

Taller 3

Métodos Computacionales para Políticas Públicas - URosario

Entrega: viernes 26-feb-2021 11:59 PM

Valentina Cuenca

norma.cuenca@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 descargue el archivo [2021-I_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:

In []: run [2021-I_mcpp_taller_3_listas_ejemplos.py](#)

Este archivo contiene tres listas ([10](#), [11](#) y [12](#)) 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 [2021-I_mcpp_taller_3_listas_ejemplos.py](#) quedó bien cargado. Debería ver:

In [1]: 10

Out[1]: []

In [2]: 11

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

In [3]: 12

Out[3]: [10, 11, 12, 13, 14, 15, 16]

1. [1]

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

In [67]: lista=[7, "xyz", 2.7]

2. [1]

Halle la longitud de la lista 11.

In [68]: len(11)

Out[68]: 4

3. [1]

Escriba expresiones para obtener el valor 5.7 de la lista 11 y para obtener el valor 5 a partir del cuarto elemento de 11.

In [69]: 11[2]

Out[69]: 5.7

In [8]: 11[3][2]

Out[8]: 5

4. [1]

Prediga qué ocurrirá si se evalúa la expresión 11[4] y luego pruébelo.

Respuesta

Nos va a dar error de indexacion.

In [70]: 11[4]

IndexError

Traceback (most recent call last)

<ipython-input-70-9ee375f35245> in <module>

----> 1 11[4]

IndexError: list index out of range

5. [1]

Prediga qué ocurrirá si se evalúa la expresión `12[-1]` y luego pruébelo.

Respuesta

Creo que nos da 16

```
In [76]: 12[-1]
```

```
Out[76]: 16
```

6. [1]

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

```
In [77]: l1[3][1]= 15.0  
l1
```

```
Out[77]: [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 [78]: s=l2[1:6]  
s
```

```
Out[78]: [11, 12, 13, 14, 15]
```

8. [1]

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

```
In [79]: m=l2[0:3]  
m
```

```
Out[79]: [10, 11, 12]
```

9. [1]

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

```
In [80]: w=l2[1:]  
w
```

```
Out[80]: [11, 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 (quitelo de la lista). ¿Cuántos "appends" debe hacer?

Respuesta

Se debe hacer 4 appends

```
In [81]: l0=[]
for i in range(4):
    l0.append(i)
print(l0)
l0.pop(2)
l0
```

[0, 1, 2, 3]

```
Out[81]: [0, 1, 3]
```

11. [1]

Cree una nueva lista n1 concatenando la nueva versión de l0 con l1, y luego actualice un elemento cualquiera de n1. ¿Cambia alguna de las listas l0 o l1 al ejecutar los anteriores comandos?

Respuesta

```
In [82]: n1=l0 + l1
n1
```

```
Out[82]: [0, 1, 3, 1, 'abc', 5.7, [1, 15.0, 5]]
```

```
In [83]: n1[2]="Holaaaaaaaa"
n1
```

```
Out[83]: [0, 1, 'Holaaaaaaaa', 1, 'abc', 5.7, [1, 15.0, 5]]
```

Cambiar n1 no afecta ni a l0 ni l1.

```
In [84]: print(l0)
print(l1)
```

[0, 1, 3]

[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 [85]: c=0
for i in range(len(l2)):
```

```

if l2[i] > 0:
    c = c + 1
if c == len(l2):
    all_pos=True
else:
    all_pos= False

all_pos

```

Out[85]: True

13. [2]

Escriba un código para crear una nueva lista que contenga solo los valores positivos de la lista 13.

```

In [86]: l3=[1,-3,4,-6,7,-4,9,-5,2,-9]
positivos=[]
for i in range(len(l3)):
    if l3[i] > 0:
        positivos.append(l3[i])
positivos

```

Out[86]: [1, 4, 7, 9, 2]

14. [2]

Escriba un código que use append para crear una nueva lista n1 en la que el i-ésimo elemento de n1 tiene el valor True si el i-ésimo elemento de l3 tiene un valor positivo y Falso en otro caso.

```

In [87]: n1= []
for i in range(len(l3)):
    if l3[i] > 0:
        n1.append(True)
    else:
        n1.append(False)
n1

```

Out[87]: [True, False, True, False, True, False, True, False, True, False]

15. [3]

Escriba un código que use range, para crear una nueva lista n1 en la que el i-ésimo elemento de n1 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 [88]: n1= []
for i in range(len(l3)):
    n1.append(False)
for i in range(len(l3)):
    if l3[i]>0:

```

```
n1[i]=True
```

```
n1
```

```
Out[88]: [True, False, True, False, True, False, True, False, True, False]
```

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 [89]: import random
```

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

for x in range(0,10):
    countx=0
    for y in range(len(random_numbers)):
        if x==random_numbers[y]:
            countx= countx + 1
    count.append(countx)
count
```

```
Out[89]: [1002, 1046, 982, 1001, 991, 991, 981, 1055, 976, 975]
```