FUNDAMENTOS DE COMPUTACIÓN
GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES
Universidad de Cantabria

Práctica 10 (laboratorio)
5 de diciembre del 2018

Ficheros en PYTHON

- Todos estamos familiarizados con el sistema de archivos que los ordenadores utilizan para almacenar información. Es útil conocer cómo podemos acceder desde un programa a los datos que contiene un archivo para manejarlos de manera automatizada.
- También es conveniente almacenar en ficheros los datos que devuelve un algoritmo, para poder utilizarlos con posterioridad, publicarlos o compartirlos con otras aplicaciones.

Analizaremos algunas operaciones básicas de edición de archivos de texto, tratándolos como objetos dentro de un programa Python. Es importante saber (o exigir) el formato que presenta el fichero de entrada.
Cuanto menos rígida sea esta especificación, más complicado será el trabajo del programador.

1. Supongamos que disponemos del fichero `gastos.txt`. Veamos, con el siguiente fragmento de código, como acceder a su contenido:

```
1 fichero=open("gastos.txt","r")
2 total=0
3 for linea in fichero:
4   num=int(linea)
5   total+=num
6 fichero.close()
7 print ("TOTAL",total)
```

gastos.txt
```
1500
350
2600
```

Destaquemos tres instrucciones del programa anterior:

1. `fichero=open("gastos.txt","r")` Con esta instruccion *preparamos* el archivo para ser leído (parámetro `r`, read) y lo asociamos a la variable fichero.
2. `fichero.close()`. Es importante *cerrar* los archivos que ya no vamos a utilizar.
3. `for linea in fichero:`. Un fichero de lectura puede interpretarse como una lista formada por cadenas de texto (sus líneas).

El método `readlines()` retorna una lista con cada línea del archivo de texto.

```
>>> fichero=open("gastos.txt",r")
>>> fichero.readlines()
['1500\n', '350\n', '2600']
```

- Existen dos modos adicionales a `r` (lectura) para abrir un archivo: `w` (escritura) y `a` para añadir datos al final de un archivo. Con el modo `w` se borran los datos que el archivo contuviera anteriormente.

```
1 print("Vayan escribiendo sus nombres.")
2 print('Escriban "FIN" cuando no quede nadie más.')
3 cadena="PERSONAL\n"
4 fichero=open("lista.txt","w")
5 while cadena!="FIN":
6   fichero.write(cadena+"\n")
7   cadena=input()
8 fichero.close()
```

```
>>> ====RESTART ===
>>>
Vayan escribiendo sus nombres.
Escriban "FIN" cuando no quede nadie más.
Claude E. Shannon
George Boole
Bertrand Russell
FIN
```

Se ha generado el archivo `lista.txt`:

lista.txt
```
PERSONAL

Claude E. Shannon
George Boole
Bertrand Russell
```

2. El *sistema de D'Hondt* gobierna la asignación de escaños parlamentarios en muchos procesos electorales. En esta hoja construiremos un programa para automatizarlo, calculando, a partir de un archivo que contenga la distribución del voto entre las distintas candidaturas, otro donde aparezcan los representantes obtenidos por cada una de ellas.

- Leyendo las líneas de un fichero.

```
1 archivo=open('votos.txt','r')
2 for linea in archivo:
3     print(linea)
4 archivo.close()
```

votos.txt
```
P.A. 314

P.B. 1678

P.C. 501

P.D. 10
```

- Para leer y escribir en un archivo con caracteres fuera del rango ASCII, necesitamos cargar el modulo `codecs`. E indicar la codificación del archivo con un tercer argumento.

```
1 import codecs
2 archivo1=codecs.open('circunscripción5.txt','r','utf8')
3 archivo2=codecs.open('c_circunscripción5.txt','w','utf8')
4 for linea in archivo1:
5     archivo2.write(linea)
6 archivo1.close()
7 archivo2.close()
```

circunscripción5.txt
```
Escaños_circunscripción5: 3
P.A. 314
P.B. 1678
P.C. 501
P.D. 10
```

También Python abre y manipula otros tipo de archivos `.pdf`, `.doc` .etc. Pero, como decíamos el trabajo del programador es mayor. Por ejemplo, para abrirlo en binario se cambia el modo `'r'` por `'rb'`. Para simplificar escritura, en esta hoja manipularemos archivos de texto plano en el rango ASCII.

- No nos sirve de mucho presentar por la pantalla los contenidos de un fichero de texto, lo que conseguimos, por otra parte, con el comando `cat`. Vamos a construir una lista de *tuplas*: cada elemento de la lista será un par `(nombre del partido,número de votos)`.

```
archivo=open('votos.txt','r')
lista=[]
for linea in archivo:
    par=linea.split()
    lista.append((par[0],int(par[1])))
archivo.close()
print (lista)
```

Observa que, para que este fragmento de código funcione como esperamos, es preciso que cada línea del archivo de entrada se atenga al formato `TEXTO NÚMERO`. Obtendremos un error si, por ejemplo, el nombre de un partido consta de varias palabras.

- En las elecciones al Congreso de los Diputados, un primer requisito para obtener representación por una circunscripción es recibir al menos el 3 % de los votos. Contémoslos (utilizando una variable llamada `suma`, por ejemplo) y quedémonos después con las candidaturas que cumplan esta condición:

```
umbral=suma*.03
pasan=[]
for x in lista:
    if x[1]>=umbral:
        pasan.append(x)
print(pasan)
```

- El sistema de D'Hondt asigna sucesivamente los escaños que hay que repartir. El primer escaño le corresponde a la lista que tenga más votos. Amplía el programa para que localice la candidatura más votada.

```
i=ganador(pasan)
print('El primer escaño va para',pasan[i][0]+'.')
```

- Cada uno de los escaños restantes, hasta completar los que hay para repartir, también se otorga a la candidatura que tenga un número asociado mayor. Pero este número no se mantiene constante: no es siempre el número de votos (si así fuese, todos los escaños serían para la lista más votada). El número que se asocia a cada candidatura es el cociente entre los votos que ha recibido y el número de escaños que ya se le han asignado más uno.
  Por ejemplo, para asignar el segundo escaño, se ordenan las candidaturas según su número de votos, salvo la que tiene más (y que ya ha recibido un escaño), cuyo número de votos se divide por dos. Para asignar el tercero, habrá que dividir por tres los votos de la candidatura ganadora, si le han correspondido los dos escaños anteriores; o por dos los de las dos primeras, si son dos los partidos que ya tienen representante.

```
print ('Escaños totales: ')
total=int(input())

cuentas=[]
asig=[]
for x in pasan:
    cuentas.append(x[1])
    asig.append(0)

#  Hemos guardado en la lista «cuentas» una copia de los votos de cada
# candidatura, para ir haciendo aquí las divisiones oportunas.
#  En la lista «asig» iremos anotando los escaños que se asignen.

for i in range(1,total+1):
    j=maxi(cuentas)
    asig[j]+=1
    cuentas[j]=pasan[j][1]/(asig[j]+1)
print(asig)
```

  Este programa pregunta directamente al usuario cuántos parlamentarios hay en juego, pero sería más conveniente que esta información estuviera ya contenida en el fichero de entrada. Modifica el programa según esta especificación.

- Una vez calculados los resultados, los colocamos en un fichero de texto.

```
archivo=open('resultados.txt','w')
for i in range(len(pasan)):
    if asig[i]>0:
        archivo.write('{0:10s} {1:1d}\n'.format(pasan[i][0]+':',asig[i]))
archivo.close()
```

- En vez de ejecutar el programa cada vez que queramos calcular los resultados de una circunscripción, es interesante, por ejemplo, que el programa lea varios archivos y haga las cuentas para cada uno. Si prevemos tener en un directorio las votaciones de varias provincias, en archivos que comienzan por `c_`, podemos hacer lo siguiente:

```
from os import getcwd,listdir
ruta=getcwd()
archivos=listdir(ruta)
datos=[]
for x in archivos:
    if x[:2]=='c_':
        datos.append(x)
for provincia in datos:
    archivo=open(provincia,'r')
    (...)
    n_salida='res'+provincia[1:]
    salida=open(n_salida,'w')
    (...)
```

- ¿Qué críticas podemos hacerle a nuestro programa? Supón que el escrutinio ha resultado en 1 000 votos para cada una de las cuatro candidaturas que se presentan y hay cinco escaños a repartir...

EJERCICIOS

1. Diseña un programa que dado un fichero de texto *fuente.txt*, genere otro fichero denominado *parte.txt* conteniendo solamente las lineas impares del fichero *fuente.txt*.

2. Diseña un programa que dado un fichero de texto *distancias.txt* conteniendo en cada línea, los kilómetros recorridos por un taxista, genere otro fichero *media.txt* con la media aritmética:

distancias.txt
```
50.2
99.8
30
```

media.txt
```
50
99.8
30
Media diaria: 60. Kilometros
```

3. En un archivo de datos se tiene la siguiente información sobre calificaciones de alumnos:

DatosCalificaciones.txt
```
NOMBRE Y APELLIDOS    CONTINUA  EXAMEN FINAL
George Boole,             5.0,      9.1
John von Neumann,         8.6,      7,8
Bertrand  Russell,        9.0,      8.2
Leonardo Torres,          8.2,      7,1
Alan Turing,              7.3,      8.6
```

La calificación de la asignatura es una media ponderada: 30 % de la CONTINUA y el 70 % del EXAMEN FINAL. Por ejemplo, la calificación de George Boole es:

$$5{,}0 \times 0{,}3 + 9{,}1 \times 0{,}7 = 7{,}8699999999$$

Redondeando a una cifra decimal 7,9. Para redondear podemos utilizar la función `round`(7.8699999999,1)=7.9

Construir un programa en PYTHON que leyendo el archivo `DatosCalificaciones.txt` cree otro archivo denominado `CalificacionFinal.txt`:

CalificacionFinal.txt
```
NOMBRE Y APELLIDOS    CONTINUA   EXAMEN FINAL  CALIFICACION FINAL
George Boole,             5.0,      9.1,          7.9
John von Neumann,         8.6,      7.8,          8.0
Bertrand  Russell,        9.0,      8.2,          8.4
Leonardo Torres,          8.2,      7.1,          7.4
Alan Turing,              7.3,      8.6,          8.2
MEDIA ARITMETICA,         7.6,      8.2,          8.0
```

donde la última "columna" contiene la calificación de la asignatura y, la última "fila" contiene la media aritmética de la cada "columna", por ejemplo,

$$7{,}62 = (5{,}0 + 8{,}6 + 9{,}0 + 8{,}2 + 7{,}3)/5, \qquad \texttt{round(7.62,1)=7.6}$$

4. Una sala de cine que resiste abierta al público cuenta con ciento sesenta butacas, repartidas en diez filas de dieciséis butacas cada una. Hagamos un programa para el despacho de localidades.

1. Cuando acudan a comprar entradas, los clientes verán en un monitor un mapa de la sala para que puedan elegir los asientos de su agrado. Comencemos por escribir una función que dibuje ese mapa:

```
>>> dibuja()
          ----------------

- - - - - - - -     - - - - - - - -
- - - - - - - -     - - - - - - - -
- - - - - - - -     - - - - - - - -
- - - - - - - -     - - - - - - - -
- - - - - - - -     - - - - - - - -
- - - - - - - -     - - - - - - - -
- - - - - - - -     - - - - - - - -
- - - - - - - -     - - - - - - - -
- - - - - - - -     - - - - - - - -
- - - - - - - -     - - - - - - - -
```

2. Según vayamos vendiendo billetes, registraremos en una lista los asientos libres. El programa podría comenzar con la instrucción `asientos=[True]*160`, donde cada elemento de la lista indica si la butaca correspodiente está libre.

   Cuando queramos comprobar el estado de un asiento, tendremos que acceder al elemento de una posición determinada (un número entre el 0 y el 159) de la lista. Sin embargo, en el programa también necesitaremos localizar los puestos mediante dos coordenadas: el número de fila y el de butaca (dentro de cada fila). Escribe un par de funciones para pasar de una representación de los asientos a la otra, suponiendo que las filas se numeran del 0 al 9 y, dentro de cada fila, las butacas van consecutivamente desde la 0 hasta la 15.

```
>>> par(15)
(0, 15)
>>> par(16)
(1, 0)
>>> par(159)
(9, 15)
```

```
>>> absoluta(1,0)
16
>>> absoluta(3,10)
58
```

3. Con ayuda de las funciones recién definidas (en realidad, basta `absoluta`), modifica la función `dibuja` para que represente las localidades vendidas con otro símbolo.
4. Escribe un programa que dibuje el mapa de la sala y dé opción al usuario de vender una localidad o anular una venta. Después, se volverá a dibujar el mapa.
5. Escribe una función similar a `dibuja` que guarde en un fichero de texto el esquema con la ocupación del cine.
6. Los ficheros que generamos con esta función son útiles para «consumo humano». Si lo que queremos es aprovechar la información para reutilizarla en otro programa, puede ser más conveniente (no por espacio, pero sí para la lectura posterior) guardar directamente la representación de la variable `asientos`:

```
salida=open('guarda.txt','w')
salida.write(repr(asientos))
salida.close()
entrada=open('guarda.txt')
linea=entrada.readline()
entrada.close()
asientos=eval(linea)
```

7. Amplía el programa para que gestione varias proyecciones (por ejemplo, todos los pases de una película durante una semana), guardando la información de cada una en ficheros de texto distintos, y cambiando de una a otra según se precise.