

Clase Número: 6

Estructura de datos tipo lista (primera parte)

Hasta ahora hemos trabajado con variables que permiten almacenar un único valor:

```
edad=12
altura=1.79
nombre="juan"
```

En Python existe un tipo de variable que permite almacenar una colección de datos y luego acceder por medio de un subíndice (similar a los string)

Creación de la lista por asignación

Para crear una lista por asignación debemos indicar sus elementos encerrados entre corchetes y separados por coma.

```
lista1=[10, 5, 3] # lista de enteros
lista2=[1.78, 2.66, 1.55, 89,4] # lista de valores float
lista3=["lunes", "martes", "miercoles"] # lista de string
lista4=["juan", 45, 1.92] # lista con elementos de distinto tipo
```

Si queremos conocer la cantidad de elementos de una lista podemos llamar a la función len:

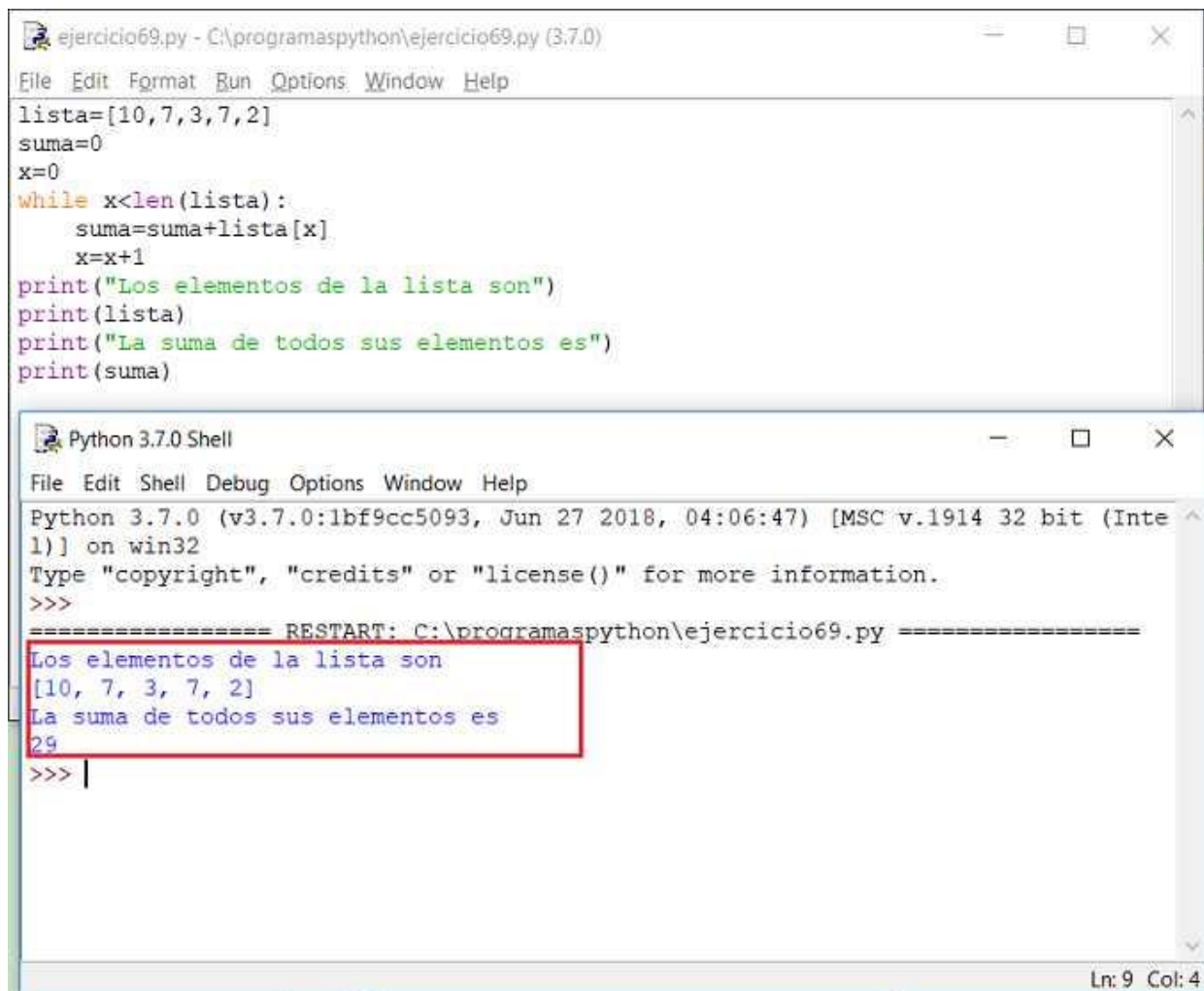
```
lista1=[10, 5, 3] # lista de enteros
print(len(lista1)) # imprime un 3
```

Problema 1:

Definir una lista que almacene 5 enteros. Sumar todos sus elementos y mostrar dicha suma.

Programa:

```
lista=[10,7,3,7,2]
suma=0
x=0
while x<len(lista):
    suma=suma+lista[x]
    x=x+1
print("Los elementos de la lista son")
print(lista)
print("La suma de todos sus elementos es")
print(suma)
```

The image shows a screenshot of a Python IDE. The top window, titled 'ejercicio69.py - C:\programaspython\ejercicio69.py (3.7.0)', contains the following Python code:

```
lista=[10,7,3,7,2]
suma=0
x=0
while x<len(lista):
    suma=suma+lista[x]
    x=x+1
print("Los elementos de la lista son")
print(lista)
print("La suma de todos sus elementos es")
print(suma)
```

The bottom window, titled 'Python 3.7.0 Shell', shows the execution output. It includes the standard Python 3.7.0 startup text and a 'RESTART' message. The output of the script is displayed in a red-bordered box:

```
Los elementos de la lista son
[10, 7, 3, 7, 2]
La suma de todos sus elementos es
29
>>> |
```

Primero definimos una lista por asignación con 5 elementos:

```
lista=[10,7,3,7,2]
```

Definimos un acumulador para sumar los elementos de la lista y un contador para indicar que posición de la lista accedemos:

```
suma=0
x=0
```

Mediante un ciclo while recorremos y sumamos cada elementos de la lista:

```
while x<len(lista):
    suma=suma+lista[x]
    x=x+1
```

Cuando llamamos a la función print pasando como dato una lista luego se muestra en pantalla todos los elementos de la lista entre corches y separados por coma tal cual como lo definimos:

```
print("Los elementos de la lista son")
print(lista)
```

Finalmente mostramos el acumulador:

```
print("La suma de todos sus elementos es")
```

```
print(suma)
```

Problema 2:

Definir una lista por asignación que almacene los nombres de los primeros cuatro meses de año. Mostrar el primer y último elemento de la lista solamente.

Programa:

```
meses=["enero", "febrero", "marzo", "abril"]
print(meses[0]) # se muestra enero
print(meses[3]) # se muestra abril
```

Como queremos imprimir solo el primer y último elemento de la lista indicamos entre corchetes la posición de la lista del cual queremos rescatar el valor.

Si llamamos a print y pasamos solo el nombre de la lista luego se nos muestra todos los elementos:

```
print(meses) # se muestra ["enero", "febrero", "marzo", "abril"]
```

Problema 3:

Definir una lista por asignación que almacene en la primer componente el nombre de un alumno y en las dos siguientes sus notas. Imprimir luego el nombre y el promedio de las dos notas.

Programa:

```
lista=["ana", 7, 9]
print("Nombre del alumno:")
print(lista[0])
promedio=(lista[1]+lista[2])/2
print("Promedio de sus dos notas:")
print(promedio)
```

Como vemos en este problema los elementos de una lista pueden ser de distinto tipo, aquí tenemos el primer elemento de tipo string y los dos siguientes de tipo int.

Recordemos que el operador // se utiliza para dividir dos valores y retornar solo la parte entera.

Problema propuesto

- 1 - Definir por asignación una lista con 8 elementos enteros. Contar cuantos de dichos valores almacenan un valor superior a 100.
- 2 - Definir una lista por asignación con 5 enteros. Mostrar por pantalla solo los elementos con valor iguales o superiores a 7.
- 3 - Definir una lista que almacene por asignación los nombres de 5 personas. Contar cuantos de esos nombres tienen 5 o más caracteres.

Solución a los problemas

```
lista=[1000, 6000, 400, 23, 130, 400, 60, 2000]
cantidad=0
x=0
while x<len(lista):
    if lista[x]>100:
        cantidad=cantidad+1
    x=x+1

print("La lista esta constituida por los elementos:")
print(lista)
print("La cantidad de valores mayores a 100 en la lista son:")
print(cantidad)
```

ejercicio73.py

```
lista=[8,1,9,2,10]
x=0
print("Elementos de la lista con valores iguales o superiores a 7")
while x<len(lista):
    if lista[x]>=7:
        print(lista[x])
    x=x+1
```

ejercicio74.py

```
nombres=["juan", "ana", "marcos", "carlos", "luis"]
cantidad=0
x=0
while x<len(nombres):
    if len(nombres[x])>=5:
        cantidad=cantidad+1
    x=x+1

print("Todos los nombres son")
print(nombres)
print("Cantidad de nombres con 5 o mas caracteres")
print(cantidad)
```

Carga por teclado de sus elementos

Una lista en Python es una estructura mutable (es decir puede ir cambiando durante la ejecución del programa)

Hemos visto que podemos definir una lista por asignación indicando entre corchetes los valores a almacenar:

```
lista=[10, 20, 40]
```

Una lista luego de definida podemos agregarle nuevos elementos a la colección. La primera forma que veremos para que nuestra lista crezca es utilizar el método `append` que tiene la lista y pasar como parámetro el nuevo elemento:

```
lista=[10, 20, 30]
print(len(lista))      # imprime un 3
lista.append(100)
print(len(lista))      # imprime un 4
print(lista[0])         # imprime un 10
print(lista[3])         # imprime un 100
```

Definimos una lista con tres elementos:

```
lista=[10, 20, 30]
```

Imprimimos la cantidad de elementos que tiene la lista, en nuestro caso lo definimos con 3:

```
print(len(lista))      # imprime un 3
```

Agregamos una nuevo elemento al final de la lista llamando al método `append`:

```
lista.append(100)
```

Si llamamos nuevamente a la función `len` y le pasamos el nombre de nuestra lista ahora retorna un 4:

```
print(len(lista))      # imprime un 4
```

Imprimimos ahora el primer y cuarto elemento de la lista (recordar que se numeran a partir de cero):

```
print(lista[0])         # imprime un 10
print(lista[3])         # imprime un 100
```

Problema :

Definir una lista vacía y luego solicitar la carga de 5 enteros por teclado y añadirlos a la lista. Imprimir la lista generada.

Programa:

```
#definimos una lista vacia
lista=[]
#disponemos un ciclo de 5 vueltas
for x in range(5):
```

```

    valor=int(input("Ingrese un valor entero:"))
    lista.append(valor)

#imprimimos la lista
print(lista)

```

El algoritmo propuesto crea primero una lista vacía (debemos asignar los corchetes de apertura y cerrado sin contenido):

```
lista=[]
```

Luego mediante un for (podemos utilizar un while si queremos) solicitamos en forma sucesiva la carga de un entero por teclado y procedemos a agregarlo al final de la lista llamando al método append:

```

for x in range(5):
    valor=int(input("Ingrese un valor entero:"))
    lista.append(valor)

```

Finalmente mostramos los elementos de la lista creada:

```
print(lista)
```

Problema:

Realizar la carga de valores enteros por teclado, almacenarlos en una lista. Finalizar la carga de enteros al ingresar el cero. Mostrar finalmente el tamaño de la lista.

Programa:

```

lista=[]
valor=int(input("Ingresar valor (0 para finalizar):"))
while valor!=0:
    lista.append(valor)
    valor=int(input("Ingresar valor (0 para finalizar):"))

print("Tamano de la lista:")
print(len(lista))

```

En este problema la lista crecerá hasta que el operador ingrese el valor cero. La carga del primer valor se efectúa antes del ciclo while ya que la condición depende del valor ingresado:

```
valor=int(input("Ingresar valor (0 para finalizar):"))
```

Luego dentro del ciclo while procedemos a agregar al final de la lista el valor ingresado y solicitar la carga del siguiente valor:

```

while valor!=0:
    lista.append(valor)
    valor=int(input("Ingresar valor (0 para finalizar):"))

```

Cuando salimos del ciclo repetitivo procedemos a obtener el tamaño de la lista mediante la función len:

```
print(len(lista))
```

Problema propuesto

1 - Almacenar en una lista los sueldos (valores float) de 5 operarios. Imprimir la lista y el promedio de sueldos.

2 - Cargar por teclado y almacenar en una lista las alturas de 5 personas (valores float)

Obtener el promedio de las mismas. Contar cuántas personas son más altas que el promedio y cuántas más bajas.

3 - Una empresa tiene dos turnos (mañana y tarde) en los que trabajan 8 empleados (4 por la mañana y 4 por la tarde) Confeccionar un programa que permita almacenar los sueldos de los empleados agrupados en dos listas. Imprimir las dos listas de sueldos.

Solución a los problemas

```
suealdos=[]
suma=0
for x in range(5):
    valor=float(input("Ingrese el sueldo del operario:"))
    suealdos.append(valor)
    suma=suma+valor
print("Lista de suealdos")
print(suealdos)
promedio=suma/5
print("Promedio de suealdos")
print(promedio)
```

```
alturas=[]
suma=0
for x in range(5):
    valor=float(input("Ingrese la altura:"))
    alturas.append(valor)
    suma=suma+valor

print("Las alturas ingresadas son")
print(alturas)
promedio=suma/5
print("El promedio de las alturas es")
print(promedio)
```

```
altas=0
bajas=0
for x in range(5):
    if alturas[x]>promedio:
        altas=altas+1
    else:
        if alturas[x]<promedio:
            bajas=bajas+1

print("La cantidad de personas mas bajas al promedio es")
print(bajas)
print("La cantidad de personas mas altas al promedio es")
print(altas)
```

```
suealdosman=[]
print("Suealdos turno manana")
for x in range(4):
    valor=float(input("Ingrese sueldo:"))
    suealdosman.append(valor)

suealdostar=[]
print("Suealdos turno tarde")
for x in range(4):
    valor=float(input("Ingrese sueldo:"))
    suealdostar.append(valor)

print("Turno manana")
print(suealdosman)
print("Turno tarde")
print(suealdostar)
```

Mayor y menor elemento de una lista

Es una actividad muy común la búsqueda del mayor y menor elemento de una lista.

Es necesario que la lista tenga valores del mismo tipo por ejemplo enteros. Pueden ser de tipo cadenas de caracteres y se busque cual es mayor o menor alfabéticamente, pero no podemos buscar el mayor o menor si la lista tiene enteros y cadenas de caracteres al mismo tiempo.

Problema :

Crear y cargar una lista con 5 enteros. Implementar un algoritmo que identifique el mayor valor de la lista.

Programa:

```
lista=[]
for x in range(5):
    valor=int(input("Ingrese valor:"))
    lista.append(valor)

mayor=lista[0]
for x in range(1,5):
    if lista[x]>mayor:
        mayor=lista[x]

print("Lista completa")
print(lista)
print("Mayor de la lista")
print(mayor)
```

Primero procedemos a cargar por teclado los 5 valores en la lista:

```
for x in range(5):
    valor=int(input("Ingrese valor:"))
    lista.append(valor)
```

Para identificar el mayor de una lista primero tomamos como referencia el primer elemento, considerando a este en principio como el mayor de la lista:

```
mayor=lista[0]
```

Seguidamente disponemos un for que se repita 4 veces esto debido a que no debemos controlar el primer elemento de la lista (recordar que la primer vuelta del for x toma el valor 1, luego el 2 y así sucesivamente hasta el 4):

```
for x in range(1,5):
```

Dentro del for mediante una estructura condicional verificamos si el elemento de la posición x de la lista es mayor al que hemos considerado hasta este momento como mayor:

```
    if lista[x]>mayor:
        mayor=lista[x]
```

Como vemos en las dos líneas anteriores si el if se verifica verdadero actualizamos la variable mayor con el elemento de la lista que estamos analizando.

Cuando finaliza el for procedemos a mostrar el contenido de la variable "mayor" que tiene almacenado el mayor elemento de la lista:

```
print("Mayor de la lista")
print(mayor)
```

Problema :

Crear y cargar una lista con 5 enteros por teclado. Implementar un algoritmo que identifique el menor valor de la lista y la posición donde se encuentra.

Programa:

```
lista=[]
for x in range(5):
    valor=int(input("Ingrese valor:"))
    lista.append(valor)

menor=lista[0]
posicion=0
for x in range(1,5):
    if lista[x]<menor:
        menor=lista[x]
        posicion=x

print("Lista completa")
print(lista)
print("Menor de la lista")
print(menor)
print("Posicion del menor en la lista")
print(posicion)
```

Iniciamos una lista vacía y cargamos 5 elementos enteros:

```
lista=[]
for x in range(5):
    valor=int(input("Ingrese valor:"))
    lista.append(valor)
```

Para identificar el menor y la posición de dicho valor en la lista definimos dos variables. La primera le cargamos el primer elemento de la lista, que es el que consideramos como menor hasta este momento:

```
menor=lista[0]
```

Por otro lado la segunda variable almacena un cero que es la posición donde se encuentra el menor hasta este momento:

```
posicion=0
```

Seguidamente disponemos un for que se repita 4 veces (que son la cantidad de elementos que nos faltan analizar de la lista):

```
for x in range(1,5):
```

Dentro del for mediante un if verificamos si el elemento que estamos analizando de la lista es menor a la variable "menor":

```
    if lista[x]<menor:
```

En caso que sea menor a la que hemos considerado "menor" hasta este momento procedemos a actualizar la variable "menor" con el nuevo valor y también actualizamos la variable "posicion" con el valor del contador del for que nos indica que posición de la lista estamos analizando:

```
        menor=lista[x]  
        posicion=x
```

Cuando salimos del for mostramos la lista completa, el menor de la lista y la posición que tiene en la lista dicho menor:

```
print("Lista completa")  
print(lista)  
print("Menor de la lista")  
print(menor)  
print("Posicion del menor en la lista")  
print(posicion)
```

Problema propuesto

1 - Ingresar por teclado los nombres de 5 personas y almacenarlos en una lista. Mostrar el nombre de persona menor en orden alfabético.

2 - Cargar una lista con 5 elementos enteros. Imprimir el mayor y un mensaje si se repite dentro de la lista (es decir si dicho valor se encuentra en 2 o más posiciones en la lista)

Solución a los problemas

```
nombres=[]
for x in range(5):
    nom=input("Ingrese nombre de persona:")
    nombres.append(nom)

nombremenor=nombres[0]
for x in range(1,5):
    if nombres[x]<nombremenor:
        nombremenor=nombres[x]

print("La lista completa de nombres ingresado son")
print(nombres)
print("El nombre menor en orden alfabetico es:")
print(nombremenor)
```

```
lista=[]
for x in range(5):
    valor=int(input("Ingrese valor:"))
    lista.append(valor)

mayor=lista[0]
for x in range(1,5):
    if lista[x]>mayor:
        mayor=lista[x]

print("Lista completa")
print(lista)
print("Mayor de la lista")
print(mayor)

# contamos cuantas veces se encuentra el mayor en la lista
cantidad=0
for x in range(5):
    if lista[x]==mayor:
        cantidad=cantidad+1
if cantidad>1:
    print("El mayor se repite en la lista")
```

Lista paralelas

Podemos decir que dos listas son paralelas cuando hay una relación entre las componentes de igual subíndice (misma posición) de una lista y otra.

<i>nombres</i>	Juan	Ana	Marcos	Pablo	Laura
<i>edades</i>	12	21	27	14	21

Si tenemos dos listas que ya hemos inicializado con 5 elementos cada una. En una se almacenan los nombres de personas en la otra las edades de dichas personas.

Decimos que la lista nombres es paralela a la lista edades si en la componente 0 de cada lista se almacena información relacionada a una persona (Juan - 12 años)

Es decir hay una relación entre cada componente de las dos listas.

Esta relación la conoce únicamente el programador y se hace para facilitar el desarrollo de algoritmos que procesen los datos almacenados en las estructuras de datos.

Problema:

Desarrollar un programa que permita cargar 5 nombres de personas y sus edades respectivas. Luego de realizar la carga por teclado de todos los datos imprimir los nombres de las personas mayores de edad (mayores o iguales a 18 años)

Programa:

```
nombres=[]
edades=[]
for x in range(5):
    nom=input("Ingrese el nombre de la persona:")
    nombres.append(nom)
    ed=int(input("Ingrese la edad de dicha persona:"))
    edades.append(ed)

print("Nombre de las personas mayores de edad:")
for x in range(5):
    if edades[x]>=18:
        print(nombres[x])
```

Definimos dos listas para almacenar los nombres y las edades de las personas respectivamente:

```
nombres=[]
edades=[]
```

Mediante un for cargamos en forma simultanea un elemento de cada lista, es decir un nombre de persona y la edad de dicha persona:

```
for x in range(5):
    nom=input("Ingrese el nombre de la persona:")
    nombres.append(nom)
    ed=int(input("Ingrese la edad de dicha persona:"))
    edades.append(ed)
```

Para imprimir los nombres de la personas mayores de edad procedemos a analizar dentro de un for y mediante un if cada una de las edades almacenadas en la lista "edades", en el caso que su valor sea mayor o igual a 18 mostramos el elemento de la lista nombres de la misma posición:

```
for x in range(5):  
    if edades[x]>=18:  
        print(nombres[x])
```

Problema propuesto

1 - Crear y cargar dos listas con los nombres de 5 productos en una y sus respectivos precios en otra. Definir dos listas paralelas. Mostrar cuantos productos tienen un precio mayor al primer producto ingresado.

2 - En un curso de 4 alumnos se registraron las notas de sus exámenes y se deben procesar de acuerdo a lo siguiente:

- a) Ingresar nombre y nota de cada alumno (almacenar los datos en dos listas paralelas)
- b) Realizar un listado que muestre los nombres, notas y condición del alumno. En la condición, colocar "Muy Bueno" si la nota es mayor o igual a 8, "Bueno" si la nota está entre 4 y 7, y colocar "Insuficiente" si la nota es inferior a 4.
- c) Imprimir cuantos alumnos tienen la leyenda "Muy Bueno".

3 - Realizar un programa que pida la carga de dos listas numéricas enteras de 4 elementos cada una. Generar una tercer lista que surja de la suma de los elementos de la misma posición de cada lista. Mostrar esta tercer lista.

Solución a los problemas

```
productos=[]
precios=[]
for x in range(5):
    nom=input("Ingrese el nombre del producto:")
    productos.append(nom)
    pre=int(input("Ingrese el precio de dicho producto:"))
    precios.append(pre)

cantidad=0
for x in range(1,5):
    if precios[x]>precios[0]:
        cantidad=cantidad+1

print("Cantidad de productos con un precio mayor al primer producto
ingresado")
print(cantidad)
```

```
nombres=[]
notas=[]
for x in range(4):
    nom=input("Ingrese nombre del alumno:")
    nombres.append(nom)
    no=int(input("Ingrese la nota de dicho alumno:"))
    notas.append(no)

cantidad=0
for x in range(4):
    print(nombres[x])
    print(notas[x])
    if notas[x]>=8:
        print("Muy Bueno")
        cantidad=cantidad+1
    else:
        if notas[x]>=4:
            print("Bueno")
        else:
            print("Insuficiente")

print("La cantidad de alumnos muy buenos son")
print(cantidad)
```

```
lista1=[]
print("Carga de la primer lista")
for x in range(4):
    valor=int(input("Ingrese valor:"))
    lista1.append(valor)

lista2=[]
print("Carga de la segunda lista")
for x in range(4):
    valor=int(input("Ingrese valor:"))
    lista2.append(valor)

listasuma=[]
for x in range(4):
    suma=lista1[x]+lista2[x]
    listasuma.append(suma)

print("Lista resultante:")
```



```
print(listasuma)
```