mcpp_taller5_julian_ramirez

September 6, 2019

1 Taller 5

Métodos Computacionales para Políticas Públicas - URosario
Entrega: viernes 6-sep-2019 11:59 PM
Julián Santiago Ramírez julians.ramirez@urosario.edu.co

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_taller5_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. Si tiene algún problema con la conversión, descárguelo en HTML.
  2. Suba los dos archivos (.pdf -o .html- y .ipynb) a su repositorio en GitHub antes de la fecha y hora límites.

(Todos los ejercicios tienen el mismo valor.)

1.1.1 1

Escríba una función que ordene (de forma ascedente y descendente) un diccionario según sus valores.

```
In [115]: def ordenar(dicc,forma):
              if forma=="ascendente":
                  ## Creamos una lista con los items del diccionario
                  b=list(dicc.items())
```

```
            ## Dado que b es una lista podemos usar el metodo "sort"
            ## Este metodo requiere que se le diga que se quiere organizar en la lista,
            ## Luego seleccionamos solo los valores y esto lo logramos con una funcion l
            b.sort(key=lambda x:x[1])
            diccionario_nuevo={}
            for i in b:
                diccionario_nuevo[i[0]]=i[1]
            return diccionario_nuevo
        elif forma=="descendente":
            b=list(dicc.items())
            ## Dado que b es una lista podemos usar el metodo "sort"
            ## Este metodo requiere que se le diga que se quiere organizar en la lista,
            ## Luego seleccionamos solo los valores y esto lo logramos con una funcion l
            b.sort(key=lambda x:x[1],reverse=True)
            diccionario_nuevo={}
            for i in b:
                diccionario_nuevo[i[0]]=i[1]
            return diccionario_nuevo


    dicc={"a":4,"b":3,"c":9,"d":5,"e":8,"f":1}
    nuevo=ordenar(dicc,"ascendente")
    print ("diccionario principal: ", dicc)
    print ("diccionario organizado: ", nuevo)
    nuevo=ordenar(dicc,"descendente")
    print ("diccionario principal: ", dicc)
    print ("diccionario organizado: ", nuevo)
```

```
diccionario principal:  {'a': 4, 'b': 3, 'c': 9, 'd': 5, 'e': 8, 'f': 1}
diccionario organizado:  {'f': 1, 'b': 3, 'a': 4, 'd': 5, 'e': 8, 'c': 9}
diccionario principal:  {'a': 4, 'b': 3, 'c': 9, 'd': 5, 'e': 8, 'f': 1}
diccionario organizado:  {'c': 9, 'e': 8, 'd': 5, 'a': 4, 'b': 3, 'f': 1}
```

```
In [108]: lista=[1,2,3,4,5,6]
          lista.reverse()
```

1.1.2 2

Escriba una función que agregue una llave a un diccionario.

```
In [46]: ### El programa no especifica que valor debo agregar con la llave por ende la agrego
```

```
    def agregar(dicc,llave):
        dicc[llave]=0
        return dicc

    dicc={"a":4,"b":3,"c":9,"d":5,"e":8,"f":1}
```

```
        dicc=agregar(dicc,"y")
        print ("diccionario principal: ", dicc)
```

```
diccionario principal:  {'a': 4, 'b': 3, 'c': 9, 'd': 5, 'e': 8, 'f': 1, 'y': 0}
```

1.1.3 3

Escriba un programa que concatene los siguientes tres diccionarios en uno nuevo:
dicc1 = {1:10, 2:20} dicc2 = {3:30, 4:40} dicc3 = {5:50,6:60} Resultado esperado: {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}

```
In [53]: dicc1 = {1:10, 2:20}
         dicc2 = {3:30, 4:40}
         dicc3 = {5:50,6:60}

         d1=list(dicc1.items())
         d2=list(dicc2.items())
         d3=list(dicc3.items())

         total=d1+d2+d3

         diccionario_nuevo={}
         for i in total:
             diccionario_nuevo[i[0]]=i[1]

         print("diccionario concatenado: ",diccionario_nuevo)
```

```
diccionario concatenado:  {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
```

1.1.4 4

Escriba una función que verifique si una determinada llave existe o no en un diccionario.

```
In [63]: def existencia(dicc,llave):
             lista=list(dicc.items())
             booll=False
             for i in lista:
                 if i[0]==llave:
                     booll=True
                     print("la llave "+"'"+str(llave)+"'"+" esta en el diccionario")
                     break
             if booll==False:
                 print("'"+str(llave)+"'"+"No esta en el diccionario")

         dicc={"a":4,"b":3,"c":9,"d":5,"e":8,"f":1}
         existencia(dicc,"a")
         existencia(dicc,"k")
```

```
la llave ''a'' esta en el diccionario
''k''No esta en el diccionario
```

1.1.5 5

Escriba una función que imprima todos los pares (llave, valor) de un diccionario.

```
In [65]: def imprimir (dicc):
             for k,v in dicc.items():
                 tup=(k,v)
                 print(tup)
         dicc={"a":4,"b":3,"c":9,"d":5,"e":8,"f":1}
         imprimir(dicc)
```

```
('a', 4)
('b', 3)
('c', 9)
('d', 5)
('e', 8)
('f', 1)
```

1.1.6 6

Escriba una función que genere un diccionario con los números enteros entre 1 y n en la forma (x: x**2).

```
In [71]: def generar(n):
             diccionario={}
             for i in range(1,n+1):
                 valor=i**2
                 diccionario[i]=valor
             return diccionario

         d=generar(10)
         print("diccionario generado: ",d)
```

```
diccionario generado:  {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100}
```

1.1.7 7

Escriba una función que sume todas las llaves de un diccionario. (Asuma que son números.)

```
In [75]: def suma(dicc):
             lla=list(dicc.keys())
             x=sum(lla)
             print("La suma de las llaves del dicccionario es: ",x)
```

```
        d={10: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
        suma(d)
```

```
La suma de las llaves del dicccionario es:  30
```

1.1.8 8

Escriba una función que sume todos los valores de un diccionario. (Asuma que son números.)

```
In [76]: def suma_valores(dicc):
             val=list(dicc.values())
             x=sum(val)
             print("La suma de los valores del dicccionario es: ",val)

         d={10: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
         suma_valores(d)
```

```
La suma de los valores del dicccionario es:  [10, 20, 30, 40, 50, 60]
```

1.1.9 9

Escriba una función que sume todos los ítems de un diccionario. (Asuma que son números.)

```
In [82]: def suma_total(dicc):
             lista=list(dicc.items())
             total=0
             for i in lista:
                 s=sum(i)
                 total=total+s
             print("Suma de llaves y valores: ", total)

         d={10: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
         suma_total(d)
```

```
Suma de llaves y valores:  240
```

1.1.10 10

Escriba una función que tome dos listas y las mapee a un diccionario por pares. (El primer elemento de la primera lista es la primera llave del diccionario, el primer elemento de la segunda lista es el valor de la primera llave del diccionario, etc.)

```
In [84]: def crear_diccionario(lista1,lista2):
             diccionario={}
             ### En principio asumire que es necesario que ambas listas tengan el mismo tamaño
```

```
            ### ... diciendo que tienen tamaños diferentes.
            if len(lista1)==len(lista2):
                for i in range(0,len(lista1)):
                    diccionario[lista1[i]]=lista2[i]
                print("Diccionario resultante: ", diccionario)
            else:
                print("Es necesario que el tamaño de las dos listas sea igual")



        nombres=["julian","pablo","juana","laura"]
        edad=[20,10,15,12]
        crear_diccionario(nombres,edad)
```

```
Diccionario resultante:  {'julian': 20, 'pablo': 10, 'juana': 15, 'laura': 12}
```

1.1.11 11

Escriba una función que elimine una llave de un diccionario.

```
In [87]: def eliminar(dicc,llave):
            llav=list(dicc.keys())
            u=False
            for i in llav:
                if i==llave:
                    u=True
                    del dicc[llave]
                    return dicc
                    break
            if u==False:
                print("La llave "+str(llave)+" no fue encontrada en el diccionario")

        d={'julian': 20, 'pablo': 10, 'juana': 15, 'laura': 12}
        dic=eliminar(d,"julian")
        print("nuevo diccionario: ", dic)
        a=eliminar(d,"ana")
```

```
nuevo diccionario:  {'pablo': 10, 'juana': 15, 'laura': 12}
La llave ana no fue encontrada en el diccionario
```

1.1.12 12

Escriba una función que arroje los valores mínimo y máximo de un diccionario.

```
In [93]: def encontrar(dicc):
            ## Asumo que todos los valores en el diccionario son del mismo tipo
            val=list(dicc.values())
```

```
            minimo=min(val)
            maximo=max(val)
            print("Valor minimo: ", minimo)
            print("valor maximo: ", maximo)

        d={'julian': 20, 'pablo': 10, 'juana': 15, 'laura': 12}
        encontrar(d)
```

```
Valor minimo:  10
valor maximo:  20
```

1.1.13 13

sentence = "the quick brown fox jumps over the lazy dog" words = sentence.split() word_lengths = [] for word in words: if word != "the": word_lengths.append(len(word))

Simplifique el código anterior combinando las líneas 3 a 6 usando list comprehension. Su código final deberá entonces tener tres líneas.

```
In [102]: sentence = "the quick brown fox jumps over the lazy dog"
          words = sentence.split()
          word_lengths_mio = [len(x) for x in words if x!="the"]
          ##################### prueba de que esta bien
          print(word_lengths_mio)
```

```
[5, 5, 3, 5, 4, 4, 3]
```

1.1.14 14

Escriba UNA línea de código que tome la lista a y arroje una nueva lista con solo los elementos pares de a.

```
In [103]: ## En realidad dos por que necesito escribir la lista a
          a=[1,2,3,4,5,6,7,8,9,20,22,44]
          print([x for x in a if x%2==0])
```

```
[2, 4, 6, 8, 20, 22, 44]
```

1.1.15 15

Escriba UNA línea de código que tome la lista a del ejercicio 14 y multiplique todos sus valores.

```
In [105]: from functools import reduce
```

```
In [106]: reduce(lambda x,y: x*y, a)
```

```
Out[106]: 7025356800
```

1.1.16 16

Usando "list comprehension", cree una lista con las 36 combinaciones de un par de dados, como tuplas: [(1,1), (1,2),...,(6,6)].

```
In [107]: [(x,y) for x in range(1,7) for y in range(1,7)]
```

```
Out[107]: [(1, 1),
           (1, 2),
           (1, 3),
           (1, 4),
           (1, 5),
           (1, 6),
           (2, 1),
           (2, 2),
           (2, 3),
           (2, 4),
           (2, 5),
           (2, 6),
           (3, 1),
           (3, 2),
           (3, 3),
           (3, 4),
           (3, 5),
           (3, 6),
           (4, 1),
           (4, 2),
           (4, 3),
           (4, 4),
           (4, 5),
           (4, 6),
           (5, 1),
           (5, 2),
           (5, 3),
           (5, 4),
           (5, 5),
           (5, 6),
           (6, 1),
           (6, 2),
           (6, 3),
           (6, 4),
           (6, 5),
           (6, 6)]
```
