**Arrays/Arreglos en python**

Se ha discutido en clase lo que constituye un vector o arreglo de elementos. Tradicionalmente en otros lenguajes los arreglos eran estructuras fijas que se pre definían de un tamaño dado y eran luego inmodificables en el programa. En Python estas estructuras están implementadas sobre listas, que son estructuras dinámicas cuya cantidad de elementos se puede modificar en tiempo de ejecución por lo que algunas de las cosas que haremos con arreglos, no se pueden realizar en otros lenguajes.

En primera instancia trabajaremos con estas listas que están implementadas directamente en python. Es decir, que no se requiere ninguna biblioteca especial para utilizarlos (existen otros arreglos que son parte de la biblioteca numpy, que son mucho más versátiles). En Python, a diferencia de otros lenguajes, el arreglo tiene algunas facilidades para su uso. Por ejemplo para conocer la cantidad de elementos en un arreglo x, se puede utilizar la función len(x).

Para inicializarlo con una cantidad n de datos nulos

x=[None]\*n

o para declararlo vacío

x=[ ]

o para inicializarlo con una lista de números ya conocida, se utilizan los corchetes y los valores separados con coma como por ejemplo

x=[1,3,5,6]

Para agregar elementos al arreglo x

x.append(elemento)

Para acceder a un elemento determinado en un arreglo, se utilizan los corchetes: x[2] por ejemplo hace referencia al elemento en la tercera posición del arreglo (el primer elemento está en la posición 0).

De esta manera, aunque no para asignarles valores, se puede hacer referencia a intervalos de valores dentro del arreglo. Por ejemplo x[0:3] refiere a los primeros 4 valores, o x[2:] refiere a los elementos desde el 3ero en adelante.

Para modificar el valor en la posición i de un arreglo

x[i]=valor

1) Realizar un programa que solicite el ingreso de números enteros hasta que se ingrese el cero. El Programa debe imprimir luego todos los números ingresados.

2) Realizar un programa que tome un arreglo de números A = [1, 5, 8, 3, 30, 9, 13]

El programa debe imprimir la suma de todos los elementos y el promedio.

3) Con el anterior arreglo, realizar un programa que solicite el ingreso de un número e imprima el elemento en la posición solicitada. Si llega a ingresarse un número mayor a la cantidad de elementos, el programa debe imprimir todos los elementos y aclarar esa situación.

4) Realizar un programa que solicite el ingreso de números enteros hasta que se ingrese el cero (que no debe ubicarse en el vector). Luego debe imprimir los elementos desde el último hasta el primero en orden inverso de carga.

5)Dado el siguiente arreglo de números:

[1, 5, 8, 3, 30, 9, 13]

Imprimir en pantalla los números impares mayores a 3.

A = [1, 5, 8, 3, 30, 9, 13]

6) Realizar un programa que solicite el ingreso de números enteros hasta que se ingrese el cero. A continuación debe imprimir el máximo, el promedio y la suma de todos los números ingresados.

7) Realizar un programa que genere un vector tiempo con números del 0 al 3 con un paso de 0.1 segundos. A continuación se debe llenar un vector posiciones que calcule las posiciones correspondientes a un tiro vertical de 3 segundos de duración que comience con velocidad inicial 30 m/s, y una altura inicial de 45 m.

Si ambos vectores tienen el mismo tamaño agregando import matplotlib.pyplot as plt

al principio del código, se pueden graficar ambos vectores mediante el comando plt.plot(t,y)

8) Ordenar de la manera tradicional un arreglo de menor a mayor, implementando un burbujeo o bubble sort.

**Otros ejercicios**

9) ¿Qué hace/muestra por pantalla el siguiente código?

a=[]

b=int(input("ingresa numero"))

while b :

a.append(b)

b=int(input("ingresa numero"))

print(a)

10) ¿Qué muestra por pantalla el siguiente código?

a=[1,5,3,6,8,10]

for i in range(len(a)-2):

print(a[i])

11) ¿Qué muestra por pantalla el siguiente código?

a=[1,5,3,6,8,10]

for b in a:

if b%2!=0:

print(b)

12) Explicar qué muestra por pantalla el siguiente código y qué queda en el vector al final del programa. Si cambiáramos la instrucción del if por un b%2!=0 , ¿qué pasaría?

a=[1,50,13,6]

i=0

for b in a:

if b%2==0:

print(a[i])

a.append(b-1)