Taller 3

Métodos Computacionales para Políticas Públicas - URosario

Entrega: viernes 22-feb-2019 11:59 PM

Francisco Monsalve

francisco.monsalve@urosario.edu.co

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 2019_1_mcpp_taller_3_listas_ejemplos.py del repositorio, guárdelo en la misma carpeta en la que está trabajando este taller y ejecútelo con el siguiente comando:

run 2019_1_mcpp_taller_3_listas_ejemplos.py

```
In [9]: run 2019_1_mcpp_taller_3_listas_ejemplos.py

In [2]: l0

Out[2]: []

In [3]: l1

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

In [4]: l2

Out[4]: [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 2019_1_mcpp_taller_3_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. [1]

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

In [5]:
```
list1 = ["xyz", 2.7]
list1
```

Out[5]:
```
['xyz', 2.7]
```

2. [1]

Halle la longitud de la lista l1.

In [6]:
```
len(list1)
```

Out[6]:
```
2
```

3. [1]

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

In [7]:
```
print(l1[2])

print(l1[3][2])
```

```
5.7
5
```

4. [1]

Prediga qué ocurrirá si se evalúa la expresión `l1[4]` y luego pruébelo.

Error porque la l1 tiene hasta 3 elementos en su rango, no 4.

In [8]:
```
l1[4]
```

```
---------------------------------------------------------------------
----
IndexError                                Traceback (most recent call l
ast)
<ipython-input-8-7ffdcb2c9f2e> in <module>
----> 1 l1[4]

IndexError: list index out of range
```

5. [1]

Prediga qué ocurrirá si se evalúa la expresión `l2[-1]` y luego pruébelo.

Muestra el último elemento de la lista l2, en este caso es 16

In [10]:
```
l2[-1]
```

Out[10]:
```
16
```

6. [1]

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

In [11]:
```
l1=[1, 'abc', 5.7, [1, 3, 5]]
l1
```

Out[11]:
```
[1, 'abc', 5.7, [1, 3, 5]]
```

In [12]:
```
l1[3][1] = 15.0
l1
```

Out[12]:
```
[1, 'abc', 5.7, [1, 15.0, 5]]
```

7. [1]

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

In [13]:
```
l2[1:4]
```

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

8. [1]

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

In [14]:
```
l2[:3]
```

Out[14]:
```
[10, 11, 12]
```

9. [1]

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

In [15]:
```
l2[2:]
```

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

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 [16]:
```
l0 =[]
l0
```

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

In [17]:
```
l0.append("e1")
l0.append("e2")
l0.append("e3")
l0.append("e4")
print(l0)
l0[2]
#realiza 4 elementos
```

```
['e1', 'e2', 'e3', 'e4']
```

Out[17]:
```
'e3'
```

In [ ]:
```

```

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 [18]:
```
n1=[]
n1
```

Out[18]:
```
[]
```

In [19]:
```
l0.append(l1)
print(l0)
```

```
['e1', 'e2', 'e3', 'e4', [1, 'abc', 5.7, [1, 15.0, 5]]]
```

In [20]:
```
n1=l0
```

In [21]:
```
n1[0]="e0"
n1
```

Out[21]:
```
['e0', 'e2', 'e3', 'e4', [1, 'abc', 5.7, [1, 15.0, 5]]]
```

In [22]:
```
print(l0)
print (l1)
#Cambia la lista de l0 al hacer el append y asignarla a n1
```

```
['e0', 'e2', 'e3', 'e4', [1, 'abc', 5.7, [1, 15.0, 5]]]
[1, 'abc', 5.7, [1, 15.0, 5]]
```

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 [23]:
```
l3 =[1,-4,5,6,1,5]
l3
```

Out[23]:
```
[1, -4, 5, 6, 1, 5]
```

In [24]:
```
for x in l3:
    if x < 0:
```

```
        all_pos = False
        break
    else:
        all_pos= True
print(all_pos)
```

```
False
```

In [ ]:

13. [2]

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

In [25]:
```
list_nueva = []

for x in l3:
    if x > 0:
        list_nueva.append(x)
print(list_nueva)
```

```
[1, 5, 6, 1, 5]
```

In [ ]:

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 [26]:
```
n1_nueva = []
```

```
n1_nueva
```

Out[26]:
```
[]
```

In [27]:
```
for x in l3:
    if x > 0:
        valor=True
        n1_nueva.append(valor)
    else :
        valor=False
        n1_nueva.append(valor)

print(l3)
print(n1_nueva)
```

```
[1, -4, 5, 6, 1, 5]
[True, False, True, True, True, True]
```

In [ ]:
```

```

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 False en otro caso.

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

In [29]:
```
n1_segunda = [False, False,False,False,False,False]
len(n1_segunda)
l3 = [1, -4, 5, 6, 1, 5]
```

In [30]:
```
tam = len(n1_segunda)
for j in range(tam):
    if (l3[j] > 0):
        n1_segunda[j] = True
    else:
```

```
        n1_segunda[j] = False
print(n1_segunda)
```

```
[True, False, True, True, True, True]
```

16. [4]

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

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

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

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

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

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.

In [31]:
```
import random

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

In [ ]:
```
print(random_numbers)
```

In [32]:
```
lista = [0,0,0,0,0,0,0,0,0,0]
```

```
len(lista)
```

Out[32]:
```
10
```

In [33]:
```
for i in range(len(random_numbers)):
    if (random_numbers [i] == 0):
        lista[0] = 1 + lista [0]
    elif (random_numbers [i] == 1):
        lista[1] = 1 + lista [1]
    elif (random_numbers [i] == 2):
        lista[2] = 1 + lista [2]
    elif (random_numbers [i] == 3):
        lista[3] = 1 + lista [3]
    elif (random_numbers [i] == 4):
        lista[4] = 1 + lista [4]
    elif (random_numbers [i] == 5):
        lista[5] = 1 + lista [5]
    elif (random_numbers [i] == 6):
        lista[6] = 1 + lista [6]
    elif (random_numbers [i] == 7):
        lista[7] = 1 + lista [7]
    elif (random_numbers [i] == 8):
        lista[8] = 1 + lista [8]
    elif (random_numbers [i] == 9):
        lista[9] = 1 + lista [9]

print(lista)
```

```
[1024, 999, 962, 1049, 940, 1017, 981, 996, 985, 1047]
```

In [ ]:
```

```