Lista de Exercícios de Python usando paradigma Funcional

Nesta lista, vocês devem usar o paradigma funcional. Isto é: você deve usar apenas funções sem efeitos colaterais; iterações devem ser feitas apenas usando recursão. Não usem os métodos prontos de Python (a não ser quando explicitado)! Se quiserem, utilizem avaliação preguiçosa (através de geradores);

Lembrem-se: Funções, em programação funcional pura, seguem o padrão das funções matemáticas. Isso é, elas **não são sequências de comandos**, ou elas são uma expressão ou uma função definida por várias partes (cada parte pode ter apenas uma expressão). Ex:

```
def exemplo(x, y):
    if (x>0):
        if (y>0):
            return x*y
        else:
            return 0
    else:
        return 0
```

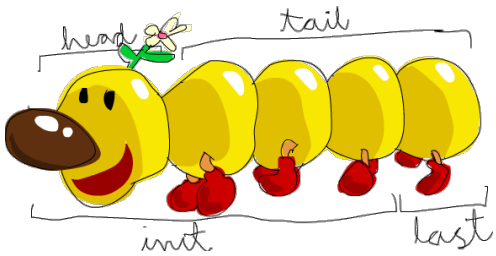
```
def exemplo(x, y):
    if (x>0) and (y>0):
        return x*y
    if (x>0) and not (y>0):
        return 0
    if not (x>0):
        return 0
```

Questão 1) Escreva uma função 'head' que retorna o primeiro elemento de uma lista

Questão 2) Escreva uma função 'tail' que retorna toda a lista, exceto o primeiro elemento

Questão 3) Escreva uma função 'init' que retorna toda a lista, exceto o último elemento

Questão 4) Escreva uma função 'last' que retorna o último elemento de uma lista



A partir daqui, pare de usar slices!

Questão 5) A sequência de Fibonacci é dada pela seguinte série: 0 1 1 2 3 5 8 13 ... Em termos matemáticos, a sequência de Fibonacci pode ser definida através da seguinte relação de recorrência:

$$f(n) = \begin{cases} 0, & n=0 \\ 1, & n=1 \\ f(n-1) + f(n-2) \end{cases}$$

Construa uma função para retornar o n-ésimo termo da sequência.

Questão 6) Faça uma função que concatena duas listas de forma recursiva. Utilize as funções head/tail para acessar os elementos. O comportamento deve ser o mesmo do operador + (listas). O operador + até pode ser usado, mas um dos operandos deve conter no máximo 1 elemento.

Questão 7) Escreva uma função que verifique se um elemento pertence a uma lista. Não usar o operador “in”;

Questão 8) Escreva uma função para realizar a união de duas listas. A função é similar à feita na Q6, mas elementos repetidos não são permitidos.

Questão 9) Defina uma função que dada uma lista de inteiros e um número n, retorne o total de elementos de valor superior a n.

Questão 10) Defina uma função que dada uma lista de inteiros e um número n, retorne outra lista contendo apenas de elementos de valor superior a n. Use a função feita na Q6.

Questão 11a) Escreva uma função que inverte o conteúdo de uma lista. Use apenas as funções da Q1 e a da Q6:

invertelista (“abcd”) = “dcba”.

Questão 12) Escreva uma função que receba uma palavra e gere seu palíndromo. Ex.:

geraPalindromo (“abcd”) = “abcddcba”.

Questão 13) Escreva uma função que retorne o tamanho (a quantidade de elementos) de uma lista. Não usar a função len para isso.

Questão 14) Escreva a função ehPrimo para verificar se um número dado é primo.

Questão 15) Defina a função strip que dadas duas listas, retira da segunda todos os elementos que ocorrem na primeira, em qualquer quantidade.

Questão 16) Defina a função consoantList que retorna verdadeiro se somente se todas as consoantes da segunda lista, incluindo repetições, ocorrem na primeira lista, na mesma ordem. Exemplos:

ordSublist ("sdd", "saude") -> False

ordSublist ("sdd", "saudade") -> True

Dica: use a função strip.

Questão 17) Fefina a função matches que recebe uma lista de palavras e uma sequência de consoantes e retorna uma lista de possíveis palavras representadas pelas consoantes. Use a função da Q14. Exemplos:

dic = ["arara","arreio","haskell", ,"vaca","vacuo","velho","vermelho","vicio"]

matches (dic, "rr") -> ["arara","arreio"]

matches (dic, "vc") -> ["vaca","vacuo","vicio"]

Questão 18) Faça uma função que, dado um número, retorna o menor número primo que é maior que o número. Ex: proximoPrimo(2) → 3

Questão 19) Faça a função primes, que retorna a lista de fatores primos de um número que ela recebe. Ex: primes(8)→ [2,2,2]

Questão 20) Defina a função primeFactors que fatora um número inteiro em uma lista de pares (fator,frequência). Exemplos:

primeFactors (8) -> [(2,3)]

primeFactors (42) -> [(2,1),(3,1),(7,1)]

Questão 21) Defina a função splitToken que recebe um valor e uma lista e retorna uma lista de listas utilizando o valor dado como marcador. Ex:

SplitToken (2, [1,2,3,4,2,5,67,8,9,0,3]) -> [[1],[3,4],[5,67,8,9,0,3]]

Questão 22) Defina a função joinToken que recebe um valor e uma lista de listas e retorna a concatenação das sublistas usando o primeiro parâmetro como separador.

Questão 23) Defina a função splitHalf que divide uma lista em duas, de tamanho iguais (ou com diferença de apenas um elemento no caso de uma lista de tamanho ímpar).

Questão 24) Uma tripla (x,y,z) de números inteiros positivos é chamada pitagórica se $x^2+y^2 = z^2$. Usando list comprehension, defina uma função pyths que mapeia um inteiro n a uma lista de todas as triplas pitagóricas componentes no intervalo [1..n]. Por exemplo:

pyths (5) -> [(3,4,5),(4,3,5)]

Questão 25) Um número inteiro positivo é perfeito se ele igual à soma de todos os seus fatores, excluindo o próprio número. Usando list comprehension, defina uma função perfects que retorna a lista de todos os números perfeitos de zero até um dado limite. Por exemplo:

perfects (500) -> [6,28,496]

Questão 26) O produto escalar de dois vetores v e w de tamanho n é dado pela soma dos produtos dos elementos correspondentes. Usando list comprehension, defina uma função que retorna o produto escalar de dois vetores representados por listas.

Questão 27) O problema das n rainhas consiste em posicionar em um tabuleiro de xadrez n×n, n rainhas de modo que cada rainha não ataque as demais. Uma rainha pode atacar qualquer outra que esteja na mesma linha, coluna, ou nas mesmas diagonais. Considere que a representação da solução será feita por meio de uma lista de pares (Linha, Coluna), de coordenadas das rainhas. Defina a função ataca que dada uma posição e uma lista de posições diz se a primeira posição ataca qualquer uma das posições da lista.

Questão 28) Implemente a função isPalindrome que verifica se uma string é palindroma ou não.

Questão 29) Implemente a função compress que elimina duplicadas consecutivas em uma lista.

Questão 30) Implemente a função pack que empacota os elementos duplicados consecutivos em sublistas.

Questão 31) Implemente a função encode que especifica o método de compressão de dados baseado no tamanho da sequência repetida. Neste método os elementos duplicados consecutivos são codificados como duplas (N,E), onde N é o número de duplicadas do elemento E. Ex:

encode "aaaabccaadeeee" -> [(4,’a’),(1,’b’),(2,’c’),(2,’a’),(1,’d’),(4,’e’)]

Questão 32) Implemente a função decode a qual, dada uma lista codificada como no exercício anterior, gera a lista original.