

Taller de Programación

Clase 06: Listas

Daniela Opitz
dopitz@udd.cl



Basada en presentaciones oficiales de libro Introduction to Programming in Python (Sedgewick, Wayne, Dondero).

Disponible en <https://introcs.cs.princeton.edu/python>

Outline

- Listas
- Procesamiento de datos con listas


Listas

- **Lista:** secuencia de elementos de cualquier tipo.
- **Propósito:** facilitar el almacenamiento y procesamiento de datos.

Ejemplos:

- 52 cartas en un mazo
- 27 alumnos en una clase
- 8 millones de píxeles en una imagen
- 4 mil millones de nucleótidos en una base de ADN
- 86 mil millones de neuronas en el cerebro
- $6.02 \cdot 10^{23}$ partículas en un mol

<i>index</i>	<i>value</i>
0	2♥
1	6♠
2	A♦
3	A♥
...	
49	3♣
50	K♣
51	4♠



Elementos de una Lista

Corchetes
crean una lista

Elementos separados
por coma

Tercer elemento
en la lista

```
cartas = ['Diamante', 'Corazón', 'Pica', 'Trébol']  
print(cartas[2])
```

Acceso al i-ésimo
elemento

```
$ python3.7 lista.py  
Pica
```

Importante: El primer elemento está en la posición 0

Utilidad de una Lista

Sin una lista

Tedioso y propenso a generar errores

```
a0 = 0
a1 = 0
a2 = 0
a3 = 0
a4 = 3.1
a5 = 0
a6 = 0
a7 = 0
a8 = 5.2
a9 = 0

x = a4 + a8
```

Usando una lista

Sencillo


```
a = 10*[0]
a[4] = 3.1
a[8] = 5.2

x = a[4] + a[8]
```

**No es una multiplicación.
Repite 10 veces la lista [0]**

Utilidad de una Lista

```
a = 1000000*[0]  
a[234567] = 3.1  
a[891234] = 5.2  
  
x = a[234567] + a[891234]
```



Se puede escalar a millones de elementos!

Procesando Elementos en Listas

Utilizando ciclo for

```
1 meses = ['Enero', 'Febrero', 'Marzo', 'Abril', 'Mayo', 'Junio'  
2         , 'Julio', 'Agosto', 'Septiembre', 'Octubre', 'Noviembre'  
3         , 'Diciembre']  
4  
5 for mes in meses:  
6     print(mes)
```

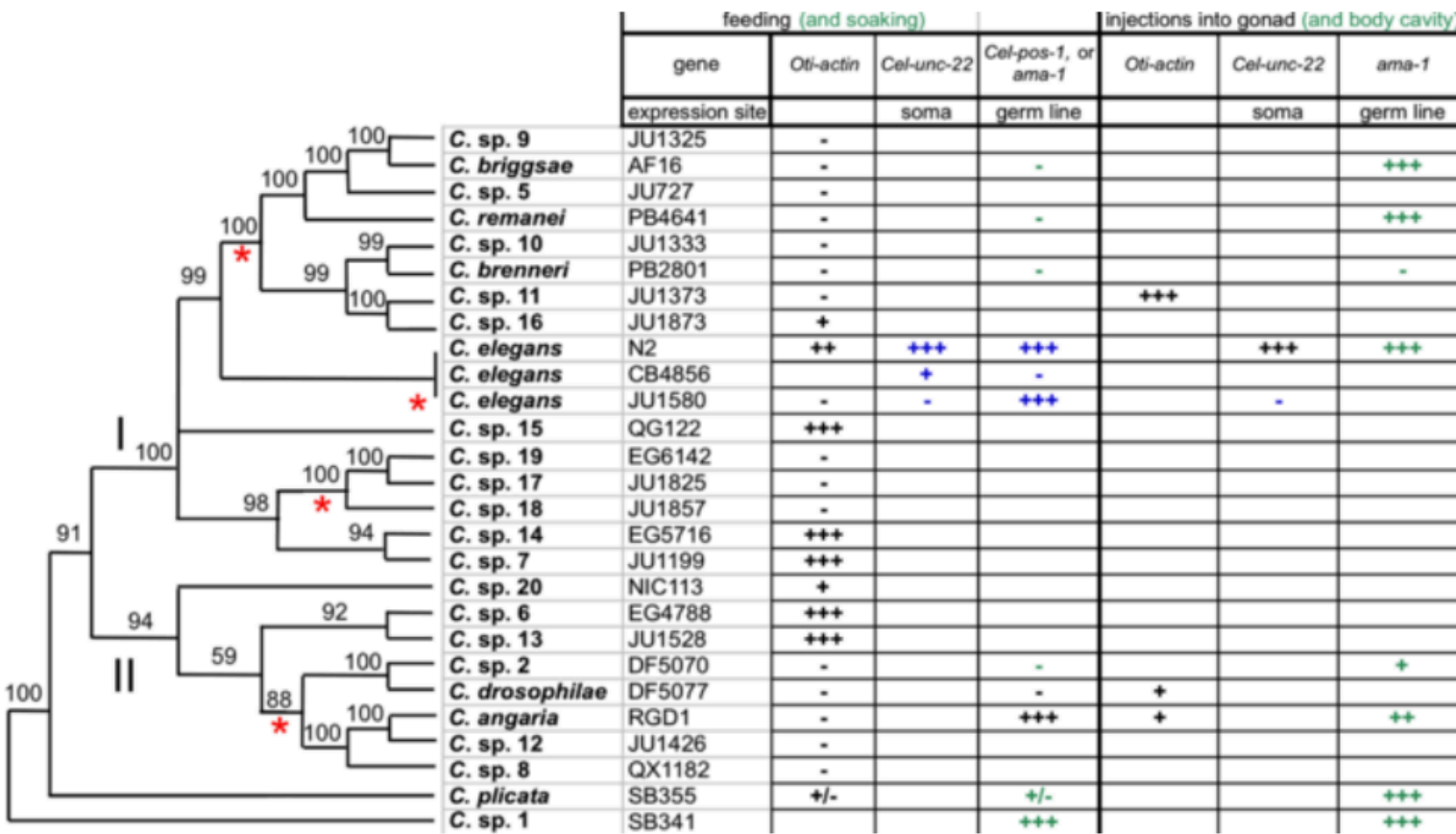
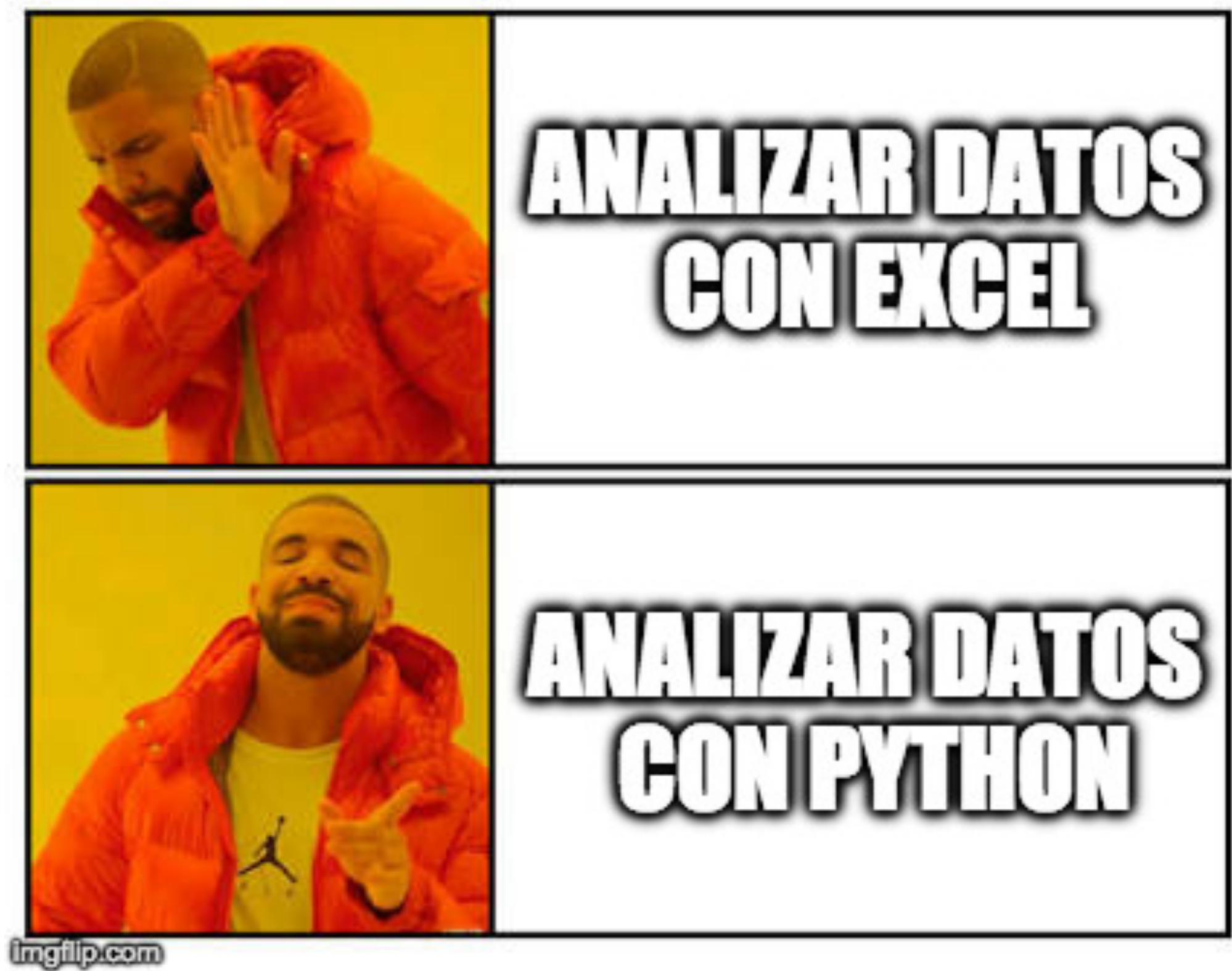
Simple, menos propenso
a creación de bugs

Utilizando ciclo for + generación de posiciones

```
1 meses = ['Enero', 'Febrero', 'Marzo', 'Abril', 'Mayo', 'Junio'  
2         , 'Julio', 'Agosto', 'Septiembre', 'Octubre', 'Noviembre'  
3         , 'Diciembre']  
4  
5 n = len(meses) # tamaño lista meses  
6  
7 for i in range(n):  
8     print(meses[i])
```

1. Genera las posiciones en el arreglo
2. Recupera el elemento en la posición i

Python vs Excel



PLOS ONE PHYLOGENY/FLICKR (CC BY 2.0)

One in five genetics papers contains errors thanks to Microsoft Excel

By [Jessica Boddy](#) | Aug. 29, 2016 , 1:45 PM

Autoformatting in Microsoft Excel has caused many a headache—but now, a new study shows that one in five genetics papers in top scientific journals **contains errors from the program**, *The Washington Post* reports. The errors often arose when gene names in a spreadsheet **were automatically changed** to calendar dates or numerical values. For example, one gene called *Septin-2* is commonly shortened to *SEPT2*, but is changed to 2-SEP and stored as the date 2 September 2016 by Excel. The researchers, who published their analysis in *Genome Biology*, say the issue can be fixed by formatting Excel columns as text and remaining vigilant—or switching to Google Sheets, where gene names are stored exactly as they're entered.

<http://www.sciencemag.org/news/2016/08/one-five-genetics-papers-contains-errors-thanks-microsoft-excel>

Procesando Elementos en Listas

P: ¿Cuándo usar for, o for + range?

R: La opción más simple es la adecuada ;)

```
1 # Calcula producto punto
2 x = [0.30, 0.60, 0.10]
3 y = [0.50, 0.10, 0.40]
4 total = 0.0
5 for i in range(len(x)):
6     total += x[i]*y[i]
7 print(total)
```

Procesando Elementos en Listas

- Los elementos de la lista se pueden acceder con el operador corchete []
- Si la posición del elemento es negativo, se accede desde el final.
- Si accedes una posición que no existe: **ERROR!**

0	1	2	3	4	5
2	3	5	7	11	
-5	-4	-3	-2	-1	

Input

```
1 L = [2, 3, 5, 7, 11]
2 print('L[0]', L[0])
3 print('L[1]', L[1])
4
5 print('L[-1]', L[-1])
6 print('L[-2]', L[-2])
7 print('L[99]', L[99])
```

Output

```
L[0] 2
L[1] 3
L[-1] 11
L[-2] 7
Traceback (most recent call last):
  File "lista-neg.py", line 7, in <module>
    print('L[99]', L[99])
IndexError: list index out of range
```

Error, programa se caerá. Lista L tiene 5 elementos.

Procesando Elementos en Listas

- Append: agregar nuevo elemento a la lista
- Concatenar: unir dos listas
- Obtener sublista: L[inicio:fin]
- Contiene: elem in L (devuelve True o False)

```
>>> 'a' in ['b','c','d','a']  
True
```

```
1 L = [11, 3, 5, 7, 2]  
2 print('L', L)  
3  
4 if 5 in L:  
5     print('cinco está en L')  
6  
7 # Actualizar elemento  
8 L[4] = 9999  
9 print('L[4]=9999', L)  
10  
11 # Agregar elemento a listas  
12 L.append(100) #Modifica lista  
13 print('L.append(100)', L)  
14  
15 # Concatenar lista  
16 L2 = L + [19, 17, 13] #Crea lista nueva  
17 print('L+[19, 17, 13]', L2)  
18  
19 # Sublista  
20 L3 = L[2:5] # Elementos 2,3 y 4  
21 print('L[2:5]', L3)
```

```
$ python3 ops.py  
L [11, 3, 5, 7, 2]  
cinco está en L  
L[4]=9999 [11, 3, 5, 7, 9999]  
L.append(100) [11, 3, 5, 7, 9999, 100]  
L+[19, 17, 13] [11, 3, 5, 7, 9999, 100, 19, 17, 13]  
L[2:5] [5, 7, 9999]
```

Variable Alias

- Si ambos elementos son listas, el operador de asignación crea un nuevo nombre a la variable

```
1 L = [1, 2, 3]
```

```
2 C = L
```

```
3 L[0] = 99
```

```
4
```

```
5 print(L)
```

```
6 print(C)
```



Crea un alias de la lista

Importante: El operador de asignación '=' crea un alias (dos nombres para una misma variable).
Si quieres copiar una lista usa la función `.copy()`

Variable Alias

- Si ambos elementos son listas, el operador de asignación crea un nuevo nombre a la variable
- Para crear una copia usa lista.copy()

```
1 L = [1, 2, 3]
```

```
2 C = L
```

```
3 L[0] = 99
```

```
4
```

```
5 print(L)
```

```
6 print(C)
```

Reemplazar por



```
2 C = L.copy()
```

Importante: El operador de asignación '=' crea un alias (dos nombres para una misma variable).

Si quieres copiar una lista usa la función .copy()

Actividad 1

Un grupo de estudiantes desea conocer la variabilidad de los tiempos de viaje a la universidad. Para ello, se cuenta con una lista de los tiempos en minutos de cada estudiante. Se necesita una solución en Python que calcule la desviación estándar de los tiempos de viaje utilizando únicamente funciones y operaciones matemáticas básicas. La solución debe tomar en cuenta todos los tiempos de viaje de la lista.

- ¿Qué tanto varía tu tiempo de viaje a la universidad?

Día	Tiempo de viaje en minutos
1	67
2	45
3	84
s	19,553

20 minutos de variación

Día	Tiempo de viaje en minutos
1	70
2	70
3	70
s	0

Sin variación

Desviación Estándar

$$s = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2}$$

Promedio

- La desviación estándar permite calcular cuánto se aleja cada medición al promedio.

P1: ¿Cómo harías un programa que calcule la desviación estándar?

P2: ¿Y si el número de días es 1000?

Ejemplo 1

Selecciona una carta de un mazo al azar

Genera números aleatorios
dentro de un rango

```
1 from random import randrange
2 TRAJES = ['Picas', 'Diamantes', 'Treboles', 'Corazones']
3 VALORES = ['2', '3', '4', '5', '6', '7', '8', '9', '10',
4            'Jota', 'Reina', 'Rey', 'As']
5
6 valor = randrange(0, len(VALORES))
7 traje = randrange(0, len(TRAJES))
8 print(VALORES[valor], 'de', TRAJES[traje])
```

Este código en Python simula el lanzamiento de una carta de una baraja francesa generando de manera aleatoria un valor (número o figura) y un traje (picas, diamantes, tréboles o corazones) de la carta. Los valores y trajes de la baraja están definidos en dos listas. Al ejecutar el código, se imprime en pantalla el valor y traje de la carta seleccionada aleatoriamente.

Utilidades

Crear lista con valores de teclado	<pre>L = [] #lista vacía for i in range(N): v = int(input()) L.append(v)</pre>
Imprimir valores en lista (uno por uno)	<pre>for elem in L: print(elem) # alternativa for i in range(N): print(L[i])</pre>
Encontrar el máximo valor en una lista	<pre>maxi = L[0] for elem in L: if elem > maxi: maxi = elem print(maxi)</pre>
Encontrar el mínimo valor en una lista	<pre>mini = L[0] for elem in L: if elem < mini: mini = elem print(mini)</pre>

Obtener el promedio	<pre># promedio suma = 0.0 for elem in L: suma = suma + elem prom = suma/N</pre>
Copiar elementos a otra lista	<pre>L2 = [] for elem in L: L2.append(elem)</pre>
Crear nueva listas con elementos invertidos	<pre>N = len(L) R = [] for i in range(N): j = N-i-1 R.append(L[j])</pre>
Invertir elementos del arreglo	<pre>for i in range(N): temp = L[i] L[i] = L[N-i-1] L[N-i-1] = temp</pre>

Ejemplo 2

Creando un Mazo. El siguiente código en Python genera un mazo de cartas de la baraja española. Se tienen dos listas, una que contiene los trajes posibles (Picas, Diamantes, Tréboles, Corazones) y otra que contiene los valores posibles de las cartas (2 al 10, Jota, Reina, Rey, As).

```
1 from random import randrange
2 TRAJES = ['Picas', 'Diamantes', 'Treboles', 'Corazones']
3 VALORES = ['2', '3', '4', '5', '6', '7', '8', '9', '10',
4            'Jota', 'Reina', 'Rey', 'As']
5
6 mazo = []
7 for traje in TRAJES:
8     for valor in VALORES:
9         mazo.append(valor + ' de ' + traje)
10
11 print(mazo)
```

index()

Para utilizar este método, se llama al método `index()` en la lista que se desea buscar y se le pasa el valor del elemento como argumento. El método devuelve el índice de la **primera** ocurrencia del elemento en la lista.

```
# Creamos una lista de números  
numeros = [1, 3, 7, 11, 3, 8]
```

```
# Encontramos el índice de 1  
indice = numeros.index(3)
```

```
# Imprimimos el índice  
print(indice)
```

1

Actividad 2

Supongamos que se está haciendo una lista de compras en un supermercado. Se tienen dos listas: la primera lista, llamada `precios`, contiene los precios de diferentes productos que se quieren comprar, mientras que la segunda lista, llamada `cantidades`, contiene la cantidad de cada producto que se desea comprar.

```
precios = [7875, 15562, 22500, 11437, 4492]
```

```
cantidades = [3, 2, 4, 1, 5]
```

```
productos = ["manzanas", "leche", "pan", "huevos", "agua"]
```

Se pide escribir un programa en Python que calcule el costo total de la lista de compras, multiplicando el precio de cada producto por la cantidad correspondiente y sumando los resultados. Además, el programa deberá encontrar el producto más caro y el producto más barato en base a sus precios, utilizando las funciones `max()` y `min()`. Finalmente, el programa deberá imprimir el costo total de la lista de compras, el precio del producto más caro y el precio del producto más barato.

Resumen

Conceptos

- **Lista:** secuencia de elementos
- **Alias:** nuevo nombre a una variable. Si modifico el contenido en una, se modifica en la otra también.

False	await	else	import	pass
None	break	except	in	raise
True	class	finally	is	return
and	continue	for	lambda	try
as	def	from	nonlocal	while
assert	del	global	not	with
async	elif	if	or	yield

https://docs.python.org/3/reference/lexical_analysis.html

Funciones

- **len(lista):** tamaño de una lista o de un string
- **elem.copy():** crear copia de variable elem

		Built-in Functions		
abs()	delattr()	hash()	memoryview()	set()
all()	dict()	help()	min()	setattr()
any()	dir()	hex()	next()	slice()
ascii()	divmod()	id()	object()	sorted()
bin()	enumerate()	input()	oct()	staticmethod()
bool()	eval()	int()	open()	str()
breakpoint()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	

<https://docs.python.org/3/library/functions.html>