mcpp_taller3_Nathalie_BastoA

August 25, 2016

1 Taller 3

Métodos Computacionales para Políticas Públicas - URosario

Entrega: viernes 26-ago-2016 11:59 PM

[Nathalie Basto A] [nathaliekupa@hotmail.com]

1.1 Instrucciones:

- Guarde una copia de este *Jupyter Notebook* en su computador, idealmente en una carpeta destinada al material del curso.
- Modifique el nombre del archivo del *notebook*, agregando al final un guión inferior y su nombre y apellido, separados estos últimos por otro guión inferior. Por ejemplo, mi *notebook* se llamaría: mcpp_taller3_santiago_matallana
- Marque el *notebook* con su nombre y e-mail en el bloque verde arriba. Reemplace el texto "[Su nombre acá]" con su nombre y apellido. Similar para su e-mail.
- Desarrolle la totalidad del taller sobre este *notebook*, insertando las celdas que sea necesario debajo de cada pregunta. Haga buen uso de las celdas para código y de las celdas tipo *markdown* según el caso.
- Recuerde salvar periódicamente sus avances.
- Cuando termine el taller:

    1. Descárguelo en PDF.
    2. Suba los dos archivos (.pdf y .ipynb) a su repositorio en GitHub antes de la fecha y hora límites.

(El valor de cada ejercicio está en corchetes [ ] después del número de ejercicio.)

Antes de iniciar, por favor descarge el archivo mcpp_taller3_listas_ejemplos.py del repositorio, guárdelo en la misma carpeta en la que está trabajando este taller y ejecútelo con el siguiente comando:

```
In [2]: run mcpp_taller3_listas_ejemplos.py

In [3]: l0

Out[3]: []
```

```
In [4]: l1

Out[4]: [1, 'abc', 5.7, [1, 3, 5]]

In [5]: l2

Out[5]: [10, 11, 12, 13, 14, 15, 16]
```

Este archivo contiene tres listas (l0, l1 y l2) que usará para las tareas de esta sección. Puede ver los valores de las listas simplemente escribiendo sus nombres y ejecutándolos en el Notebook. Inténtelo para verificar que mcpp_taller3_listas_ejemplos.py quedó bien cargado. Debería ver:
In [1]: l0 Out[1]: []
In [2]: l1 Out[2]: [1, 'abc', 5.7, [1, 3, 5]]
In [3]: l2 Out[3]: [10, 11, 12, 13, 14, 15, 16]

1.2 1. [1]

Cree una lista que contenga los elementos 7, "xyz" y 2.7.

```
In [6]: lista3 = [7, "xyz", 2.7]

In [7]: lista3

Out[7]: [7, 'xyz', 2.7]
```

1.3 2. [1]

Halle la longitud de la lista l1.

```
In [8]: len(l1)

Out[8]: 4
```

1.4 3. [1]

Escriba expresiones para obtener el valor 5.7 de la lista l1 y para obtener el valor 5 a partir del tercer elemento de l1.

```
In [10]: l1[2]

Out[10]: 5.7

In [12]: l1[3][2]

Out[12]: 5
```

1.5 4. [1]

Prediga qué ocurrirá si se evalúa la expresión l1[4] y luego pruébelo.
Saldra un error puesto que el ultimo elemento de la lista l1 se ubica en la posición 3, en la posicion 4 ya no hay nada.

```
In [13]: l1[4]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-13-2c2a81dbcaa5> in <module>()
----> 1 l1[4]

IndexError: list index out of range
```

1.6 5. [1]

Prediga qué ocurrirá si se evalúa la expresión l2[-1] y luego pruébelo.
saldra el numero 16 que es el que esta en la ultima posición de la lista l2

```
In [14]: l2[-1]
```

```
Out[14]: 16
```

1.7 6. [1]

Escriba una expresión para cambiar el valor 3 en el tercer elemento de l1 a 15.0.

```
In [15]: l1[3][1]=15.0
```

```
In [16]: l1
```

```
Out[16]: [1, 'abc', 5.7, [1, 15.0, 5]]
```

1.8 7. [1]

Escriba una expresión para crear un "slice" que contenga del segundo al quinto elemento (inclusive) de la lista l2.

```
In [18]: l2[1:5] ##saca del elemento #2, el 3, el 4 y el 5
```

```
Out[18]: [11, 12, 13, 14]
```

1.9 8. [1]

Escriba una expresión para crear un "slice" que contenga los primeros tres elementos de la lista l2.

```
In [20]: l2[:3]

Out[20]: [10, 11, 12]
```

1.10 9. [1]

Escriba una expresión para crear un "slice" que contenga del segundo al último elemento de la lista l2.

```
In [21]: l2[1:]

Out[21]: [11, 12, 13, 14, 15, 16]
```

1.11 10. [1]

Escriba un código para añadir cuatro elementos a la lista l0 usando la operación append y luego extraiga el tercer elemento (quítelo de la lista). ¿Cuántos "appends" debe hacer?

```
In [32]: l0=[]

In [33]: nuevos = ["hola", 2, 999.9, 3]  ##estoy haciendo 4 appends 1 porcada eleme
         for nuevo in nuevos:
             l0.append(nuevo)

In [34]: l0

Out[34]: ['hola', 2, 999.9, 3]

In [35]: ##ahora voy a extraer el tercer elemento de la lista
         del l0[2]

In [36]: l0 ##ahora me borro el 999.9 que era el tercer elemento

Out[36]: ['hola', 2, 3]
```

1.12 11. [1]

Cree una nueva lista nl concatenando la nueva versión de l0 con l1, y luego actualice un elemento cualquiera de nl. ¿Cambia alguna de las listas l0 o l1 al ejecutar los anteriores comandos?

```
In [38]: n1=l0 + l1

In [40]: n1

Out[40]: ['hola', 2, 3, 1, 'abc', 5.7, [1, 15.0, 5]]
```

```
In [42]: ##para modificar un elemento de n1 voy a cambair el 15.0 que habia cambiad
         n1[6][1]=3
         n1

Out[42]: ['hola', 2, 3, 1, 'abc', 5.7, [1, 3, 5]]

In [43]: l1 ##esta lista no se ha transformado

Out[43]: [1, 'abc', 5.7, [1, 3, 5]]

In [44]: l0 ##esta lista no se ha transformado porque todo lo que hice fue en una n

Out[44]: ['hola', 2, 3]
```

1.13 12. [2]

Escriba un loop que compute una variable all_pos cuyo valor sea True si todos los elementos de la lista l3 son positivos y False en otro caso.

```
In [109]: l3 = [2,-1,-7,6,0,28]

In [110]: all_pos = True
          for num in l3:
              if num <= 0:
                  all_pos=False
          print("all_pos", all_pos)##tan pronto encuentra un numero en la lista cuy
              ##osea si todos son positivos entonces la variable permanece en TRUE
          if all_pos == True:
              print("todos son positivos")
          else:
              print("al menos uno no es positivo")

all_pos False
al menos uno no es positivo
```

1.14 13. [2]

Escriba un código para crear una nueva lista que contenga solo los valores positivos de la lista l3.

```
In [111]: l3positiva = []
          for i in range(len(l3)):
              if l3[i] > 0:
                  l3positiva.append(l3[i])
                  ##print(l3positiva)
          print("los positivos en l3 estan en la nueva lista l3positiva=", l3positi

los positivos en l3 estan en la nueva lista l3positiva= [2, 6, 28]
```

1.15 14. [2]

Escriba un código que use append para crear una nueva lista nl en la que el i-ésimo elemento de nl tiene el valor True si el i-ésimo elemento de l3 tiene un valor positivo y Falso en otro caso.

```
In [112]: print(l3)
          n1=[]
          for num in l3:
              if num>0:
                  n1.append(True)
              else:
                  n1.append(False)
          print(n1)
```

```
[2, -1, -7, 6, 0, 28]
[True, False, False, True, False, True]
```

1.16 15. [3]

Escriba un código que use range, para crear una nueva lista nl en la que el i-ésimo elemento de nl es True si el i-ésimo elemento de l3 es positivo y Falso en otro caso.

Pista: Comience por crear una lista de longitud adecuada, con False en cada índice.

```
In [117]: print(l3)
          n1=[False]*(len(l3)) ##me crea una lista con el mismo # de elementos de n


          for i in range(len(l3)): ##el rango me toma los valores de cero hasta el
              if l3[i]>0:
                  n1[i]=True  ##si el numero en la posición i de l3 es positivo, la
          print(n1)
```

```
[2, -1, -7, 6, 0, 28]
[False, False, False, False, False, False]
[True, False, False, True, False, True]
```

1.17 16. [4]

En clase construimos una lista con 10000 números aleatorios entre 0 y 9, a partir del siguiente código:

```
In [118]: import random

          N = 10000
          random_numbers = []
          for i in range(N):
              random_numbers.append(random.randint(0,9))
          print(random_numbers)
```

```
[9, 5, 3, 3, 3, 4, 0, 0, 3, 1, 6, 2, 2, 8, 4, 8, 1, 2, 4, 4, 2, 5, 5, 0, 7, 0, 6, 2
```

Y creamos un "contador" que calcula la frecuencia de ocurrencia de cada número del 0 al 9, así:

```
In [123]: count = []
          for x in range(0,10):
              count.append(random_numbers.count(x))
          print(count)

[1012, 1079, 979, 1038, 986, 943, 980, 969, 1032, 982]
```

Cree un "contador" que haga lo mismo, pero sin hacer uso del método "count". (De hecho, sin usar método alguno.)

```
In [122]: freq = [0]*10
          print(freq)
          for num in random_numbers:
              for i in range(10):
                  if num == i:
                      freq[i]=freq[i]+1
          print(freq)

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[1012, 1079, 979, 1038, 986, 943, 980, 969, 1032, 982]
```

Pistas:

- Esto puede lograrse con un loop muy sencillo. Si su código es complejo, piense el problema de nuevo.
- Es muy útil iniciar con una lista "vacía" de 10 elementos. Es decir, una lista con 10 ceros.
