Ejercicios de Listas

September 24, 2018

1 Problemas biólogicos con listas

```
In [1]: s = "GGCAGATTCCCCCTAGACCCGCCCGCACCATGGTCAGGCATGCCCCTCCTCATCGCTGGGCACAGCCCAGAGGGT ATAAA
        len (s)

Out[1]: 1247

In [2]: s.count (" ")

Out[2]: 16

In [5]: # funcion split para partir una cadena en listas
        s1 = "hola mundo bonito"
        ls1 = s1.split (" ")
        print (ls1)

['hola', 'mundo', 'bonito']


In [6]: s1.split ("o")

Out[6]: ['h', 'la mund', ' b', 'nit', '']

In [7]: # Vamos a partir la cadena de ADN con espacios
        print (s)

GGCAGATTCCCCCTAGACCCGCCCGCACCATGGTCAGGCATGCCCCTCCTCATCGCTGGGCACAGCCCAGAGGGT ATAAACAGTGCTGGAGGC


In [9]: lsSec = s.split(" ")
        len (lsSec)

Out[9]: 17

In [10]: print (lsSec [0])

GGCAGATTCCCCCTAGACCCGCCCGCACCATGGTCAGGCATGCCCCTCCTCATCGCTGGGCACAGCCCAGAGGGT
```

```
In [11]: print (lsSec [-1])

ATCCCAGCTGCTCCCAAATAAACTCCAGAAG


In [19]: s0 = lsSec[0]
         len (s0)

Out[19]: 75

In [24]: s0r = s0[:-40]
         s0r

Out[24]: 'GGCAGATTCCCCCTAGACCCGCCCGCACCATGGTC'

In [28]: # Vamos a recortar las secuencias 40 ultimos
         lsr = [] #
         for x in lsSec:
             y = x [:-40]
             lsr.append (y)

         lsr.pop()
         lsr

Out[28]: ['GGCAGATTCCCCCTAGACCCGCCCGCACCATGGTC',
          'ATAAACAGTGCTGGAGGCTGGCGGGGCAGGCCAGC',
          'ATGAGAGCCCTCACACTCCTCGCCCTATTGGCCCT',
          'CACCTCCCCTCAGGCCGCATTGCAGTGGGGGCTGA',
          'GCTGGCAGTCCCTTTGCAGTCTAACCACCTTGTTG',
          'GAGAGGAGGGAAGAGCAAGCTGCCCGAGACGCAGG',
          'CAGGCTCCCTTTCCTTTGCAGGTGCGAAGCCCAGC',
          'TGATGGGTTCCTGGACCCTCCCCTCTCACCCTGGT',
          'GCCATCAGGAAGGCCAGCCTGCTCCCCACCTGATC',
          'CACAGCCTTTGTGTCCAAGCAGGAGGGCAGCGAGG',
          'GTGAGAGAAAAGGCAGAGCTGGGCCAAGGCCCTGC',
          'GCCTCTCTGGGTTGTGGTGGGGGTACAGGCAGCCT',
          'AGGGATGGGCATTTTGCACGGGGGCTGATGCCACC',
          'CCTGGAGCCCAGGAGGGAGGTGTGTGAGCTCAATC',
          'GGCCTATCGGCGCTTCTACGGCCCGGTCTAGGGTG',
          'CTCCAGGCACCCTTCTTTCCTCTTCCCCTTGCCCT']

In [34]: # Funcion cambiar ADN a ARN
         def trascribir (cadenaADN):
             cadenaARN = ""
             for x in cadenaADN:
                 if x == "T":
                     cadenaARN += "U"
                 elif x == "t":
                     cadenaARN += "u"
```

```
            else:
                cadenaARN += x

        return cadenaARN
```

```
In [33]: lsARN = []
         for adn in lsr:
             arn = trascribir (adn)
             lsARN.append (arn)

         lsARN
```

```
Out[33]: ['GGCAGAUUCCCCCUAGACCCGCCCGCACCAUGGUC',
          'AUAAACAGUGCUGGAGGCUGGCGGGGCAGGCCAGC',
          'AUGAGAGCCCUCACACUCCUCGCCCUAUUGGCCCU',
          'CACCUCCCCUCAGGCCGCAUUGCAGUGGGGGCUGA',
          'GCUGGCAGUCCCUUUGCAGUCUAACCACCUUGUUG',
          'GAGAGGAGGGAAGAGCAAGCUGCCCGAGACGCAGG',
          'CAGGCUCCCUUUCCUUUGCAGGUGCGAAGCCCAGC',
          'UGAUGGGUUCCUGGACCCUCCCCUCUCACCCUGGU',
          'GCCAUCAGGAAGGCCAGCCUGCUCCCCACCUGAUC',
          'CACAGCCUUUGUGUCCAAGCAGGAGGGCAGCGAGG',
          'GUGAGAGAAAAGGCAGAGCUGGGCCAAGGCCCUGC',
          'GCCUCUCUGGGUUGUGGUGGGGGUACAGGCAGCCU',
          'AGGAUGGGCAUUUUGCACGGGGGCUGAUGCCACC',
          'CCUGGAGCCCAGGAGGGAGGUGUGUGAGCUCAAUC',
          'GGCCUAUCGGCGCUUCUACGGCCCGGUCUAGGGUG',
          'CUCCAGGCACCCUUCUUUCCUCUUCCCCUUGCCCU']
```

```
In [35]: def contarGC (cadena):
             conteo = 0
             for x in cadena:
                 if x == "G" or x=="C":
                     conteo += 1

             return conteo
```

```
In [36]: s0 = lsARN [0]
         s0
```

```
Out[36]: 'GGCAGAUUCCCCCUAGACCCGCCCGCACCAUGGUC'
```

```
In [37]: contarGC (s0)
```

```
Out[37]: 24
```

```
In [40]: lsGC = []
         for x in lsARN:
             n = contarGC (x)
```

```
                lsGC.append (n)

        lsGC

Out[40]: [24, 23, 21, 24, 19, 23, 22, 22, 23, 22, 22, 23, 22, 22, 24, 21]

In [41]: # Problema: buscar el menor de una lista:
         #           cual es? y en que posición?

In [42]: lsGC = [24, 23, 21, 24, 19, 23, 22, 22, 24, 21]

In [52]: def buscarMenor (lst):
             menor = lst [0]
             pos = 0

             for i,x in enumerate (lst): ## retorna indice y posicion
                 if x < menor:
                     menor = x
                     pos   = i

             return ([menor, pos])

In [54]: print (lsGC)
         lsResultado = buscarMenor (lsGC)

[24, 23, 21, 24, 19, 23, 22, 22, 24, 21]


In [55]: lsResultado

Out[55]: [19, 4]

In [58]: s= "hola mundo"
         s = s.replace ("hola", "bola")

In [62]: s0 = lsARN [0]
         s0

Out[62]: 'GGCAGAUUCCCCCUAGACCCGCCCGCACCAUGGUC'

In [67]: while "C" in s0:
             s0 = s0.replace ("C","")

         s0

Out[67]: 'GGAGAUUUAGAGGAAUGGU'
```