Introdução a Swift (I.adicional)

1. RUN LENGTH ENCODING

Run-length encoding (RLE) é um mecanismo de compressão, em que elementos iguais e consecutivos são substituídos por um único elemento e um valor contador que mostra seu número de repetições. Por exemplo, o texto original indicado a seguir (53 caracteres) pode ser representado com apenas 13 caracteres de forma comprimida:

"WWWWWWWWWWWWBWWWWWWWWWWWWBBBWWWWWWWWWWWWWWWWWWWWBWWWWWWWWWWWWB"

→

"12WB12W3B24WB"

A informação original pode ser perfeitamente reconstruída a partir da representação comprimida:

"AABCCCDEEEE" → "2AB3CD4E" → "AABCCCDEEEE"

É necessário:

- Definir o método ***comprimirRLE(texto: String) -> String*** que, dado o texto recebido como parâmetro, retorna a representação comprimida. Supomos que o texto a ser comprimido só contém letras e NÃO está vazio.

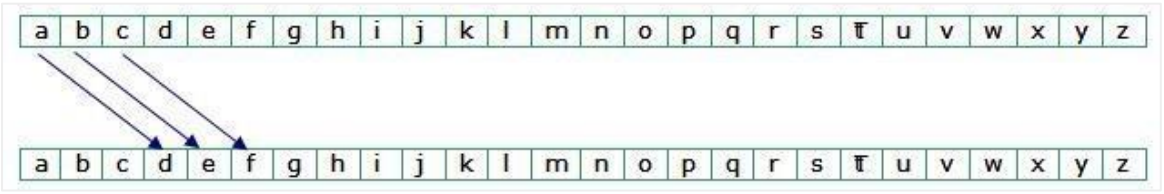
2. CRIPTOGRAFIA CÉSAR

Definir o método ***`criptografiaCesar(texto : String, deslocamento: Int) -> String`*** que deve analisar um texto e um número inteiro e codificar o texto utilizando o método de criptografia César. Vamos supor que o texto contém apenas letras minúsculas e sem acentos. Ou seja, o alfabeto terá 26 letras.

Criptografia César:

*Nesse método de criptografia, cada letra do texto é substituída por outra, que está **n** posições mais adiante no alfabeto. Consideramos que o alfabeto é circular, ou seja, depois da letra "z" vem a letra "a". Os espaços continuam iguais.*

*Então, por exemplo, se **n** é igual a 3, o "a" se transformaria em "d", o "b" em "e", o "c" em "f" e assim por diante.*



Exemplos de criptografia César:

✓ *Se o texto é **"casa"** e n = 3, o texto criptografado é **"fdvd"***

✓ *Se o texto é **"zorro"** e n = 10 o texto criptografado é **"jybby"***