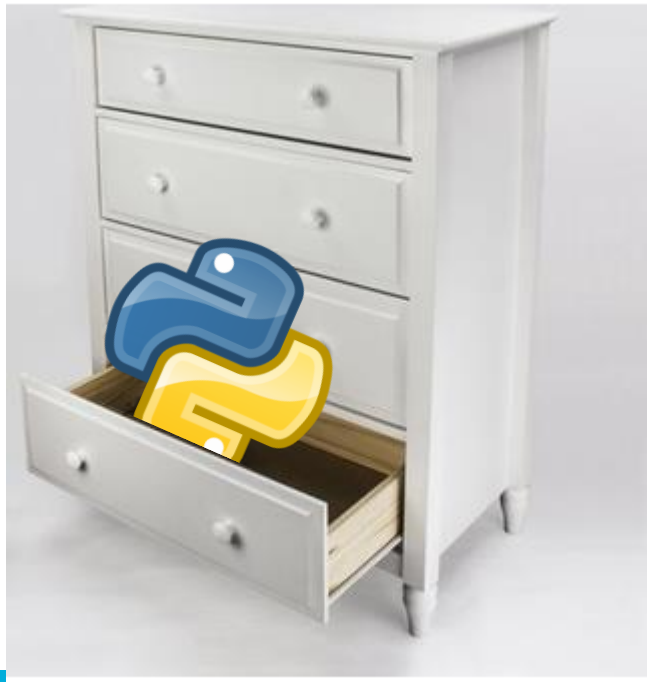
Python (V): Listas

Programación (tics 100)
Semestre 01/2021

¿Qué veremos hoy?

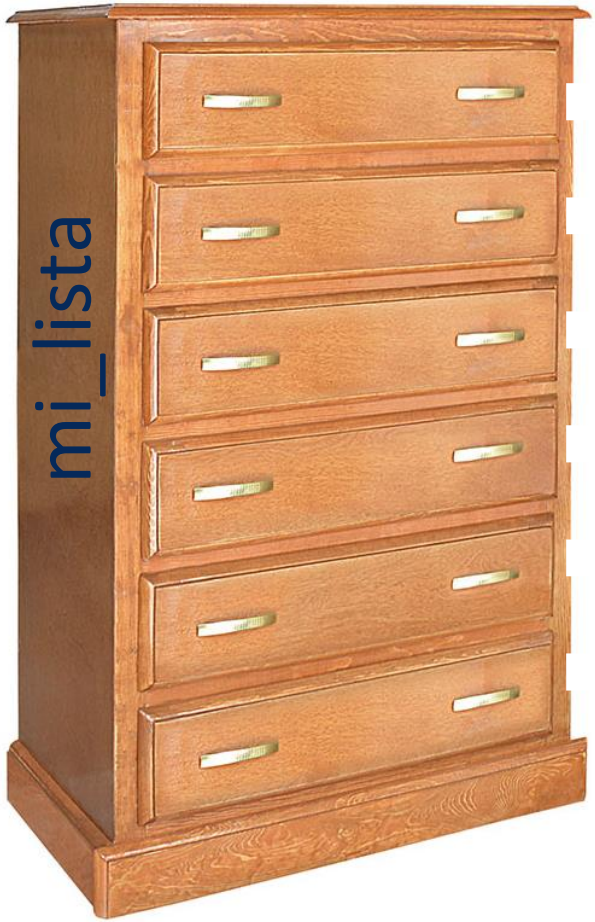
Listas: una estructura de datos en Python



¿Qué es una lista?

Una lista es una variable que puede guardar muchas cosas al mismo tiempo dentro.

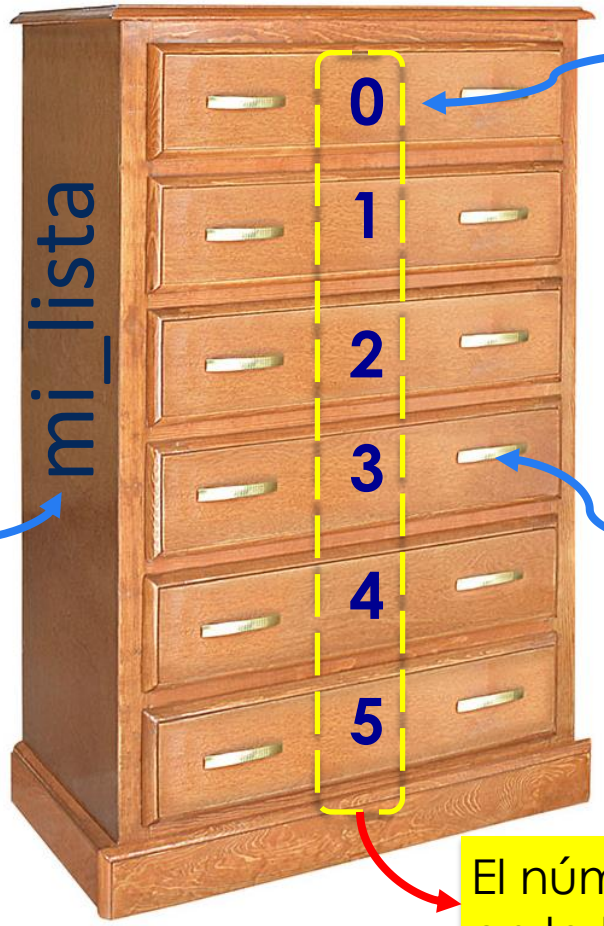
Usamos como analogía una cajonera que puede contener variadas cosas.



- Calcetines
- Poleras
- Pijamas
- Cables
- Herramientas
- Documentos

¿Qué es una lista?

Algunas características de las listas son:



Nombre de variable

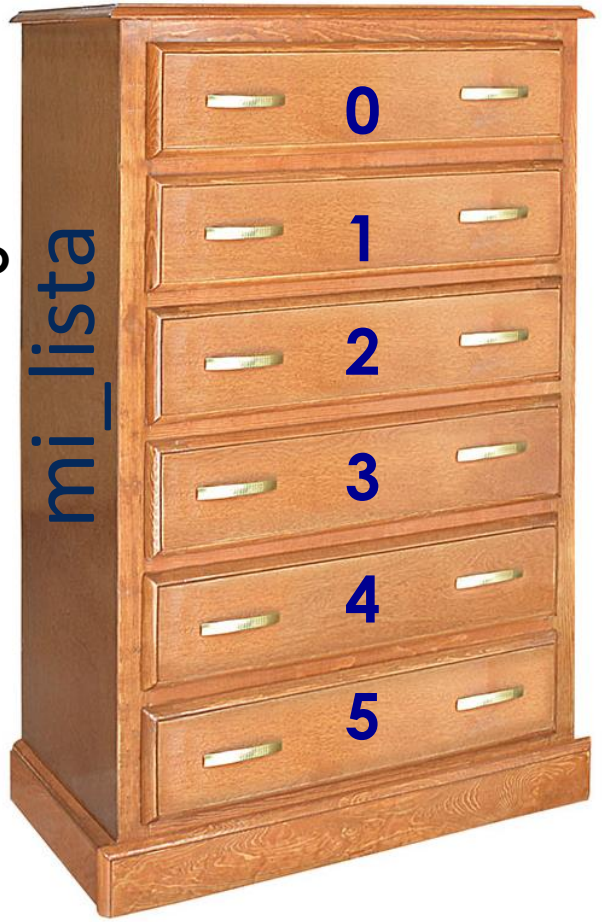
Cada espacio se identifica con un número que **comienza en 0**. Es decir el primer elemento es el lugar 0 de la lista.

Puedes guardar lo que quieras en cada espacio. Números, palabras o frases

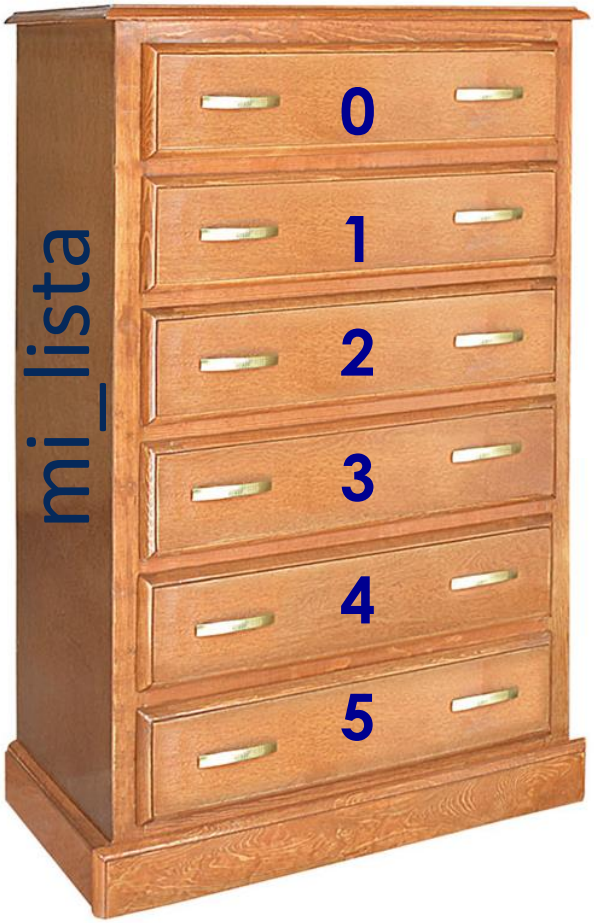
El número que identifica la posición en la lista lo llamaremos **índice**

¿Para qué sirve una lista?

- Para **ahorrar código** teniendo varios valores en una sola variable.
- **Relacionar valores a un concepto** en particular por ejemplo, el listado de las notas.
- **Ordenar** valores
- **Trabajar con múltiples valores** de forma **ordenada** y **dinámica**.



¿Cómo se ve en Python?




```
mi_lista = [45,5,3,"hola","Otra cosa"]
```

Leyendo valores de una lista

Los valores que se asignan a una lista, van separados por comas y entre corchetes.

Python

```
mi_lista = [45,5,3,"hola","Otra cosa"]
print(mi_lista[0])
```

```
45
```

El formato es el siguiente:

```
nombre_lista = [elementos]
```

Para obtener un elemento debo indicar su índice:

```
print(mi_lista[0])
```

Python

```
mi_lista = [45,5,3,"hola","Otra cosa"]
print(mi_lista[3])
```

```
hola
```

Largo de una lista

El largo de una lista se obtiene con la función len( ).

Python

```
mi_lista = [45,5,3,"hola","Otra cosa"]
largo_lista = len(mi_lista)
print(largo_lista)
```

```
5
```

El formato es el siguiente:

```
largo_lista = len(mi_lista)
```

El resultado es un número entero

Listas y ciclo for

Para mostrar todos los valores aprovechamos la estructura del ciclo for, de tal forma de escribir el código necesario y así recorrer toda la lista.

```
lista = [45,5,3,"hola","Otra cosa"]
for elemento in lista:
    print(elemento)
```

```
45
5
3
Hola
Otra cosa
```

Esta forma recorre todos los elementos de la lista y permite realizar acciones con cada uno. El formato es el siguiente:

```
for elemento in lista:
    <<acciones>>
```

- **elemento** es una variable que guardará cada elemento de la lista
- **lista** es el nombre de la lista
- <<acciones>> es código que se ejecuta para cada elemento de la lista

Otra manera de recorrer la lista

Otra forma de observar cada valor de la lista es usando el índice como identificador.

```
lista = [45,5,3,"hola","Otra cosa"]
for i in range (0,len(lista)):
    print(lista[i])
```

```
45
5
3
Hola
Otra cosa
```

Dado que el índice está ordenado y sabemos el largo de la lista, fabricamos el rango en donde la variable i debe moverse. El formato es el siguiente:

```
for i in range (0,len(lista)):
        <<acciones>>
```

- **i** es una variable que avanza de uno en uno
- **len()** es una función que nos entrega el largo de la lista
- <<acciones>> es código que se ejecuta para cada elemento de la lista

Ingreso dinámico de datos

Esta bien ingresar datos por código, pero ¿cómo podemos hacer para que el usuario ingrese los datos?

Usamos los ciclos para hacerlo de forma dinámica porque no sabemos cuántos datos ingresarán.

lista vacía nos sirve para decirle al computador que usaremos una lista, que queda vacía esperando a que ingresemos información

```
print("ingrese el número de elementos: ")
num = int(input())

lista = []
for i in range (0,num):
    print("ingrese es elemento",i)
    elem = int(input())
    lista.append(elem)
```

.append(<contenido>) es una función que nos permite agregar un valor al final de la lista.

Ingreso dinámico de datos

Esta bien ingresar datos por código, pero ¿cómo podemos hacer para que el usuario ingrese los datos?

Usamos los ciclos para hacerlo de forma dinámica porque no sabemos cuántos datos ingresarán.

```
print("ingrese el número de elementos: ")
num = int(input())

lista = []
for i in range (0,num):
    print("ingrese el elemento",i)
    elem = int(input())
    lista.append(elem)
```

recorremos el listado posición por posición

```
print ("la lista es:")
for i in range (0,len(lista)):
    print(lista[i])
```

Ingreso dinámico de datos

Python

```
print("ingrese el num. de elementos: ")
num = int(input())

lista = []
for i in range (0,num):
    print("ingrese numero",i)
    lista.append(input())

print ("la lista es:")
for i in range (0,len(lista)):
    print(lista[i])
```

```
ingrese el num. de elementos
5
ingrese numero 0
4
ingrese numero 1
56
ingrese numero 2
786
ingrese numero 3
54
ingrese numero 4
5678
la lista es:
4
56
786
54
5678
```

Probemos

Llene una lista con números enteros y calcule la suma de ellos

```
lista = [3,56,5,38,2,3,765]
suma = 0
for elemento in lista:
    suma = suma + elemento
print(suma)
```

```
lista = [3,56,5,38,2,3,765]
suma = 0
for i in range (0,len(lista)):
    suma = suma + lista[i]
print(suma)
```

```
872
```

Importante:
La lista debe contener solo valores numéricos, de lo contrario no podrá sumarlos.
Ejemplo: `lista = [1,2,4, "Hola"]`

Probemos

Dada una lista con números enteros calcule, cuántos números pares hay en la misma

```
lista = [3,56,5,38,2,3,765]
cont=0
for element in lista:
    if (element % 2 == 0):
        cont = cont + 1
print("no. de pares es", cont)
```

```
lista = [3,56,5,38,2,3,765]
cont=0
for i in range (0,len(lista)):
    if (lista [i] % 2 == 0):
        cont = cont + 1
print("no. de pares es", cont)
```

```
no. de pares es 3
```

Probemos

Dada una lista con números enteros, encuentre el número menor de esta

```
lista = [3,56,5,38,2,3,765]
menor = lista[0]
for element in lista:
    if (element < menor):
        menor = element
print("menor elemento: ", menor)
```

```
lista = [3,56,5,38,2,3,765]
menor = lista[0]
for i in range (0,len(lista)):
    if (lista[i] < menor):
        menor = lista[i]
print("menor elemento: ", menor)
```

```
menor elemento: 2
```

¡Ahora tú!

Tiempo : 10 minutos

Genere una lista con 10 números enteros ingresados por el usuario y calcule su promedio

```
ingrese número de elementos:
4
ingrese numero 0
3
ingrese numero 1
4
ingrese numero 2
3
ingrese numero 3
4
la lista es:
3
4
3
4
promedio es: 3.5
```

¡Ahora tú!

Tiempo : 10 minutos

Genere una lista con 10 números enteros ingresados por el usuario y calcule su promedio

```
print("ingrese numero de elementos ")
num = int(input())
lista = []

for i in range (0,num):
    print("ingrese numero",i)
    lista.append(input())

print ("la lista es:")
for i in range (0,len(lista)):
    print(lista[i])

suma=0
for elemento in lista:
    suma = suma + elemento

promedio = suma/len(lista)
print ("promedio es:", promedio)
```

```
ingrese número de elementos:
4
ingrese numero 0
3
ingrese numero 1
4
ingrese numero 2
3
ingrese numero 3
4
la lista es:
3
4
3
4
promedio es: 3.5
```

¡Ahora tú!

Tiempo : 15 minutos

Crea un programa que pida al usuario cuántos números primos requiere calcular, y que calcule y grabe en una lista los primeros n números primos. Al final imprima el contenido de la lista

```
primos_buscados = int(input("Cuántos
números primos quieres calcular?\n"))
numero_probado = 1
lista_primos= []
while len(lista_primos) < primos_buscados:
  divisores = 0
  for i in range(numero_probado):
    if numero_probado % (i+1) == 0:
        divisores = divisores +1
  if divisores <= 2:
    lista_primos.append(numero_probado)
  numero_probado = numero_probado +1

for i in range(len(lista_primos)):
  print(f"{lista_primos[i]} es primo")
```

```
Cuántos números primos quieres calcular?
5
1 es primo
2 es primo
3 es primo
5 es primo
7 es primo
```

¡Ahora tú!

Tiempo : 15 minutos

Crea un programa que permita ordenar una lista de nombres ocupando un for dentro de un for que compare cada uno de los elementos de la lista con los elementos que le siguen. Si la primera letra del primer elemento es mayor que la primera letra del otro elemento se invierten las posiciones…Seguimos hasta que se haya recorrida la lista completa.

```
comparing Pedro with Rene
comparing Pedro with Francisco
swapping Pedro with Francisco
comparing Francisco with Andrés
swapping Francisco with Andrés
comparing Rene with Pedro
swapping Rene with Pedro
comparing Pedro with Francisco
swapping Pedro with Francisco
comparing Rene with Pedro
swapping Rene with Pedro
```

¡Ahora tú!

Tiempo : 15 minutos

Crea un programa que permita ordenar una lista de nombres ocupando un for dentro de un for que compare cada uno de los elementos de la lista con los elementos que le siguen. Si la primera letra del primer elemento es mayor que la primera letra del otro elemento se invierten las posiciones…Seguimos hasta que se haya recorrida la lista completa. Asumimos que comparamos sólamente la primera letra de los elementos.

```
a = ["Pedro","Rene","Francisco","Andrés"]

for index1 in range(len(a)):
    elemento1=a[index1]
    for index2 in range(index1+1,len(a)):
        elemento2 = a[index2]
        print(f"comparing {elemento1} with {elemento2}")
        if elemento1[0]>elemento2[0]:
            print(f"swapping {elemento1} with
{elemento2}")
            a[index1] = elemento2
            a[index2] = elemento1
            elemento1 = a[index1]
            elemento2 = a[index2]
print(a)
```

```
comparing Pedro with Rene
comparing Pedro with Francisco
swapping Pedro with Francisco
comparing Francisco with Andrés
swapping Francisco with Andrés
comparing Rene with Pedro
swapping Rene with Pedro
comparing Pedro with Francisco
swapping Pedro with Francisco
comparing Rene with Pedro
swapping Rene with Pedro
```