

FUNDAMENTOS DE INFORMÁTICA

LISTAS

PROF. ING. VERÓNICA GALATI

TEMAS CLASE

- Listas:
 - Operaciones con listas:
 - Crear, Agregar, Modificar y Eliminar valores.
 - Consultar cantidad de elementos.
- Funciones y Listas

Listas

Estructura de datos que permite guardar más de un valor.

lista
Sub-índice

1	7	5	6	9
0	1	2	3	4

```
lista=[1,7,5,6,9]
```

Ejemplo 1: Crear una lista con 3 valores y mostrar la lista por pantalla

```
lista=[3,25,18]  
print(lista)
```

Ejemplo 2: Crear una lista con 3 valores,
Agregar un valor y mostrarlos por pantalla

```
lista=[3,25,18]  
print(lista)  
lista.append(20)  
print(lista)
```

Ejemplo 3: Crear una lista vacía, agregar dos valores del teclado y mostrar la lista.

```
lista=[]  
  
nro = int(input("Ingrese un numero"))  
lista.append(nro)  
nro = int(input("Ingrese otro numero"))  
lista.append(nro)  
  
print(lista)
```

Ejemplo 4: Crear una lista vacía, agregar N valores del teclado y mostrar la lista. Solicitar cuántos elementos se desea agregar por teclado

```
lista=[]

N=int(input("Ingrese cuántos elementos desea agregar a la lista:"))
while N<=0:
    N=int(input("Error. Cuántos elementos desea agregar a la lista:"))

for i in range(N):
    nro = int(input("Ingrese un numero"))
    lista.append(nro)

print(lista)
```

Ejemplo 5: Crear una lista con n números al azar entre 10 y 99, n se ingresa del teclado.

```
import random
n = int(input("ingrese la cantidad de numeros a crear"))
while n <=0:
    print("debe ser positivo")
    n = int(input("ingrese la cantidad de numeros a crear"))

lista=[]
i=0
while i < n:
    numero = random.randint(10,99)
    lista.append(numero)
    i = i + 1
print(lista)
```


Cantidad de elementos de una lista

numeros

4	7	9	2	1	8	6	0	5	3
0	1	2	3	4	5	6	7	8	9

```
len(numeros)
```



10

Mostrar un elemento de la lista

Referenciando con el subíndice a la posición

numeros

4	7	9	2	1	8	6	0	5	3
0	1	2	3	4	5	6	7	8	9

```
print(numeros[4])
```

1

Modificar elementos de una Lista

Referenciando con el subíndice a la posición y utilizando el operador de asignación

numeros

4	7	9	2	1	8	6	0	5	3
0	1	2	3	4	5	6	7	8	9

```
numeros[4] = 5
```

4	7	9	2	5	8	6	0	5	3
0	1	2	3	4	5	6	7	8	9

Eliminar el último elemento de la lista:

`miLista.pop()`

1	7	5	6	9
0	1	2	3	4

1	7	5	6
0	1	2	3

Eliminar un elemento que se encuentra ubicado en una posición determinada.

Ejemplo: eliminar el elemento que se encuentra en la posición 2

`miLista.pop(2)`

1	7	5	6	9
0	1	2	3	4

1	7	6	9
0	1	2	3

- ✓ Cambia la longitud de la lista, se elimina la última posición de la lista.
- ✓ Los elementos que se encuentran a la derecha del elemento eliminado, cambian de posición. Se produce un corrimiento.
- ✓ Para eliminar varios elementos, es recomendable recorrer del último al primer elemento.

Lista: secuencia de valores

Sub-índice: Se utiliza para acceder a una posición de la lista.

Si tenemos la lista: `miLista = [1, 7, 5, 6, 9]`

`miLista[0]` En la posición 0 se encuentra el valor 1

`miLista[1]` En la posición 1 se encuentra el valor 7

`miLista[2]` En la posición 2 se encuentra el valor 5

`miLista[3]` En la posición 3 se encuentra el valor 6

`miLista[4]` En la posición 4 se encuentra el valor 9

1	7	5	6	9
0	1	2	3	4

Lista: secuencia de valores

Sub-índice: Se utiliza para acceder a un elemento

```
lista=[1,17,5,26,9,35,19]
```

```
print(lista[2])
```



5

```
indice=4
```

```
print(lista[indice])
```



9

```
print(lista[indice+2])
```



19

Pueden usarse
constantes, variables
y expresiones
aritméticas

```
print(lista[indice+5])
```



```
Traceback (most recent call last):  
  File "D:\UADE\2019\2doCuatrimestre\Vi  
    print(lista[indice+5])  
IndexError: list index out of range
```


Operaciones con listas:

✓ Crear lista vacía:

```
lista=[]
```

✓ Crear lista con elementos:

```
lista=[26, 11]
```

✓ Agregar un elemento:

```
lista.append(nro)
```

✓ Cantidad de elementos:

```
len(numeros)
```

✓ Mostrar todos los elementos:

```
print(lista)
```

Operaciones con listas:

✓ Eliminar ultimo elemento:

```
lista.pop()
```

✓ Mostrar elemento de posición 2:

```
print(lista[2])
```

✓ Eliminar elemento de posición 2:

```
lista.pop(2)
```

✓ Modificar elemento de posición 2:

```
lista[2] = valor
```

CONSULTAS



Práctica!!!



Varios ejemplos de Funciones utilizando Listas

Ejemplo 1: Función Crear una lista con elementos ingresados desde teclado hasta ingresar -1

```
#Crear una lista hasta ingresar un -1|
def GenerarLista():
    lista=[]
    nro = int(input("Ingrese un numero, -1 para finalizar"))
    while nro != -1:
        lista.append(nro)
        nro = int(input("Ingrese un numero, -1 para finalizar"))

    return lista
```

LA FUNCIÓN RETORNA LA LISTA

Ejemplo 2: Crear una lista con N elementos al azar de dos dígitos. La función recibe cuántos elementos a crear

```
import random
```

```
#Crear una lista de n numeros al azar  
#de dos digitos
```

```
def GenerarListaAzar(n):
```

```
    lista=[]
```

```
    for cont in range(0, n):
```

```
        nro=random.randint(10,99)
```

```
        lista.append(nro)
```

```
    return lista
```

LA FUNCIÓN RETORNA LA LISTA

Ejemplo 3: Mostrar una lista por pantalla separando cada elemento con un espacio.

`end=" "` dentro del **print** indica que la siguiente impresión debe aparecer en el mismo renglón, separada por un espacio.

```
#Mostrar la lista, separando los elementos con un espacio
def MostrarLista(lista):
    i = 0
    while i < len(lista):
        print(lista[i], end=" ")
        i = i + 1
    print()
```

LA FUNCIÓN NO RETORNA VALOR O LISTA
(Retorna **NONE**)

Ejemplo 6: Función para retornar la suma de los elementos de la lista.

```
def SumarLista(lista):  
    suma = 0  
  
    for i in range(len(lista)):  
        suma = suma + lista[i]  
  
    return suma
```

LA FUNCIÓN RETORNA UN VALOR

Ejemplo 4: Modificar los elementos de una lista. Duplicar todos sus valores.

```
#Duplicar cada valor de la lista
def Duplicar(lista):
    i = 0
    while i < len(lista):
        lista[i] = lista[i] * 2
        i = i + 1
```

**SI MODIFICA LA LISTA, NO ES NECESARIO RETONAR
SIEMPRE SE MODIFICA LA LISTA ORIGINAL QUE SE RECIBE POR PARAMETRO**

Ejemplo 5: Eliminar un valor de la lista.

```
def EliminarValor(lista, valor):  
    i= len(lista) - 1  
    while i >= 0 :  
        if valor == lista[i]:  
            lista.pop(i)  
            i = i - 1
```

**SI MODIFICA LA LISTA, NO ES NECESARIO RETONAR
SIEMPRE SE MODIFICA LA LISTA ORIGINAL QUE SE RECIBE POR PARAMETRO**

Cómo ejecutamos las funciones



Si la función retorna la lista, al momento de llamar a la función se debe asignar a una variable

```
#Programa Principal  
N = int(input("ingrese la cantidad de elementos que desea crear"))  
miLista = generarListaAzar(N)
```

Si la función no retorna ningún valor, al momento de llamar a la función NO se debe asignar a una variable

```
#Programa Principal
N = int(input("ingrese la cantidad de elementos que desea crear"))

miLista = generarListaAzar(N)    #función que retorna una lista, la recibe la variable miLista

MostrarLista(miLista)           #función que no retorna valor.
```

Es importante prestar atención si la función No retorna la lista, No se debe asignar:

```
#Programa Principal  
N = int(input("ingrese la cantidad de elementos que desea crear"))
```

```
miLista = generarListaAzar(n)    #función que retorna una lista, la recibe la variable miLista
```

```
MostrarLista(miLista)            #funciones que no retorna valor.
```

```
Duplicar(miLista)
```

```
MostrarLista(miLista)
```


Si retorna un valor se debe asignar a una variable, utilizarla en un condicional o un print:

```
#Programa Principal
N = int(input("ingrese la cantidad de elementos que desea crear"))
miLista = generarListaAzar(N)
MostrarLista(miLista)
print("La suma de los valores de la Lista es", SumarLista(miLista))
```

```
suma = SumarLista(miLista)
print("La suma de los valores de la Lista es", suma)
```

```
print("La suma de los valores de la Lista es", SumarLista(miLista))
```

CONSULTAS



Búsqueda Secuencial

Consiste en ir recorriendo la lista elemento por elemento hasta encontrar el valor buscado o hasta llegar al final de la lista, lo que significa que el valor no se encuentra presente.

```
#Buscar un valor, retornar True o False
def BuscarValorSecuencial(lista, nro):
    i = 0
    while i < len(lista) and lista[i] != nro:
        i = i + 1

    if i == len(lista):
        encontrado = False
    else:
        encontrado = True
    return encontrado
```

```
#Buscar un valor, retornar True o False
def BuscarValorSecuencial(lista, nro):
    i = 0
    while i < len(lista) and lista[i] != nro:
        i = i + 1
    return i < len(lista)
```

#Programa Principal

```
N = int(input("ingrese la cantidad de elementos que desea crear"))
```

```
miLista = generarListaAzar(N) #función que retorna una lista, la recibe la variable miLista  
MostrarLista(miLista)        #función que no retorna valor.
```

```
valor = int(input("ingrese el valor a buscar:"))
```

```
if BuscarValorSecuencial(miLista, valor) == True: #función que no retorna True/False.  
    print("El valor se encuentra en la lista")  
else:  
    print("El valor No se encuentra en la lista")
```

CONSULTAS

