

Escribir programas (o programar) es una actividad muy creativa y gratificante. Puedes escribir programas por muchas razones, desde resolver un problema complicado de análisis de datos hasta pasar un rato divertido con alguien resolviendo un problema. Este curso asume que todos necesitan saber cómo programar, y que, una vez que aprendes a programar, serás capaz de encontrar qué quieres hacer con ese nuevo conocimiento.

En nuestra vida diaria nos encontramos rodeados de computadoras, desde computadoras portátiles hasta teléfonos celulares. Podemos pensar en esas computadoras como “asistentes personales” que pueden ocuparse de muchas cosas por nosotros. El hardware en las computadoras de hoy es esencialmente construido para preguntarnos continuamente, “¿Qué te gustaría que haga ahora?”

Nuestras computadoras son rápidas y tienen cantidades grandes de memoria, y pueden sernos muy útiles solamente si sabemos hablar el lenguaje correcto para explicarle a la computadora lo que queremos que haga ahora. Si supiéramos este lenguaje podríamos decirle a la computadora que se encargue de las tareas que repetimos con frecuencia. Es interesante saber que las cosas que las computadoras pueden hacer mejor son con frecuencia las cosas que los humanos encontramos aburridas y poco interesantes.

# Patrón del Contador

```
contadores = dict()
print('Ingresa una línea de texto:')
lineaa = input(' ')

palabras = lineaa.split()

print('Palabras:', palabras)

print('Contando...')
for palabra in palabras:
    contadores[palabra] = contadores.get(palabra, 0) + 1
print('Contadores', contadores)
```

El patrón general para contar las palabras en una línea de texto es **dividir** la línea en palabras, y después recorrer las palabras y usar un **diccionario** para mantener la cuenta de cada palabra de forma independiente.

```
python contador_palabras.py
```

Ingresa una línea de texto:

```
the clown ran after the car and the car ran into the tent  
and the tent fell down on the clown and the car
```

```
Palabras: ['el', 'payaso', 'corrio', 'detras', 'del',  
'carro', 'y', 'el', 'carro', 'corrio', 'dentro', 'de',  
'la', 'tienda', 'y', 'la', 'tienda', 'cayo', 'sobre',  
'el', 'payaso', 'y', 'el', 'carro']
```

Contando...

```
Contadores {'el': 4, 'payaso': 2, 'corrio': 2, 'detras':  
1, 'del': 1, 'carro': 3, 'y': 3, 'dentro': 1, 'de': 1,  
'la': 2, 'tienda': 2, 'cayo': 1, 'sobre': 1}
```



```
contadores = dict()
lineaa = input('Ingresa una línea de texto:')
palabras = lineaa.split()
```

```
print('Palabras:', palabras)
print('Contando...')
```

```
for palabra in palabras:
    contadores[palabra] =
contadores.get(palabra,0) + 1
print('Contadores', contadores)
```



```
python contador_palabras.py
Ingresa una línea de texto:
el payaso corrio detras del carro y el carro
corrio dentro de la tienda y la tienda cayo
sobre el payaso y el carro
```

```
Palabras: ['el', 'payaso', 'corrio', 'detras',
'del', 'carro', 'y', 'el', 'carro', 'corrio', 'dentro',
'de', 'la', 'tienda', 'y', 'la', 'tienda', 'cayo',
'sobre', 'el', 'payaso', 'y', 'el', 'carro']
```

```
Contando...
```

```
Contadores {'el': 4, 'payaso': 2, 'corrio': 2,
'detras': 1, 'del': 1, 'carro': 3, 'y': 3, 'dentro':
1, 'de': 1, 'la': 2, 'tienda': 2, 'cayo': 1, 'sobre':
1}
```

# Bucles Finitos y Diccionarios

A pesar de que los **diccionarios** no se almacenan en orden, podemos escribir un bucle **for** que recorre todas las **entradas** en un **diccionario** – de hecho recorre todas las **claves** en el **diccionario** y **busca** los valores


```
>>> contadores = { 'chuck' : 1 , 'fred' : 42, 'jan': 100}
>>> for clave in contadores:
...     print(clave, contadores[clave])
...
jan 100
chuck 1
fred 42
>>>
```

# Recuperando listas de Claves y Valores

Puedes obtener  
una lista de  
claves, valores, o  
ítems (ambos) de  
un diccionario

```
>>> jjj = { 'chuck' : 1 , 'fred' : 42, 'jan': 100}
>>> print(list(jjj))
['jan', 'chuck', 'fred']
>>> print(jjj.keys())
['jan', 'chuck', 'fred']
>>> print(jjj.values())
[100, 1, 42]
>>> print(jjj.items())
[('jan', 100), ('chuck', 1), ('fred', 42)]
>>>
```

¿Qué es una  
“tupla”? - próximamente...





# Bonus: Dos Variables de Iteración!

- Iteramos a través de los pares **clave-valor** en un diccionario usando \*dos\* variables de iteración

```
jjj = { 'chuck' : 1 , 'fred' : 42, 'jan': 100}  
for aaa,bbb in jjj.items() :  
    print(aaa, bbb)
```

```
jan 100  
chuck 1  
fred 42
```

aaa	bbb
[jan]	100
[chuck]	1
[fred]	42

- En cada iteración, la primera variable es la **clave** y la segunda variable es el **valor** correspondiente a la clave

```
nombre = input('Ingresa un nombre de archivo:')  
manejador = open(nombre)
```

```
contadores = dict()  
for linea in manejador:  
    palabras = linea.split()  
    for palabra in palabras:  
        contadores[palabra] = contadores.get(palabra,0) + 1
```

```
grancontador = None  
granpalabra = None  
for palabra,contador in contadores.items():  
    if grancontador is None or contador > grancontador:  
        granpalabra = palabra  
        grancontador = contador
```

```
print(granpalabra, grancontador)
```

```
python palabras.py  
Enter file: palabras.txt  
a 16
```

```
python palabras.py  
Enter file: payaso.txt  
el 4
```

Usando dos bucles anidados



# Resumen

- ¿Qué es una “colección”?
- Listas contra Diccionesarios
- Constantes de Diccionesarios
- La palabra más común
- Usando el método `get()`
- Indexado y falta de orden
- Escribiendo bucles de diccionarios
- Un vistazo: tuplas
- Ordenando diccionarios



## Agradecimientos / Contribuciones



Las diapositivas están bajo el Copyright 2010- Charles R. Severance ([www.dr-chuck.com](http://www.dr-chuck.com)) de la Escuela de Informática de la Universidad de Michigan y [open.umich.edu](http://open.umich.edu), y están disponibles públicamente bajo una Licencia Creative Commons Attribution 4.0. Favor de mantener esta última diapositiva en todas las copias del documento para cumplir con los requerimientos de atribución de la licencia. Si haces un cambio, siéntete libre de agregar tu nombre y organización a la lista de contribuidores en esta página conforme sean republicados los materiales.

Desarrollo inicial: Charles Severance, Escuela de Informática de la Universidad de Michigan.

Traducción al Español por Juan Carlos Pérez Castellanos - 2020-04-25