# Clase 7 – Listas y Manipulación de Datos

## Marco Teórico

Un tema más que necesitará entender antes de poder comenzar a escribir programas en serio es el tipo de datos de la lista y su primo, la tupla. Las listas y las tuplas pueden contener múltiples valores, lo que facilita la escritura de programas que manejan grandes cantidades de datos. Y dado que las listas en sí pueden contener otras listas, puede usarlas para organizar datos en estructuras jerárquicas.

Las listas permiten almacenar múltiples datos en una sola variable.

Son estructuras dinámicas y modificables.

### El tipo de datos de la lista

Una *lista* es un valor que contiene múltiples valores en una secuencia ordenada. El término *valor* de *lista* se refiere a la lista en sí (que puede almacenar en una variable o pasar a una función, al igual que cualquier otro valor), no a los valores dentro del valor de la lista. Un valor de lista se ve así: ['gato', 'murciélago', 'rata', 'elefante']. Así como los valores de la cadena utilizan comillas para marcar dónde comienza y termina la cadena, una lista comienza con un corchete de apertura y termina con un corchete de cierre, [].

Llamamos valores dentro de los *elementos* de la lista. Los artículos están separados por comas (es decir, están *delimitados* por *comas*). Por ejemplo, introduzca lo siguiente en el shell interactivo:

```
>>> [1, 2, 3]  # A list of three integers
[1, 2, 3]
>>> ['cat', 'bat', 'rat', 'elephant']  # A list of four strings
['cat', 'bat', 'rat', 'elephant']
>>> ['hello', 3.1415, True, None, 42]  # A list of several values
['hello', 3.1415, True, None, 42]
❶ >>> spam = ['cat', 'bat', 'rat', 'elephant']
>>> spam
['cat', 'bat', 'rat', 'elephant']
```

A la variable de spam ❶ se le asigna solo un valor: el valor de lista. Pero el valor de la lista en sí contiene otros valores.

Tenga en cuenta que el valor [] es una lista vacía que no contiene valores, similares a '', la cadena vacía.

## Índices

Digamos que tiene la lista ['gato', 'murciélago', 'rata', 'elefante'] almacenada en una variable llamada spam. El código de Python spam[0] evaluaría a 'cat', el código spam[1] se evaluaría a 'murciélago', y así sucesivamente. El número entero dentro de los corchetes que sigue a la lista se llama *índice*. El primer valor en la lista está en el índice 0, el segundo 1valor está en el índice 1, el tercer valor está en el índice 2, y así sucesivamente. La Figura 6-1 muestra un valor de lista asignado al spam, junto con las expresiones de índice a las que evaluarían. Tenga en cuenta que debido a que el primer índice es 0, el último índice es el tamaño de la lista menos uno. Por lo tanto, una lista de cuatro ítems tiene 3 como último índice.

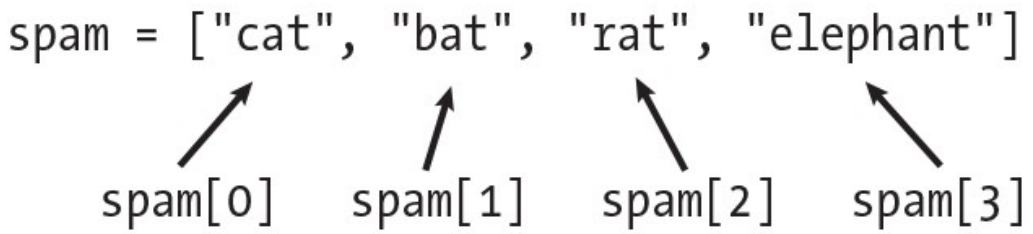


Figura 6-1: Un valor de lista almacenado en la variable spam, que muestra qué valor se refiere a cada índice a Descripción

Para un ejemplo de trabajo con índices, introduzca las siguientes expresiones en la shell interactiva. Empezamos por asignar una lista a la variable spam:

```
>>> spam = ['cat', 'bat', 'rat', 'elephant']
>>> spam[0]
'cat'
>>> spam[1]
'bat'
>>> spam[2]
'rat'
>>> spam[3]
'elephant'
>>> ['cat', 'bat', 'rat', 'elephant'][3]
'elephant'
❶ >>> 'Hello, ' + spam[0]
❷ 'Hello, cat'
>>> 'The ' + spam[1] + ' ate the ' + spam[0] + '.'
'The bat ate the cat.'
```

Observe que la expresión 'Hola, ' + spam[0] ❶ evalúa a 'Hola, ' + 'gato' porque el spam[0] evalúa a la cadena 'gato'. Esta expresión a su vez evalúa el valor de la cadena 'Hola, gato' ❷.

Python le dará un mensaje de error IndexError si utiliza un índice que excede el número de valores en el valor de su lista:

```
>> spam = ['cat', 'bat', 'rat', 'elephant']
>>> spam[10000]
Traceback (most recent call last):
  File "<python-input-0>", line 1, in <module>
    spam[10000]
IndexError: list index out of range
```

Las listas también pueden contener otros valores de lista. Puede acceder a los valores de estas listas utilizando múltiples índices, como así:

```
>>> spam = [['cat', 'bat'], [10, 20, 30, 40, 50]]
>>> spam[0]
['cat', 'bat']
>>> spam[0][1]
'bat'
>>> spam[1][4]
50
```

El primer índice dicta qué valor de lista usar, y el segundo indica el valor dentro del valor de lista. Por ejemplo, spam[0][1] imprime 'bat', el segundo valor de la primera lista.

# Características principales

- Se definen con corchetes [].
- Permiten agregar y eliminar elementos.
- Admiten distintos tipos de datos.

### Ejemplo

alumnos = ["Ana", "Luis", "Pedro"]

# Operaciones fundamentales

### Agregar elementos

alumnos.append("María")

Para agregar nuevos valores a una lista, utilice los métodos append(). El método append() añade el argumento al final de la lista:

```
>>> spam = ['cat', 'dog', 'bat']
>>> spam.append('moose')
>>> spam
['cat', 'dog', 'bat', 'moose']
```

## Eliminar elementos

alumnos.remove("Luis")

El método remove() acepta un valor para eliminar de la lista en la que se llama:

```
>>> spam = ['cat', 'bat', 'rat', 'elephant']
>>> spam.remove('bat')
>>> spam
['cat', 'rat', 'elephant']
```

Intentar eliminar un valor que no existe en la lista resultará en un error de ValueError. Por ejemplo, introduzca lo siguiente en el shell interactivo y observe el error que muestra:

```
>>> spam = ['cat', 'bat', 'rat', 'elephant']
>>> spam.remove('chicken')
Traceback (most recent call last):
  File "<python-input-0>", line 1, in <module>
    spam.remove('chicken')
ValueError: list.remove(x): x not in list
```

Si el valor aparece varias veces en la lista, el método eliminará solo la primera instancia de la misma:

```
>>> spam = ['cat', 'bat', 'rat', 'cat', 'hat', 'cat']
>>> spam.remove('cat')
>>> spam
['bat', 'rat', 'cat', 'hat', 'cat']
```

## Clasificación de valores

Puede ordenar listas de valores numéricos o listas de cadenas con el método sort(). Por ejemplo, introduzca lo siguiente en el shell interactivo:

```
>>> spam = [2, 5, 3.14, 1, -7]
>>> spam.sort()
>>> spam
[-7, 1, 2, 3.14, 5]
>>> spam = ['Ants', 'Cats', 'Dogs', 'Badgers', 'Elephants']
>>> spam.sort()
>>> spam
['Ants', 'Badgers', 'Cats', 'Dogs', 'Elephants']
```

El método devuelve los números en orden numérico y las cadenas en orden alfabético.

## Recorrer listas

```
for alumno in alumnos:
    print(alumno)
```

## La función len()

La función len() devolverá el número de valores en un valor de lista pasado a él. Por ejemplo, introduzca lo siguiente en el shell interactivo

```
>>> spam = ['cat', 'dog', 'moose']
>>> len(spam)
3
```

Este comportamiento es similar a la forma en que la función cuenta el número de caracteres en un valor de cadena.

## Actividades Prácticas

## Actividad 1 – Lista de compras

Crear una lista de productos y mostrarla completa.

## Actividad 2 – Registro de alumnos

Permitir cargar nombres hasta que el usuario escriba “fin”.

## Actividad 3 – Estadística básica

Ingresar 10 números y mostrar:

- Mayor
- Menor
- Promedio

## Actividad Integradora

Sistema de gestión de tareas:

- Agregar tareas.
- Mostrar tareas.
- Eliminar tareas realizadas.