Práctica de Lista, Diccionarios y Archivos

Prof. Luis Garreta
Informática
Biología
Pontificia Universidad Javeriana

25 de octubre de 2018

Para cada punto realize un programa.

1. Listas

Implemente un programa en python que

1. Construya tres listas:
   - Una de números enteros del 1 al 20
   - Otra de números reales (decimales) del 0.0 al 0.5 de 10 elemento.
   - Y otra de cadenas de nombres de 5 elementos
2. Itere sobre cada una de ellas e imprima:
   - De la de enteros, los impares (n%2 != 0)
   - De la de flotantes, los que estan entre 0.0 y 0.5
   - Y de la de nombres, los que contienen la letra “m”
3. Realize las siguiene operaciones en las listas:
   - En la de enteros, calcule e imprima la sumatoria
   - En la de flotantes extraiga e imprima el mínimo valor
   - En la de cadenas, cambie todos los nombre a MAYUSCULAS e imprimalos

2. Funciones

Implemente un programa que realice 3 funciones siguientes:

1. Una función que reciba como parámetro una lista de enteros y retorne una nueva lista con los enteros pares que existan en la lista de entrada.
2. Una funcion que reciba dos parámetros: una lista de numeros reales ("listaEntrada") y un numero real ("numeroEntrada"), la función itera sobre la lista de entrada y retorna una sublista con los número menores al numero de entrada. Ejemplo:
   ls = [0.5, 2.5, 0.7, 1.2, 0.3]
   n = 0.8
   nuevaLista = buscarMenores (ls, n)
   print (nuevaLista) // [0.5, 0.7, 0.3]
3. Defina una función que reciba dos parámetros: ***una secuencia de ADN y un gen***, y retorne verdadero o falso (True/False) si el gen aparece en la secuencia.

```
## Retorna True/False si dentro de la secuencia existe la cadena
sec = "acgtttacaaNNNaacgttacccggNNNaaaaaaNNNaacccggccNNNccaaat"
gen = "acccgg"
existe = buscarGen (sec, gen)
print (existe)  // true
```

4. Defina una función que limpie una secuencia de ADN de sitios basura ("NNN")

```
sec = "acgtttacaaNNNaacgttacccggNNNaaaaaaNNNaacccggccNNNccaaat"
nuevaSec = limpiarBasuraADN(sec)
print (nuevaSec)  // "acgtttacaaaacgttacccggaaaaaaaacccggccccaaat"
```

5. Defina una función que retorne una lista exones de una secuencia de ADN, los exones están separados por un codon "NNN":

```
sec = "acgtttacaaNNNaacgttacccggNNNaaaaaaNNNaacccggccNNNccaaat"
listaExones = buscarExones(sec)
print (listaExones)  // ["acgtttacaa", "aacgttacccgg", "aaaaaa", "aacccggcc", "ccaaat"]
```

3. Diccionarios

La siguiente es una línea de un archivo con información de un metagenoma (ecoli):

```
1 Organism  Location     Strand  Length  PID       Gene    Product
2 banana    6529..7959     -       476  170079670  yaaJ    transporter
3 rice      2801..3733     +       310  170079666  thrB    kinase
4 papaya    10643..11356   -       237  170079674  yaaW    hypothetical
```

- Construya tres diccionarios tomando como claves los encabezados de la primera línea y como valores la información con los datos de las tres líneas siguientes. Ejemplo:

```
D1 = {"organism": "banana", "location":[6529, 7959], "strand":"-","length": 476, ...}
```

- De los tres diccionarios imprima la información del organismo, la longitud , y el nombre del gen (Gene).
- Itere sobre estas listas e imprima en una sola línea la información del gen y de su producto. Así:
  - banana, 476, yaaJ
  - ...

4. Lectura de Archivos

- Cargue el archivo multifasta **"ejemplo.fasta"** con la información de varias secuencias de genes:
- Extraiga de este archivo solo los identificadores, segundo valor en los encabezados:
  - gi|**170079663**|ref|NC_010473.1| Escherichia coli str. K-12 substr
- Construya una lista con estos identificadores y guárdela en un nuevo archivo.

5. Lectura / Escritura de Archivos

- Dado las coordenadas (localización) de un gen de la ecoli en un archivo ("**ecoli.ptt**") obtener la sub-secuencia localizada en un segundo archivo ("**ecoli.fasta"**) y escribirlo esta subsecuencia en un nuevo archivo.
  - Cargar el archivo de información del gen: ecoli.ptt
  - Leer la tercera línea y extraer la localización dada en la primera posición:
    - **190..255** + 21 170079664 thrL ECDH10B_0001 - - thr operon leader peptide
  - De la localización el primer número es el inicio de la secuencia (190) mientras que el segundo es el final (255).
  - Ahora cargar el segundo archivo (ecoli.fasta) y extraer la secuencia de ADN dada en la segunda línea:
  - Recortar de esa cadena (slicing) la subsecuencia dada por laso posiciones anteriores.
  - Escribir esa subsecuencia en un nuevo archivo.

6. Código de ejemplo

6.1. Leer desde un archivo "ecoli.ptt"

```
mnjEcoli = open ("ecoli.ptt", "r")
for linea in mnjEcoli:
    if "regulatory" in linea:
        print (linea)

mnjEcoli.close ()
```

Resultados:

```
411153..411491  + 112 170080036 glnK  ECDH10B_0406 - COG0347E  nitrogen regulatory protein P--II 2
590790..592448  + 552 170080200 citA  ECDH10B_0579 - COG3290T  sensory histidine kinase in two-component regulatory system with citB
704050..705708  + 552 170080301 citA  ECDH10B_0687 - COG3290T  sensory histidine kinase in two-component regulatory system with citB
931399..931782  + 127 170080495 bssR  ECDH10B_0905 -- biofilm formation regulatory protein BssR
975517..975741  - 74  170080539 cspD  ECDH10B_0950 - COG1278K  stationary phase/starvation inducible regulatory protein CspD
```

6.2. Crear un archivo "ecoli_lg.ptt"

```
# Crear un archivo "ecoli_lg.ptt
mnjEcoli = open ("ecoli.ptt", "r")

mnjSalida = open ("ecoli-lg.ptt", "w")

for linea in mnjEcoli:
    if "regulatory" in linea:
        mnjSalida.write (linea)

mnjEcoli.close ()
mnjSalida.close ()
```

6.3. Trabajar con cadenas:split para recuperar la longitud

```
s="10830..11315 + 161 170079675 htgA  ECDH10B_0012  - - hypothetical protein"
ls = s.split ()
nombre = ls [4]
nuevoNombreArchivo = nombre+".fasta"

print (nombre)
posiciones = ls [0]
print (posiciones)
```

```
ls2 =posiciones.split ("..")
print (ls2)
pos1 = int (ls2 [0])
pos2 = int (ls2 [1])
print (pos1, pos2)
```

Resultados:

```
htgA
10830..11315
['10830', '11315']
10830 11315
```

6.4. Crear un archivo "ecoli_lg.ptt"

```
mnjEcoli = open ("ecoli.ptt", "r")
next (mnjEcoli)
next (mnjEcoli)
next (mnjEcoli)

mnjSalida = open ("ecoli-lg.ptt", "w")

for linea in mnjEcoli:
    ls = linea.split ()
    valor = int (ls[2])
    if valor < 200:
        mnjSalida.write (linea)

mnjEcoli.close ()
mnjSalida.close ()
```