RESUMEN PEP 2 POR LUCHITO RIOS

ARCHIVOS

Modos:

r : Modo lectura, el archivo debe existir, abre archivos en formato texto plano (ej: txt, csv, …)

w: Modo escritura, si el archivo no existe lo crea y escribe el contenido que se desea añadir, si el archivo existe y posee contenido éste borra lo existente y escribe lo que se se desea añadir.

a: Modo añadir o agregar, si el archivo no existe lo crea y escribe el contenido que se desea añadir, si el archivo existe y posee contenido escribe el nuevo contenido después del contenido existente del archivo.

Tanto el modo "w" y "a" utilizan **write** no existe ~~append~~ en el modo "a".

Readline: lee una línea del archivo

Readlines(): Lee todas las líneas del archivo y guarda éstas en una lista (cada línea es un elemento de la lista).

LEER ARCHIVO

```
def LeerArchivo(nombreArchivo):
    archivo=open(nombreArchivo, "r")
    Lista = []
    for linea in archivo:
        Lista.append(linea.strip("\n"))
    archivo.close()
    return Lista
```

Se lee el archivo en modo lectura, cada línea se almacenará en una lista, entre cada línea en el archivo existe el símbolo implícito en el archivo salto de línea "\n", para eliminar este se debe usar strip("\n")

RECORDAR:

- Siempre cerrar el archivo
- Todo contenido leído desde un archivo se encuentra en String, si se desea utilizar algún valor numérico leído, pasar al tipo correspondiente (int, float, etc).
- Todo contenido a escribir al archivo se debe pasar a string (en el caso de los valores numéricos).

LEER ARCHIVO 2

Por ejemplo si cada línea del archivo sigue el formato: "nombre, edad, estadocivil"

```
def LeerArchivo(nombreArchivo):
    archivo=open(nombreArchivo,"r")
    L = []
    for lin in archivo:
       L.append(lin.strip().split(","))
    archivo.close()
    return L
```

La diferencia es que aquí se utiliza Split para pasar el cada línea a una lista y agregar esta lista como elemento de una lista doble, ejemplo del proceso:

"nombre, edad, estcivil" → ["nombre", "edad" , " estcivil"]

L.append(["nombre", "edad" , " estcivil"]) → [["nombre", "edad" , " estcivil"]]

ESCRIBIR ARCHIVO

```
def EscribirArchivo(ListaDoble):
  archivo=open("result.txt","w")
    for elemento in ListaDoble:
       archivo.write(ListaDoble[0]+
       "tiene "+str(ListaDoble[1])+"
       años \n")
    archivo.close()
    return True
```

Se abre un archivo en modo escritura, se está suponiendo que se escribirá cada fila (contiene nombre y edad) de una lista doble, al escribir una línea se agrega un salto de línea, se retorna True dado que la función retorna "algo" (no obligatorio).

NUMPY

4 MANERAS DE IMPORTAR NUMPY

import numpy: para utilizar una función se debe anteponer "numpy", ejemplo: numpy.array

import numpy as np: se abrevia numpy como np, para utilizar se debe anteponer np, ejemplo: np.array

from numpy import *: se importan todas las funciones de numpy

from numpy import array,arrange: solo se importan las funciones de array y arange

Realización de operaciones con numpy:

Lista = [1, 2, 3]

print Lista * 2 → [1,2,3,1,2,3]

print Lista / 2 → Error

Vector = array (Lista)

print Vector * 3 → [3, 6, 9]

print Vector / 2 → [0, 1, 1]

print Vector % 2 → [1, 0, 1]

matriz1 = [[1,2,3],[4,5,6],[7,8,9]]

matriz2 = [[6,7,8],[9,1,2],[3,4,5]]

vector1= array(matriz1)

vector2 = array(matriz2)

Ambos casos son válidos para multiplicar matrices (en este caso mismo resultado), solo que en el primer caso se debe importar la función "dot".

multiplicacion1 = numpy.dot(vector1,vector2)

multiplicacion2 = vector1.dot(vector2)

Recorrer matriz con la propiedad flat:

```
for elemento in matriz1.flat:
     print element
```

Se imprimirá cada uno de los elementos.

arange: es similar al range, solo que con el arange se crean vectores, y además es posible utilizar valores flotantes, ejemplo:

numpy.arange(1,10) → se crea un vector que va desde 1 a 9

RESUMEN PEP 2 POR LUCHITO RIOS

[1 2 3 4 5 6 7 8 9]
arange(1, 10.1, 0.1) → se creará un vector partiendo desde 1 hasta 10 (10.1 – 0.1) con incremento en 0.1
[1 1.1 1.2 1.3 ….. …….10]

zeros: crea una lista o matriz de ceros, ejemplo numpy.zeros(2) → [0. , 0.] (que haya un punto después de un cero que es tipo float)
numpy.zeros((2,2)) → [[0. 0.] [0. 0.]]

ones: crea una lista de unos, misma utilización que zeros.

linspace, genera un rango especifico de números, ejemplo:

numpy.linspace(0,100,10) →[0., 11.11, 22.22, 33.33, 44.44, 55.55, 66.66, 77.77, 88.88, 100.]
linspace(inicio, final, cantidad de elementos a generar)

random: este módulo permite generar vectores de números aleatorios

La función randint genera un vector de tamaño n (tercer parámetro) que toma números enteros del número indicado como primer parámetro hasta al número indicado en el 2do parámetro -1

Por ejemplo:
numpy.random.randint(1,10,5)
se genera un vectro de tamaño 5 con números aleatorios los cuales van del 1 al 9

eye: genera una matriz identidad de nxn de acuerdo al valor ingresado como parámetro

Ejemplo:
numpy.eye(3)
[[1 0 0]
[0 1 0]
[0 0 1]]

reshape: dar una nueva "forma" a un vectorEjemplo:
En el siguiente ejemplo se da una nueva forma al vector de elementos que van del 1 al 6, ahora los lleva a un vector bidimensional de tamaño 2x3

vector = array([1,2,3,4,5,6])
nuevoVector = vector.reshape([2,3])
[[1 2 3]
[4 5 6]]

linalg: el módulo linalg de numpy se usa en sistemas de ecuaciones lineales

Ax = B
x = inv(A)*B inv = inversa
en donde el determinante de A tiene que ser distinto de 0, si es igual a 0 no hay solución

por ejemplo si tengo el siguiente sistema
0x1 + 1x2 = 0
2x1 + 3x2 = 1

los valores
a = numpy.arange(0,4).reshape([2,2])
[[0 1]
[2 3]]
b = numpy.arange(0,2).reshape([2,1])
[[0]
[1]]

con la función det obtengo la determinante
print numpy.linalg.det(a)
-2.0
con la función inv obtengo la inversa
print numpy.linalg.inv(a)
[[-1.5 0.5]
[1. 0.]]

realizo la operación para obtener la solución
print numpy.linalg.inv(a).dot(b)
[[0.5]
[0.]]

podemos usar la función solve para obtener directamente la solución sin haber realizado el cálculo anterior

print numpy.linalg.solve(a,b)
[[0.5]
[0.]]

veamos el siguiente sistema
1x1 + 2x2 = 4
2x1 + 4x2 = 7
A = numpy.array([[1,2],[2,4]])
B = numpy.array([[4],[7]])

print numpy.linalg.det(A)

al imprimir la solución arroja error, dado que el determinante es 0, es decir, no tiene solución

print numpy.linalg.solve(A,B)

MATPLOTLIB

Importar el módulo pyplot de matplotlib, se redefine como plotter

import matplotlib.pyplot as plotter

Se crea un vector que va desde -10 a 10, se tiene un dominio más amplio, por lo tanto, la línea del gráfico tendrá un aspecto más "curvilíneo"

vector =arange(-10,10.1,0.1)
resultadoFuncion=vector **2

Plot: para dibujar el gráfico, si se tiene un parámetro se toma como par ordenado x partiendo desde 0
plotter.plot(resultadoFuncion)

Dos parámetros vector x e y, ambos deben ser del mismo tamaño

plotter.plot(vector,resultadoFuncion)

RESUMEN PEP 2 POR LUCHITO RIOS

Mostrar el gráfico en pantalla
plotter.show()

Configuraciones, se deben anteponer antes del "plotter.show()"

title(string): Para indicar el título de lo que estamos graficando
xlabel(string): Para indicar un rótulo en el eje x
ylabel(string):Para indicar un rótulo en el eje y
.setp(): para darle formato a las líneas del gráfico, para ello se utilizan pares propiedad, valor, que significan:

- Marker: representa el marcador con el que se presentarán los puntos
- Linestyle: representa el estilo de la línea
- Color: representa el color de la línea y sus puntos
- Linewidth: representa el ancho de la línea

marker:

"."	point
","	pixel
"o"	circle
"v"	triangle_down
"^"	triangle_up
"<"	triangle_left
">"	triangle_right
"1"	tri_down
"2"	tri_up
"3"	tri_left
"4"	tri_right
"8"	octagon
"s"	square
"p"	pentagon
"P"	plus (filled)
"*"	star
"h"	hexagon1
"H"	hexagon2
"+"	plus
"x"	x
"X"	x (filled)
"D"	diamond
"d"	thin_diamond
"|"	vline
"_"	hline

linestyle: ['-' | '--' | '-.' | ':' | 'steps' | 'None']
color:
b: blue
g: green
r: red
c: cyan
m: magenta
y: yellow
k: black
w: white

Para utilizar setp, se debe guardar el gráfico al dibujarlo, no es necesario usarlo con todas sus propiedades.
.
Ejemplos de uso:

```
plotter.title("titulo")
plotter.xlabel("x")
plotter.ylabel("y")
grafico = plotter.plot(vector, resultadoFuncion)
plotter.setp(grafico,'marker','*','linestyle','none',
'color','g', 'linewidth',3.0)
plotter.setp(grafico,'color','r')
```

label: leyenda del gráfico
plot(x,y, label = "leyenda gráfico")

para mostrar las leyendas se usa legend()

plt.legend()

Ejemplo:

```
import matplotlib.pyplot as plt

a =plt.plot(range(0,10),label="leyenda 1")
b=plt.plot(range(-10,1,1),range(10,-1,-1),
label="leyenda 2")
plt.setp(a,"color","b")
plt.setp(b,"color","g")
plt.title("titulo del grafico")
plt.xlabel("nombre del eje x")
plt.ylabel("nombre del eje y")
plt.legend()
plt.show()
```