Clase 18: lectura/escritura de Archivos y Diccionarios en Python

Lectura/Escritura de Archivos

	Ram	Disco
Tipo de Memoria	volátil	permanente
velocidad	rápida	lenta
tiempo de acceso	constante	variable
costo	caro	barato
capacidad	limitada	mayor

Problema

Leer líneas del usuario terminando con la palabra "fin" y grabarlas en un archivo en **disco** de nombre `"lineas.txt"`.

```
#grabar lineas en archivo

#abrir (preparar) archivo para escribir(grabar)
A=open("lineas.txt","w")

#leer lineas hasta “fin”
while True:
    linea=input("palabra(o fin)? ")
    if linea=="fin": break

    #grabar linea en archivo
    A.write(linea+"\n")

#cerrar archivo
A.close()
```

```
palabra(o fin)? hola
palabra(o fin)? esto
palabra(o fin)? es
palabra(o fin)? un
palabra(o fin)? test
palabra(o fin)? fin
```

```
A=open("lineas.txt","w")
```

- `A`: variable (objeto) en memoria que representa al archivo (y contiene sus principales características: ubicación, tamaño, cursor, etc)
- `"lineas.txt"`: nombre externo del archivo (en el disco).
- El sufijo `.txt` se usa para archivos que contienen texto (caracteres).
- Función open "abre" (prepara, inicializa) archivo para escritura (`"w"`)
- si archivo no existe, se crea, y si ya existe, se regraba
- ubica el cursor del archivo al comienzo del espacio asignado (para grabación de información)

```
A.write(linea+"\n")
```

escribe (graba) una línea en el archivo

- graba caracteres del string linea
- graba una marca de fin de línea (carácter especial newline o `\n`)
- ubica cursor después de newline

Ejemplo:

```
A.write("hola\n")
```

escribe en el archivo en disco:

```
hola\n
      ^ (cursor de archivo)
```

```
A.close()
```

- "cierra" el archivo
- graba marca de fin de archivo (eof)
- libera recursos (memoria) asociada al objeto `A`
- archivo no se puede volver usar, salvo después de otro open

Ejemplo:

```
A.write("chao\n")
A.close();
```

```
hola\nchao\n
```

: marca de fin de archivo

```
A=open("lineas.txt","r")
```

- `A`: objeto que representa al archivo
- `"abre"` (prepara, inicializa) archivo para lectura (`"r"`)
- si archivo no existe, aborta
- si existe, ubica el cursor al comienzo del archivo
- si el 2° argumento es `"r"` se puede omitir

```
A.close()
```

- "cierra" el archivo
- si se omite, se cierra al terminar programa

```
for linea in A
```

- lee todas las líneas del archivo
- en cada iteración, se lee una línea del archivo
- y se asigna a la variable (incluyendo newline)

```
A=open("lineas.txt","r")
for linea in A:
    print (linea)
A.close()
```

```
hola

esto

es

un

test
```

Nota: cómo borro el `'\n'` del final del string? (hint `strip`)

```
A=open("lineas.txt","r")
for linea in A:
    print (linea.strip())
A.close()
```

```
hola
esto
es
un
test
```

Problema

Copiar un archivo en otro, obteniendo del usuario los nombres de los archivos

Ejemplo:

```
input? lineas.txt
output? lineas1.txt
```

```
#abrir archivos de entrada y salida
arch1= input("input? ")
arch2= input("output? ")

A=open(arch1,"r")
B=open(arch2,"w")

#copiar todas las líneas del archivo
for linea in A:
    B.write(linea)

#cerrar archivos
A.close()
B.close()
```

```
input? lineas.txt
output? lineas1.txt
```

Problema

mostrar las líneas de un archivo que contengan un string

Ejemplo de uso:

```
String a buscar? Jose
Archivo? nombres.txt
```

```
Jose Manuel Perez
Maria Jose Amigo
Josefina Smith
```

```
#mostrar lineas que contienen un string determinado

#obtener strings y abrir archivo
s=input("string a buscar? ")
A=open(input("archivo? "),"r")
#leer todas las líneas del archivo
for linea in A:
    #mostrar lineas con string
    if s.lower() in linea.lower(): #linea.find(s)>=0
        print (linea[0:len(linea)-1])
#cerrar archivo
A.close()
```

```
string a buscar? Jose
archivo? nombres.txt
Jose Manuel Perez
Maria Jose Amigo
Josefina Smith
```

Problema 1 (trabajo individual)

Contar las líneas y los caracteres de un archivo

```
Nombre de archivo? nombres.txt
Num. lineas: 7
Num. caracteres: 105
```

```
#obtener strings y abrir archivo
A=open(input("archivo? "),"r")
#leer todas las líneas del archivo
contador_lineas=0
contador_caracteres=0
for linea in A:
    contador_lineas+=1
    contador_caracteres+=len(linea)
```

```
print ("Num. lineas:", contador_lineas)
print ("Num. caracteres:", contador_caracteres)
#cerrar archivo
A.close()
```

```
archivo? nombres.txt
Num. lineas: 7
Num. caracteres: 105
```

Problema 2 (trabajo individual)

Leer un archivo y guardar sus lineas ordenadas en otro archivo

```
Nombre de archivo?...
Nombre archivo de salida?...
```

```
#obtener strings y abrir archivo
A=open(input("Nombre archivo? "),"r")
#leer todas las líneas del archivo
lineas = []
for linea in A:
    #leer y guardar en un arreglo
    lineas.append(linea)
#cerrar archivo
A.close()
lineas.sort()
A=open(input("Nombre archivo de salida? "),"w")
for linea in lineas:
    A.write(linea)

A.close()
```

```
Nombre archivo? nombres.txt
Nombre archivo de salida? nombres_ordenado.txt
```

Diccionarios Python

Los diccionarios son una estructura indexada mutable, al igual que los arreglos. Sin embargo no están indexados por números, sino que por "llaves" (strings)

```
diccionario = {} # diccionario vacio
```

```
diccionario
```

```
{}
```

```
diccionario = dict() # otro diccionario vacio
diccionario
```

```
{}
```

```
D={"c":1,"a":2,"d":4} #secuencia de items llave:valor
```

```
D # muestra en orden arbitrario
```

```
{'a': 2, 'c': 1, 'd': 4}
```

```
'c' in D    #pertenencia
```

```
True
```

```
D['d']      #buscar
```

```
4
```

```
D['a']=2    #agregar
D
```

```
{'a': 2, 'b': 3, 'c': 1, 'd': 4}
```

```
del D['c']  #borrar
D
```

```
{'a': 2, 'b': 3, 'd': 4}
```

```
D['a']=5    #cambiar
D
```

```
{'a': 5, 'b': 3, 'd': 4}
```

```
for llave in D:
    print (D[llave]) #imprime todos los valores guardados en D
```

```
5
4
3
```

Agenda con diccionario de Python

```
agenda={"Andres":22234646,"Maria":98723456,"David":43335555}
```

```
#agenda: dict (nombre:fono)
agenda={"Andres":22234646,"Maria":98723456,"David":43335555}

#buscar: str dict -> int (o None si no esta)
#buscar nombre en agenda y devolver fono
#ej: buscar("c",agenda)->1
def buscar(nombre,agenda):
    assert type(agenda)==dict and type(nombre)==str
    if nombre in agenda:
        return agenda[nombre]
    else:
        return None
assert buscar("Andres",agenda)==22234646
```