
LISTAS

Vamos a crear una variable llamada numeros; se le asigna no solo un número, sino que se llena con una lista que consta de cinco valores (nota: la lista comienza con un corchete abierto y termina con un corchete cerrado ; el espacio entre los corchetes es llenado con cinco números separados por comas).

```
numeros = [10, 5, 7, 2, 1]
print("lista de numeros : ",end=" ")
print(numeros)
```

```
lista de numeros :  [10, 5, 7, 2, 1]
```

Digamos lo mismo utilizando una terminología adecuada: numeros es una lista que consta de cinco valores, todos ellos números. También podemos decir que esta declaración crea una lista de longitud igual a cinco (ya que contiene cinco elementos).

Los elementos dentro de una lista pueden tener diferentes tipos . Algunos de ellos pueden ser enteros, otros son flotantes y otros pueden ser listas.

Python ha adoptado una convención que indica que los elementos de una lista están siempre numerados desde cero. Esto significa que el elemento almacenado al principio de la lista tendrá el número cero. Como hay cinco elementos en nuestra lista, al último de ellos se le asigna el número cuatro. No olvides esto.

Antes de continuar con nuestra discusión, debemos indicar lo siguiente: nuestra lista es una colección de elementos, pero cada elemento es un escalar.

La funcion len

La funcion len devuelve el numero de elementos almacenados actualmente

Recorrido de una Lista

Se puede hacer de varias maneras:

```
numeros = [10, 5, 7, 2, 1]
for x in numeros:
    print(x)
```

o tambien

```
numeros = [10, 5, 7, 2, 1]
for i in range(0,len(numeros)):
    print(numeros[i])
```

Imprimiran los elementos de la lista uno en cada linea

Operaciones con listas

Cambiar el valor del elemento de una lista

Sea la lista

numeros = [10, 5, 7, 2, 1]

Vamos a asignar un nuevo valor de 89 al primer elemento en la lista.

Lo hacemos de esta manera:

numeros[0]=89

La nueva lista sera: [89, 5, 7, 2, 1]

Copiar el valor del 5 elemento al segundo elemento

numeros[1]=numeros[4]

la nueva lista sera: [89, 1, 7, 2, 1]

El valor dentro de los corchetes que selecciona un elemento de la lista se llama un índice, mientras que la operación de seleccionar un elemento de la lista se conoce como indexación.

numeros = [10, 5, 7, 2, 1]

print("Contenido de la lista:", numeros)

.

Eliminando un elemento de la Lista

Cualquier elemento de la lista puede ser eliminado en cualquier momento, esto se hace con una instrucción llamada **del (eliminar).** Nota: es una instrucción, no una función.

SI tenemos la lista

lista = [20, 40, 90,5, 36, 28]

Si queremos eliminar el tercer elemento colocamos

del lista[2]

Ejecutaremos el siguiente programa:

```
lista = [20, 40, 90,5, 36, 28]
print('Lista inicial :',lista)

print('Longitud :',len(lista))
del lista[2]

print('Lista Final :',lista)
print('Longitud :',len(lista))
```

Los índices negativos son válidos

Un elemento con un índice igual a -1 es el último en la lista

Del mismo modo, el elemento con un índice igual a -2 es el anterior al último en la lista.

Ejercicio

Había una vez un sombrero. El sombrero no contenía conejo, sino una lista de cinco números: 1, 2, 3, 4 y 5.

Tu tarea es:

• Escribir una línea de código que solicite al usuario que reemplace el número central en la lista con un número entero ingresado por el usuario (paso 1).

• Escribir una línea de código que elimine el último elemento de la lista (paso 2).

• Escribir una línea de código que imprima la longitud de la lista existente (paso 3).

```
listaSombrero = [1, 2, 3, 4, 5]
# Esta es una lista existente de números ocultos en el sombrero.
print(listaSombrero)
indiceMedio = len(listaSombrero) // 2
listaSombrero[indiceMedio]=int(input('Ingrese valor : '))
print(listaSombrero)
del listaSombrero[-1]
print(len(listaSombrero))
print(listaSombrero)
```

Agregar un elemento a una Lista : Metodo append

Un nuevo elemento puede ser añadido al final de la lista existente:

lista.append(valor)

Dicha operación se realiza mediante un método llamado append(). Toma el valor de su argumento y lo coloca al final de la lista que posee el método

La longitud de la lista aumenta en uno.

Insertar un elemento a una lista: Metodo insert

El método insert() es un poco más inteligente: puede agregar un nuevo elemento en cualquier lugar de la lista, no solo al final.

lista.insert(ubicación,valor)

Toma dos argumentos:

- El primero muestra la ubicación requerida del elemento a insertar. Nota: todos los elementos existentes que ocupan ubicaciones a la derecha del nuevo elemento (incluido el

que está en la posición indicada) se desplazan a la derecha, para hacer espacio para el nuevo elemento.

- El segundo es el elemento a insertar.

Si tenemos el siguiente codigo:

```
valores=[2, 35, 78,45, 94, 13]
valores.append(10)
print(valores)
valores.insert(3,21)
print(valores)
```

El resultado sera:

```
[2, 35, 78, 45, 94, 13, 10]
[2, 35, 78, 21, 45, 94, 13, 10]
```

Lista Vacia

Podemos inicializar una lista de la siguiente manera:

```
lista =[]
```

Invertir una lista

También hay un método de lista llamado reverse(), que puedes usar para invertir la lista, por ejemplo:

```
lst = [5, 3, 1, 2, 4]
print(lst)
lst.reverse()
print (lst) # salida: [4, 2, 1, 3, 5]
```

Ordenar una Lista

Para ordenar una lista se utiliza el metodo sort

Ejemplo:

```
lista=[12, 5, 10, 4, 30]
print('Lista Inicial : ',lista)
lista.sort()
print('Lista ordenada: ',lista)
```

Asignacion de una Lista a otra

El nombre de una lista es el nombre de una ubicación de memoria donde se almacena la lista
Cuando asignas a una lista a otra. Copias la direccion de memoria de la lista.
Ejemplo:
lista1=[20, 5, 80, 42]
lista2=lista1
La asignacion lista2=lista1 copia el nombre de la lista1 a la lista2 no su contenido.
Ejemplo:

```
lista1=[20, 5, 80, 42]
lista2=lista1

print('Lista 1: ',lista1)
print('Lista 2: ',lista2)

lista2[0]=15

print('Lista 1: ',lista1)
print('Lista 2: ',lista2)
```

El resultado sera:

```
Lista 1:  [20, 5, 80, 42]
Lista 2:  [20, 5, 80, 42]
Lista 1:  [15, 5, 80, 42]
Lista 2:  [15, 5, 80, 42]
```

Lista1 y lista2 apuntan a la misma lista. Si se hace algun cambio en la lista, al imprimir las 2 listas, estas seran iguales.

Los operadores in y not

Python ofrece dos operadores muy poderosos, capaces de revisar la lista para verificar si un valor específico está almacenado dentro de la lista o no.
Estos operadores son:

elem **in** miLista
elem **not** in miLista

El primero de ellos (**in)** verifica si un elemento dado(su argumento izquierdo) está actualmente almacenado en algún lugar dentro de la lista(el argumento derecho) - el operador devuelve True en este caso.
El segundo (**not in**) comprueba si un elemento dado (su argumento izquierdo) está ausente en una lista - el operador devuelve True en este caso.

Rodajas

Una rodaja es un elemento de la sintaxis de Python que permite hacer una copia nueva de una lista, o partes de una lista.
En realidad, copia el contenido de la lista, no el nombre de la lista

La sintaxis es la siguiente:

miLista[inicio:fin]

Una rodaja de este tipo crea una nueva lista (de destino), tomando elementos de la lista de origen: los elementos de los índices desde el principio hasta el fin-1.

Ejemplo:

```
miLista = [10, 8, 6, 4, 2]
nuevaLista = miLista [1:3]
print(nuevaLista)
```

resultado:

[8, 6]

- Si utilizamos miLista[:] crea una nueva copia de todo el contenido de la lista.
  Ejemplo

```
lista1=[20, 5, 80, 42]

#Copia todo el contenido de la lista1 a la lista2
lista2=lista1[:]

print('Lista 1: ',lista1)
print('Lista 2: ',lista2)

lista2[0]=15

print('Lista 1: ',lista1)
print('Lista 2: ',lista2)
```

  Resultado:

  Lista 1: [20, 5, 80, 42]
  Lista 2: [20, 5, 80, 42]
  Lista 1: [20, 5, 80, 42]
  Lista 2: [15, 5, 80, 42]

  Al hacer la asignacion lista2=lista1[:] Copiamos el contenido de lista1 a lista2, eso quiere decir que lista1 y lista2 son dos posiciones de memorias diferentes. Si se hace algun cambio en alguna de ellas no afecta a la otra.

- Si omites inicio en tu rodaja, se supone que deseas obtener un segmento que comienza en el elemento con índice 0.
  En otras palabras, la rodaja sería de esta forma: miLista[:fin]
  Es un equivalente más compacto: miLista[0:fin]
  Ejemplo:

```
miLista = [10, 8, 6, 4, 2]
nuevaLista = miLista [:3]
print(nuevaLista)
```

  Resultado

  [10, 8, 6]

- Si omites el fin en tu rodaja, se supone que deseas que el segmento termine en el elemento con el índice len(miLista).
  En otras palabras, la rodaja sería de esta forma: miLista[inicio:]
  Es un equivalente más compacto: miLista[inicio:len(miLista)]
  Ejemplo:
  miLista = [10, 8, 6, 4, 2]
  nuevaLista = miLista[3:]
  print(nuevaLista)
  Resultado
  [4, 2]

Rodajas con Indices negativos

miLista = [10, 8, 6, 4, 2]
nuevaLista = miLista [1:-1]
print(nuevaLista)
El resultado del fragmento es: [8, 6, 4].

Si elinicio especifica un elemento que se encuentra más allá del descrito por fin (desde el punto de vista inicial de la lista), la rodaja estará vacía:

miLista = [10, 8, 6, 4, 2]
nuevaLista = miLista [-1:1]
print(nuevaLista)

La salida del fragmento es: [].

- La instrucción del descrita anteriormente puede eliminar más de un elemento de la lista a la vez, también puede eliminar rodajas:
  Ejemplo:
  miLista = [10, 8, 6, 4, 2]
  del miLista[1:3]
  print(miLista)
  Resultado
  [10, 4, 2].
- Se pueden eliminar todos los elementos a la vez
  Ejemplo:
  miLista = [10, 8, 6, 4, 2]
  del miLista[:]
  print(miLista)
  La lista se queda vacía y la salida es: [].

Ejercicios de Listas

1. Crear una lista que tenga los numeros desde el 1 al 10

```
lista=[]
for i in range(0,10):
   lista.append(i+1)
print('Lista obtenida: ',lista)
```

La salida sera:
Lista obtenida: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

2. Calcular la suma de los elementos de la lista : [9, 12, 10, 5, 30, 74]

```
numeros=[9, 12, 10, 5, 30, 74]
suma=0
for x in numeros:
   suma += x
print('La suma de los elementos de la lista es:',suma)
```

El resultado es:
La suma de los elementos de la lista es: 140

3. Ingresar n elementos reales a una lista. Imprimir los elementos de la lista uno por linea y luego calcular el promedio de ellos.

```
while True:
   n=int(input('Numero de elementos de una lista: '))
   if n>0:
      break
lista=[]
for i in range(0,n):
   x=float(input('Ingrese numero:'))
   lista.append(x)

print('Elementos de la lista')
for x in lista:
   print(x)

suma=0
for x in lista:
   suma += x
promedio=suma/len(lista)
print('El promedio es: ',promedio)
```

4. Ingresar n elementos en una lista y calcular el mayor de ellos

```
while True:
   n=int(input('Numero de elementos de una lista: '))
   if n>0:
      break
lista=[]
for i in range(0,n):
   x=float(input('Ingrese numero:'))
   lista.append(x)

print('Lista ingresada: ',lista)

mayor=lista[0]
for x in lista:
   if x>mayor:
      mayor=x

print('El mayor es : ',mayor)
```

5. Ingresar n elementos en una lista, luego ingresar un valor y luego indicar en que posicion se encuentra de la lista o indicar que no se encuentra.

```
while True:
   n=int(input('Numero de elementos de una lista: '))
   if n>0:
      break
lista=[]
for i in range(0,n):
   x=float(input('Ingrese numero:'))
   lista.append(x)

print('Lista ingresada: ',lista)

valor=float(input('Valor que se desea buscar: '))
encontrado = False

for i in range(len(lista)):
   encontrado = lista[i] == valor
   if encontrado:
      break

if encontrado:
   print("Elemento encontrado en el índice", i)
else:
   print("ausente")
```

6. Ingresar n elementos a una lista y luego ordenelo aplicando el metodo de burbuja

```
while True:
   n=int(input('Numero de elementos de una lista: '))
   if n>0:
      break
lista=[]
for i in range(0,n):
   x=int(input('Ingrese numero:'))
   lista.append(x)

print('Lista Ingresada : ',lista)

for i in range(0,n-1):
   j=len(lista)-1
   while j>i:
      if lista[j]<lista[j-1]:
         lista[j], lista[j-1] = lista[j-1],lista[j]
      j=j-1

print('Lista ordenada: ',lista)
```

7. Ingresar una lista de n numeros reales y al final obtener una lista sin datos repetidos. El objetivo es tener una lista en la que todos los números aparezcan no más de una vez.

```
 while True:
   n=int(input('Numero de elementos de una lista: '))
   if n>0:
      break
lista=[]
for i in range(0,n):
   x=int(input('Ingrese numero:'))
   lista.append(x)

print('Lista Ingresada : ',lista)
i=0
while i<len(lista):
    j=i+1
    while j<len(lista):
      if lista[i] == lista[j]:
         del lista[j]
         j=j-1
      j=j+1
    i=i+1
print("La lista solo con elementos únicos:")
print(lista)
```

Compresion de Listas

La comprensión de listas te permite crear nuevas listas a partir de las existentes de una manera concisa y elegante. La sintaxis de una lista de comprensión es la siguiente:

[expresión for elemento in lista if condicional]

El cual es un equivalente del siguiente código:

```
for elemento in lista:
    if condicional:
        expresión
```

Ejemplo:
crear una lista de cinco elementos con los primeros cinco números naturales elevados a la potencia de 3:

```
cubos = [num ** 3 for num in range (5)]
print(cubos)
```

salida: [0, 1, 8, 27, 64]

```
cuadrados = [x**2  for x in range (10)]
print(cuadrados)
probabilidades = [x for x in cuadrados if x % 2 != 0]
print(probabilidades)
```

salida: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
[1, 9, 25, 49, 81]

Ejercicios propuestos de Listas

1) Ingresar n números enteros en una lista y mostrar luego, primero la lista de los números pares que fueron ingresados y luego la lista de los números negativos.
2) Ingresar n enteros en una lista A y otros n enteros en una lista B y mostrar la lista de enteros del vector C. Donde cada C[i]=A[i]+B[i].
3) Calcule la media armónica de un conjunto de datos. La media armónica se define como: el Inverso del promedio de los inversos.
4) Sea una lista de n elementos reales. Mostrar la lista de números menores al promedio.
5) Ingresar N notas en una lista y determinar el porcentaje de aprobados y el porcentaje de desaprobados.
6) Ingresar n elementos en una lista y luego ingresar un elemento y reportar cuantas veces aparece ese elemento en la lista
7) Ingresar dos listas y reportar si son iguales.
8) Calcule la mediana de un conjunto de datos. La mediana de un vector ordenado es el elemento central si el número de términos es impar. Y la semisuma de los términos centrales si el número de términos es par.
9) Ingresar 2 listas de n y m elementos y calcular la unión, intersección y la diferencia del primero con el segundo.

Matrices en Python

Se puede usar listas anidadas en Python para crear matrices (es decir, listas bidimensionales). Por ejemplo:
Este codigo crea una matriz 3 filas y cada fila tiene los numeros del 1 al 8

```
datos =[[num for num in range (1,9)] for j in range(3)]
for fila in datos:
   print(fila)
```

El resultado es:

```
[1, 2, 3, 4, 5, 6, 7, 8]
[1, 2, 3, 4, 5, 6, 7, 8]
[1, 2, 3, 4, 5, 6, 7, 8]
```

Se puede utilizar la notacion datos[i][j] para acceder al elemento de fila i y columna j

Ejercicios de Matrices

1. Hacer un programa para ingresar f filas y c columnas a una matriz. Calcular el mayor , el menor y el promedio.

```
while True:
   f=int(input('Numero de filas:'))
   if f>0:
      break
while True:
   c=int(input('Numero de columnas:'))
   if c>0:
      break
matriz=[]
for i in range(f):
   fila=[]
   for j in range(c):
      print('matriz[',i,'][',j,']:',sep='',end='')
      valor=int(input())
      fila.append(valor)
   matriz.append(fila)

for fila in matriz:
   print(fila)

mayor=matriz[0][0]
for fila  in matriz:
   for x in fila:
      if x>mayor:
         mayor=x
print('El mayor es: ',mayor)

menor=matriz[0][0]
```

```
for fila in matriz:
   for x in fila:
      if x<menor:
         menor=x
print('El menor es: ',menor)

suma=0
for fila in matriz:
   for x in fila:
      suma+=x
promedio=suma/(f*c)
print('El promedio es: ',promedio)
```

2. Ingresar una matriz de f filas y c columnas y encontrar la suma de filas y la suma de columnas.

```
while True:
   f=int(input('Numero de filas:'))
   if f>0:
      break

while True:
   c=int(input('Numero de columnas:'))
   if c>0:
      break

matriz=[]
for i in range(f):
   fila=[]
   for j in range(c):
      print('matriz[',i,'][',j,']:',sep='',end='')
      valor=int(input())
      fila.append(valor)
   matriz.append(fila)

for fila in matriz:
   print(fila)

i=0
for fila  in matriz:
   sf=0
   for x in fila:
      sf+=x
   print('Suma de fila',i,'=',sf)
   i=i+1

for j in range(c):
```

```
sc=0
for i in range(f):
   sc+=matriz[i][j]
print('Suma de columna',j,'=',sc)
```

Ejercicios de Matrices

1. Ingresar una matriz de f filas y c columnas y calcular su matriz transpuesta.
2. Ingresar una matriz y reportar el mayor elemento de cada fila
3. Programa para ingresar una matriz de f filas y c columnas, luego Ingresar un número de columna y eliminarla de la matriz
4. Programa para ingresar una matriz de f filas y c columnas , luego ingresar el número de fila e insertar una fila en la matriz.
5. Ingresar una matriz de f filas y c columnas y luego intercambiar 2 columnas de la Matriz. El número de las columnas a intercambiar debe ingresarse.
6. Programa que ingresa el orden de una Matriz cuadrada y generarla y luego hacer lo siguiente:
   a) Calcula la suma de los elementos de la diagonal principal.
   b) Calcula el promedio de los elementos de la diagonal secundaria
7. Ingresar una matriz cuadrada y reportar si es simétrica. Recordar que una matriz es simétrica si se cumple la condición: a[i][j]=a[j][i]
8. Programa para ingresar una matriz de f filas y columnas y luego ordenar las columnas de la matriz.