i l a"
```

[Q2] Dados dois inteiros, onde o primeiro é um número positivo e o segundo é apenas um dígito, implemente uma função **quantidade :: Int -> Int -> Int** retorne quantas vezes o dígito ocorre no range de 0 até o número dado.

```
quantidade 20 1
12
quantidade 10 5
1
```

[Q3] Implemente a função **limpaUm :: [Int] -> [Int]** que, dada uma lista com vários inteiros, remove "1" de cada um desses inteiros e devolve a lista.

```
limpaUm [123, 444, 1243, 8742, 8273, 9913]
[23, 444, 243, 8742, 8273, 993]
limpaUm [22, 49, 323]
[22, 49, 323]
limpaUm [1]
[]
```

[Q4] Faça uma função em Haskell **isPalindromo :: String -> Bool** que dada uma String retorna se a mesma é palíndromo ou não. Considere que não haverá nenhuma letra maiúscula na string.

```
isPalindromo “asd”
False
isPalindromo “ana”
True
isPalindromo “Ana”
True
```

[Q5] Faça uma função em Haskell **isPalindromo :: String -> Bool** que dada uma String retorna se a mesma é palíndromo ou não. Considere que não haverá nenhuma letra maiúscula na string.

```
isPalindromo “asd”
False
isPalindromo “ana”
True
```

[Q6] Faça uma função em Haskell **`btoi :: String -> Int`** que, dada uma string representando um número binário, retorna um inteiro na base 10 dessa string.

```
btoi “0011”
3
btoi “1100”
12
```

[Q7] Faça uma função **`metade :: [Int] -> ([Int], [Int])`** que separa uma lista de inteiros em duas partes retornando-as em uma tupla. Caso essa string possua tamanho ímpar, a segunda lista da tupla retornada terá um elemento a mais que a primeira.

Exemplo:

```
metade [1,2,3,4]
([1,2],[3,4])
metade [1,2,3]
([1],[2,3])
```

[Q8] A string **"aaaaa"** repete um único caractere, o **'a'**, 5 vezes. Crie uma função **`isReplica :: String -> Int -> Char -> Bool`** que recebe uma string, um inteiro **x** e um char verifica se essa string é a repetição do char **x** vezes.

```
isReplica "ee" 2 'e'
True
isReplica "uruu" 3 'u'
False
isReplica "xxx" 3 'y'
False
```

[Q9] Crie uma função **`decEnigma :: String -> [(Char, Char)] -> String`** que decifra uma string da linguagem A para a linguagem B. Ela recebe a string que precisa ser decifrada e uma lista de tuplas contendo os dois alfabetos. Os primeiros caracteres da tupla, representam o alfabeto de A e, os segundos, de B.

```
decEnigma "usr" [('u','j'), ('s','o'), ('r','b')]
"job"
decEnigma "msyc" [('m','e'), ('s','i'), ('y','t'), ('c','a')]
"eita"
decEnigma "qloz" [('q','h'), ('l','u'), ('o','g'), ('z','o')]
"hugo"
```

Data;Tipo;Compra;Valor;Status
14 JAN;Amazon;40.32;Aprovada;
15 JAN;Uber;14.84;Bloqueada;
25 JAN;Uber;34.24;Aprovada;
02 FEV;Spotify;8.50;Bloqueada;
06 FEV;Uber;6.94;Bloqueada;
05 MAR;Burger;29.90;Aprovada;
10 MAR;Burger;24.99;Aprovada;
15 MAR;UCI;19.00;Bloqueada;
08 ABR;Itunes;3.50;Bloqueada;
13 ABR;Picpay;20.00;Aprovada;

Considere a fatura anual de cartão acima, sendo representada pela String:
logCartao = "14 JAN;Amazon;40.32;Aprovada;15 JAN;Uber;14.84;Bloqueada;25 JAN;Uber;34.24;Aprovada;02 FEV;Spotify;8.50;Bloqueada;06 FEV;Uber;6.94;Bloqueada;05 MAR;Burger;29.90;Aprovada;10 MAR;Burger;24.99;Aprovada;15 MAR;UCI;19.00;Bloqueada;08 ABR;Itunes;3.50;Bloqueada;13 ABR;Picpay;20.00;Aprovada;"

[Q10] Crie uma função **`clearData :: String -> [(Int, String, String, Double, Bool)]`** que recebe o log de um cartão e separa cada linha em uma tupla contendo as informações daquela transação.

Novas questões poderão ser adicionadas até o dia 19/03