cadenas-python-lg

August 27, 2018

1 Cadenas de Python

Las cadenas se usan con bastante frecuencia en todo tipo de problemas computacionales y en biología las cadenas se utilizan más todavía ya que con ellas se representan las diferentes secuencias biológicas,comó: - Secuencias de ADN - Secuencias de ARN - Secuencias de Aminoácidos - Y muchos otras aplicaciones (genes, proteínas, SNPs, etc.

1.1 Ejemplos de literales de cadenas

```
In [31]: #### Cadena simple tipo mensaje
         "Hola Mundo"
         #### Cadena que representa un nombre
         "Maria"
         #### Cadena que representa una secuenca
         "acgggtactgcaaaatctacccccaaaa"
         #### Cadena que representa un número (pero no su valor)
         "236778"

Out[31]: '236778'
```

1.2 Ejemplos de variables que almacenan cadenas

```
In [32]: #### Cadena simple tipo mensaje
         a = "Hola Mundo"
         #### Cadena que representa un nombre
         nombre = "Maria"
         #### Cadena que representa una secuenca
         adn = "acgggtactgcaaaatctacccccaaaa"
         #### Cadena que representa un número (pero no su valor)
         num = "236778"
```

2 Características de las cadenas en python

- Todas las cadenas se cierran entre comillas dobles ("xxx") o simples ('xxxx')
- Las cadenas representan otro tipo de datos en python, al igual que los enteros y los reales:
  - Para los enteros en tipo es **int**

- Para los reales el tipo es **float**
- Para las cadenas el tipo es **str**

3 Funciones básicas sobre cadenas

- Para averiguar su longitud utilizamos la función **len**
- Para pegar cadenas utilizamos el operador **+**
- Averiguar si un carácter o subcadena está dentro de la cadena usa el operador **in**

3.1 Ejemplos

```
In [33]: ## Calculo de la longitud
         adn = "acgggtactgcaaaatctacccccaaaa"
         n = len (adn)
         print (n)
```

```
29
```

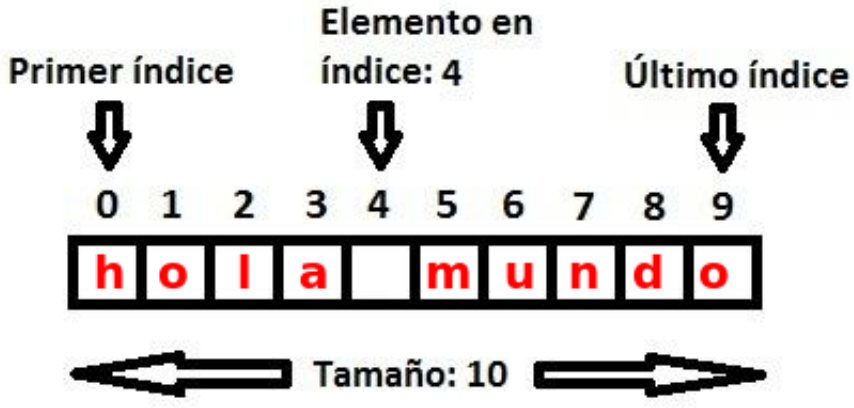
```
In [34]: ## Concatenación de dos cadenas
         cadena1 = "Hola"
         cadena2 = " "       # Espacio en blanco
         cadena3 = "Mundo"

         cadenaTotal = cadena1 + cadena2 + cadena3
         print (cadenaTotal)
```

```
Hola Mundo
```

4 Estructura de una cadena (string)

- Las cadenas se representa internamente como un arreglo de carácteres, así:::;



- La cadena va a contener **n** carácteres
- Cada carácter de la cadena está ubicado en una posición

  - El primer carácter está en la posición **0**
  - y el últirmo está en la posición **n-1**

5 Acceso a los elementos de una cadena

- El operador de acceso son los corchetes **[...]**
- Dentro de los corchetes puede ir o una posición o un rango
- Si es un rango el operador que se utiliza es los dos puntos **:**
  - El rango necesita dos posiciones: inicio y fin, así **[inicio:fin]**
  - Los elementos se incluyen desde el inicio haste el **anterior** al final
  - Se puede especificar solo el final, así **[:fin]**
  - O solo el inicio, así **[inicio:]**
  - O se puede acceder desde atraz hacia adelante, con negativos, así **[inicio:-fin]** ## Ejemplos:

```
In [35]: #### Acceso al primer elemento
         cad = "Hola Mundo Bello"
         # posi 0123456789012345
         print (cad [0])
```

```
H
```

```
In [36]: #### Asignación a una variable
         x = cad [2]
         print (x)
```

```
l
```

```
In [37]: #### Acceso a un rango
         subcadena = cad [5:10]
         print (subcadena)
```

```
Mundo
```

```
In [38]: #### Acceso a un rango sin inicio, lo toma desde 0
         subcadena = cad [:7]
         print (subcadena)
```

```
Hola Mu
```

```
In [39]: #### Acceso a un rango sin final, toma hasta el último
         subcadena = cad [5:]
         print (subcadena)
```

```
Mundo Bello
```

```
In [40]: #### Acceso desde el inicio hasta el tercer carácter desde el final
         subcadena = cad [0:-3]
         print (subcadena)
```

```
Hola Mundo Be
```

6 Recorrido de cadenas

- Para recorrer las cadenas se pueden utilizar dos instrucción de ciclos:
  - Instrucción **for**, para de a un paso: elemento por elemento
  - Instrucción **while**, para ir de más de un paso: 1, 2 o más elementos cada paso

6.1 Recorridos con ciclo for

- Para cada elemento **X** de la cadena haga:
  - Intrucciones que va a hacer con ese elemento **X**
  - Intrucciones que va a hacer con ese elemento **X**
  - ...

6.1.1 Ejemplos:

```
In [41]: ### Imprimir elemento por elemento
         cad = "Hola Mundo Bello"
         for x in cad:
             print (x)

         #
         print ("")
         print ("Adios")
```

```
H
o
l
a

M
u
n
d
o

B
e
```

```
l
l
o

Adios
```

```
In [42]: ### Concatenar la cadena ">>>>" a cada elemento
         cad = "Hola Mundo Bello"
         for x in cad:
             y = ">>>> " + x
             print (y)

         print ("")
         print ("Adios")
```

```
>>>> H
>>>> o
>>>> l
>>>> a
>>>>
>>>> M
>>>> u
>>>> n
>>>> d
>>>> o
>>>>
>>>> B
>>>> e
>>>> l
>>>> l
>>>> o

Adios
```

```
In [43]: ### Contar el número de un carácter (guaninas) en una cadena
         adn = "acgggtactgcaaaatctacccccaaaa"
         cantidad = 0
         for x in adn:
             if x == "g":
                 cantidad = cantidad + 1
             #
         #
         print ("El numero de guaninas fue: ", cantidad)
```

```
El numero de guaninas fue:  4
```

6.2 Recorridos con ciclo while

- Se deben especificar 5 elemento:
  - Indice para avanzar en el recorrido (casi siempre en cero la primera vez)
  - Variable con valor final del recorrido
  - Condición de continuación del ciclo
  - Instrucciones a ejecutar dentro del ciclo
  - Incremento para avanzar el ciclo

6.2.1 Ejemplo

```
In [44]: ## Recorre la cadena de 1 en 1 y donde encuentra una guanina imprime un mensaje
         adn = "acgggtact"

         indice = 0                  # inicia desde la posición 0
         n = len (adn)           # llega hasta el final
         while (indice < n):     # si la condicion es verdad continua sino salta todo y termina
             valor = adn [indice]   # Instrucción1 donde recupero el elemento de la cadena
             print (valor)          # Intruccion2

             indice = indice + 1
             #
         #
```

```
a
c
g
g
g
t
a
c
t
```

```
In [45]: ## Recorre INVERSO de la cadena de 1 en 1 y donde encuentra una guanina imprime un me
         adn = "acgggtact"

         n = len (adn)
         indice = n -1           # inicia desde la posición final n-1

         while (indice > 0):     # si la condicion es verdad continua sino salta todo y termina
             valor = adn [indice]   # Instrucción1 donde recupero el elemento de la cadena
             print (valor)          # Intruccion2

             indice = indice - 1
             #
         #
```

```
t
c
a
t
g
g
g
c
```

In [46]:
```
## Recorre la cadena de 3 en 3 y donde encuentra una guanina imprime un mensaje
adn = "acgggtact"

indice = 0                  # inicia desde la posición 0
final = len (adn)           # llega hasta el final
while (indice < final):     # si la condicion es verdad continua sino salta todo y ter
    valor = adn [indice]    # Instrucción1 donde recupero el elemento de la cadena
    print (valor)           # Intruccion2

    indice = indice + 3
    #
#
```

```
a
g
a
```

In [47]:
```
## Recuperación de codones (substring de tres elementos) del primer marco de lectura
adn = "acgggtact"

indice = 0                          # inicia desde la posición 0
final = len (adn)                   # llega hasta el final
while (indice < final):             # si la condicion es verdad continua sino salta t
    codon = adn [indice:indice+3]   # Instrucción1 donde recupero el elemento de la c
    print (codon)                   # Intruccion2

    indice = indice + 3
    #
#
```

```
acg
ggt
act
```

In [49]:
```
## Ejemplo4: Obtener la inversa de una cadena de ADN
adn = "acgggtact"
adnInverso = ""
```

```
        n = len (adn)                    # llega hasta el final
        i = n-1                           # inicia desde la posición 0
        while (i>=0):                  # si la condicion es verdad continua sino salta todo y term
            base = adn [i]    # Instrucción1 donde recupero el elemento de la cadena
            adnInverso = adnInverso + base
            i = i - 1
            #
        #
        print (adnInverso)
```

```
tcatgggca
```

7 Ejercicios

7.1 Uso de condicionales (if-else)

1. Dado dos cadenas de ADN: **adn1 = "acgggtact"** y **adn2 = "acgacgggt"** imprima cual es la más grande.
2. Dado tres cadenas de ADN: adn1 = **"acgggtact"**, **adn2 = "acgacgggt"**, **adn3 = "acgggtacacat"** imprimir cual de las tres es la más grande.
3. Dado la cadena **adn1 = "acgggtact"**, imprimir si es verdad que la cadena tiene el codon "gta"
4. Dado la cadena **adn1 = "acgggtact"**, imprimir cual es el codón final del primer marco de lectura.
5. Dado la cadena **adn1 = "acgggtactacg"**, imprimir si el codón de inicio es igual al codón final.

7.2 Ciclos (for, while)

1. Dado la cadena **adn1 = "acgggtactacg"**, imprimir los codones del tercer marco de lectura.
2. Dado la cadena **adn1 = "acgggtactacg"**, imprimir al final cuantas citocinas existen.
3. Dado la cadena **adn1 = "acgggtactacg"**, imprimir al final cuantas citocinas, y cuantas guaninas existen.
4. Dado la cadena **adn1 = "acgggtactacg"**, imprimir al final la cantidad por cada base (número de As, Cs, Gs, y Ts).
5. Imprimir de la cadena cadena **adn1 = "acgggtactacg"**,los codones del primer marco de lectura de la cadena inversa, es decir desde el final.
6. Realizar un programa que construya la proteína correspondiente al primer marco de la cadena de **adn1 = "acgggtactacg"**, teniendo en cuenta los siguientes aminoacidos M="ggg" R="acg" S="tac" N="acg" V="ctc" W="cgg", y el aminoácido X para cualquier otro codon que no corresponda con ninguno de los aminoácidos anteriores. Por ejemplo, para el primer marco, la proteína será: "NXXN"