DESCRIÇÃO

A atividade consiste em resolver os exercícios abaixo utilizando o paradigma funcional, e pode ser feita **em dupla**. Sugere-se que todas as respostas estejam no mesmo arquivo, assim é possível aproveitar funções para resolver outros exercícios, se necessário.

1. Crie uma função para calcular a área de uma circunferência. Teste a função anterior utilizando `let`.
2. Crie uma função que receba três medidas a, b e c, correspondentes aos lados de um triângulo, e imprima o tipo do triângulo (escaleno, isósceles ou equilátero) ou NAOTRI quando aquelas medidas não formam um triângulo.
3. Utilizando recursividade e sem usar o operador `*`, crie a função `multiplica x y`.
4. E se você pudesse supor que `x` e `y` sempre serão naturais (não negativos)? Faça `multiplicaNaturais x y`, também recursiva, sem usar o operador `*`.
5. Crie as funções `menor` e `maior`, que devem receber três valores e indicar menor e maior valor, respectivamente. (**EXTRA**) Defina a função `maior` utilizando a função `menor`, ou o contrário.
6. Crie a função XOR (também conhecida como "*ou exclusivo*"). Esta função retorna `True` se duas expressões forem diferentes (uma `True` e a outra `False`).
7. Crie a função `clonaNumeros`, que deve receber uma lista e duplicar cada valor recebido nela. Ex: `[1,2,3]` deve retornar `[1,1,2,2,3,3]`.
8. Crie uma função que receba uma lista e some os dois primeiros valores da lista.
9. Crie uma função que receba um número e crie uma lista de 0 até o valor absoluto deste número. Ex: `-9` deve retornar `[0,1,2,3,4,5,6,7,8,9]`, `0` deve retornar `[0]` e etc.
10. Crie a função `parOuImpar`, que recebe uma lista de valores e retorna uma lista *booleana* com `True` quando o valor for par e `False` quando o valor for ímpar. Exemplo: `[1, 2, 3]` retorna `[False, True, False]`.
11. Crie a função `soPar`, que remove todos os números ímpares da lista.
12. Crie uma função `soMinusculas`, que recebe uma lista `[Char]`, ou `String`, e retorna somente as letras minúsculas.

13. Crie a função `substituiVogais :: [Char] -> [Char]`, que recebe uma palavra e substitui as vogais minúsculas por maiúsculas. Exemplos: `"oi"` retorna `"OI"`, `"Ricardo"` retorna `"RIcArdO"`.
14. Crie um comando que receba uma lista de `Strings` e acrescente a `String` `" Friboi"` a todas elas. Exemplo: entrada `["Joao", "Maria", "oi"]` → saída `["Joao Friboi", "Maria Friboi", "oi Friboi"]`.
15. Declare a função `pertence`, que deve receber um valor e uma lista como parâmetros. A função deve verificar se o valor pertence à lista.
16. Declare a função `filtraLista`, que deve eliminar valores repetidos da lista.
17. Declare uma função que retorne os *n* primeiros elementos de uma lista.
18. Utilizando compreensão de lista, crie uma lista contendo os valores múltiplos de 3 no intervalo entre 0 e 300, inclusive.

PESO DA AVALIAÇÃO

Notas de aula.

OBSERVAÇÕES

- Plágio = **ZERO** **(inclui cópia ou simples alteração de trabalho de colegas)**