Manejo de archivos

Python incorpora un tipo especial de dato, denominado file, este tipo de dato se comunica con el sistema operativo para poder almacenar/recuperar datos desde el almacenamiento secundario. El manejo de archivos involucra algunas funciones que ya conocemos y otras que resultan nuevas.

Funciones para manejo de archivos:

Abrir un archivo: Se requiere para poder acceder al contenido del archivo (escribir datos al archivo o leer datos desde el archivo). En Python se debe utilizar el comando open, cuya forma de uso es la siguiente: *open(filename, filemode) en donde:*

filename, indica el nombre del archivo a abrir y filemode el modo (para que) en que se abrirá el archivo. Los valores permitidos para el filemode son:

r - Read (Valor por defecto), abre el archivo para leer los datos que contiene. Causa un error si el archivo no existe.

a - Append – Abre el archivo para escribir (siempre se añade al final del archivo) y si el archivo no existe lo crea.

w - Write – Crea un archivo para escribir sobre él, si existe lo sobrescribe.

x - Create – Crea un archivo, si el archivo existe genera un error

Adicionalmente se puede especificar si el archivo se tratará como de texto (t) o binario(b)

Ej01: Abrir un archivo para lectura, si el archivo no existe produce un fallo

```
try:
    f = open('datos.txt')
    f.close()
except:
    pass
```

Ej02: Abrir un archivo para escritura, si el archivo no existe lo crea y si existe, todo el contenido se pierde

```
try:
    f = open('datos.txt','w')
    f.close()
except:
    pass
```

Ej03: Abrir un archivo para escritura, si el archivo no existe lo crea, si existe se mantiene el contenido

```
try:
    f = open('datos.txt','a')
    f.close()
except:
    pass
```

Ej04: Crear un archivo vacío

```
try:
    f = open('datos.txt','x')
    f.close()
except:
    pass
```

Leer la información contenida en un archivo:

Para poder leer la información almacenada en un archivo, Python nos entrega las siguientes formas **(el archivo debe abrirse en alguno de los modos que permite lectura: r, r+, w+, a+)**

1) El método **read**: Este método se encarga de leer la totalidad del archivo, aun cuando tiene un argumento que se puede utilizar para especificar la cantidad de caracteres a leer del archivo.

Ej05: Leer el contenido de un archivo por completo

```
try:
    f = open('datos.txt','r')
    data = f.read()
    f.close()
except:
    pass
```

Ej06: Leer 200 bytes del archivo

```
try:
    f = open('datos.txt','r')
    data = f.read(200)
    f.close()
except:
    pass
```

2) El método **readline**: Este método lee el archivo una línea a la vez.

Ej07: Leer dos líneas del archivo

```
try:
    f = open('datos.txt','r')
    data1 = f.readline()
    data2 = f.readline()
    f.close()
except:
    pass
```

Ej08: Leer 50 bytes de la primera línea del archivo

```
try:
    f = open('datos.txt','r')
    data = f.readline(50)
    f.close()
except:
    pass
```

3) El método **readlines**, lee todo el archivo y lo carga en una lista donde cada elemento de la lista contiene una línea del archivo.

Ej09: Leer todo el archivo

```
try:
    f = open('datos.txt','r')
    data = f.readlines()
    f.close()
except:
    pass
```

Ej10: Leer líneas del archivo hasta que se tomen 200 bytes

```
try:
    f = open('datos.txt','r')
    data = f.readlines()
    f.close()
except:
    pass
```

4) En Python el tipo de dato file (archivo) es un iterable, por lo que se puede recorrer utilizando el for.

*Se recomienda utilizar el manejador de contextos **with,** dado que garantiza el cierre automático del archivo (al finalizar el bloque) aun cuando se produzca una excepción en el código. Por tanto a partir de éste momento lo utilizaremos.*

Ej11: Iterar sobre el archivo (toma una línea a la vez hasta llegar al final del archivo)

```
try:
    with open('datos.txt','r') as f:
         for data in f:
               print(data)
except:
    pass
```

Escribir datos a un archivo:

Para escribir datos a un archivo se disponen de varios métodos, los cuales ilustraremos a continuación. Tenga en cuenta que el archivo deberá estar abierto en algún modo que permita la escritura **a, w, r+, a+, w+.**

NOTA: En los ejemplos se utilizará el modo a, dada las características de escritura destructiva del modo w.

1) Podemos utilizar la sentencia **print**, indicando que la salida se envía al archivo.

Ej12. Escribir sobre un archivo los datos que introduce el usuario.

```
try:
    with open('datos.txt','at') as f:
        nombre = input("Por favor indique su nombre  : ")
        edad = int(input("Por favor indique su edad    : "))
        print(f'{nombre},{edad}',file=f)
except:
    pass
```

2) Podemos utilizar el método **write**, este método necesita que lo que se va a escribir al archivo sea convertido a texto (si el archivo se abre en modo texto) o a byte (si el archivo se abre en modo binario) y retorna el número de caracteres (bytes) escritos.

Ej13: Escribir sobre un archivo

```
try:
    with open('datos.txt','a') as f:
        f.write('Esta es una línea de texto\n')
        f.write('Esta es otra línea de texto\n')
except:
    pass
```

Ej14: Escribir sobre un archivo, los datos contenidos en una tupla

```
try:
    with open('datos.txt','at') as f:
        datos = ('Ororo Munroe', 'Tormenta', 'Africa', 24)
        f.write(str(datos)+'\n')
except:
    pass
```

Ej15: Escribir sobre un archivo binario, los datos solicitados a un usuario

```
try:
    with open('datos.dat','ab') as f:
        nombre = input('Introduzca su nombre por favor : ')
        edad   = input('Su edad por favor              : ')
        s = (nombre + '<:>' + str(edad) + '\n').encode()
        print(s)
        f.write(s)
except:
    pass
```

3) Podemos utilizar el método **writelines**, que escribe al archivo todo el contenido en una lista.

Ej16: Escribir sobre un archivo binario, los datos contenidos en una lista

```
try:
    with open('datos.txt','a') as f:
        L = ['Esta es la primera línea\n','Esta es la segunda línea\n','E
sta es la tercera línea\n']
        f.writelines(L)
except:
    pass
```

Cerrar el archivo

Cerrar el archivo, libera la conexión con el sistema operativo y asegura los datos. Es una excelente práctica cerrar un archivo de datos cuando se termine de usar. En Python se debe utilizar el método **close**. Recordar que si se utiliza el manejador de contextos **with** el archivo se cierra de forma automática tan pronto se salga del ámbito del contexto.

Otras funciones y atributos de archivos

seek(desplazamiento, desde donde=0) : desplaza el apuntador del archivo el número de bytes especificado por **desplazamiento**, contados **desde donde**. Los valores permitidos para **desde donde** son: 0 (desde el principio del archivo), 1 (desde la posición actual), 2 (desde el fin del archivo). El valor por defecto es 0

detach(): Returns the separated raw stream from the buffer

fileno(): Retorna el número del manejador de archivos (filehandler), asignado por el sistema operativo.

flush(): Ejecuta una operación de limpiezas al buffer asociado al archivo, si existen datos sin escribir forza la escritura de los mismos.

isatty(): Retorna un valor lógico que indica si el archivo es interactivo o no.

next(): Al ser un archivo un iterable, el método next retorna el siguiente elemento del archivo.

readable(): Retorna un valor lógico que indica si se puede o no leer datos desde el archivo

seekable(): Retorna un valor lógico que indica si el archivo admite el cambio manual del apuntador a la posición del archivo.

tell() : devuelve un entero representando la posición en la cual se encuentra el apuntador del archivo, para archivos binarios, representa la distancia, en bytes, desde el inicio del archivo; para archivos de texto retorna un número que no tiene gran relevancia.

truncate(): Cambia el tamaño del archivo al tamaño específicado, con la consecuente pérdida de datos.

writable(): Retorna un valor lógico que indica si se puede o no escribir datos sobre el archivo.

Algunos atributos de interés:

closed: Retorna un valor lógico que indica si el archivo se encuentra cerrado

encoding: La codificación que el archivo está utilizando. Por ejemplo, utf8

mode: El modo en que el archivo fue abierto.

name : El nombre completo del archivo

newlines: Representa el manejo del newline en archivos, para compatibilidad con el formato universal; puede ser: \n, \r, \n\r, None

Ejemplos con archivos

A continuación, se incluye un programa que implementa la matriz CRUD sobre un archivo:

```
import os
import time
#
# Gestión de clientes mediante archivos, permite Adicion, actualización, bus
queda, eliminación y listado de clientes
#
#
nombreArchivo = 'clientes.dat'
nombreEmpresa = 'EMPRESA XYZ'

def cls():
    try:
        if os.name.upper()=='NT':
            os.system('cls')
        else:
            os.system('clear')
    except Exception:
        pass

def esperar(tiempo=3,msg=None):
    try:
        if msg!=None:
            msg = f'{msg}. Por favor espere {tiempo} segundos para continuar
'
            print(msg)
        time.sleep(tiempo)
    except Exception:
        pass

def shocli():
    try:
        sal = '\nError al listar los datos'
        cls()
        with open(nombreArchivo,'r') as f:
            print('{0:^123}\n'.format(nombreEmpresa))
            print('{0:^123}\n'.format('Reporte de clientes'))
            print('{0:<10} {1:>11} {2:<50} {3:>12} {4:<40}'.format('Tip. Ide
.','Num. Ide.','Apellidos y Nombres','Teléfono','Dirección'))
            for linea in f:
                datos = linea.replace('\n','').split(',')
```

```
                print(f'{datos[0]:^10} {datos[1]:>11} {datos[2]:<50} {datos[
3]:>12} {datos[4]:<40}')
            sal = '\nListado finalizado'
    except Exception:
        pass
    return sal

def addcli():
    try:
        cls()
        sal = '\nNo se pudieron guardar los datos'
        with open(nombreArchivo,'a') as f:
            tid=input("Por favor digite el tipo de Identificación       : ")
            nid=input("Por favor digite el número de identificación     : ")
            nom=input("Por favor digite el nombre                       : ")
            tel=input("Por favor digite el número de teléfono           : ")
            dcn=input("Por favor digite la dirección                    : ")
            print(f'{tid},{nid},{nom},{tel},{dcn}',file=f)
            sal = '\nDatos guardados exitosamente'
    except Exception:
        pass
    return sal

def seacli():
    try:
        cls()
        sw = False
        sal = '\nNo se encontró un registro que coincida con la información
buscada.'
        with open(nombreArchivo,'r') as f:
            bus = input("Digite el valor a buscar                       :
")
            tip = input("Buscará por N)ombre, I)dentificación o T)eléfono :
").lower()
            for pos,linea in enumerate(f):
                datos = linea.replace('\n','').split(',')
                if (tip=="n" and datos[2].upper()==bus.upper()) or (tip=="i"
 and datos[1]==bus) or (tip=="t" and datos[3]==bus):
                    sw=True
                    break
        if sw:
            sal = f'\nSe encontró el registro buscado en la línea {pos} del
archivo.'
    except Exception:
       pass
```

```
    return sal

def fincli(bus):
    try:
        sw = False
        pos = -10
        with open(nombreArchivo,'r') as f:
            for pos,linea in enumerate(f):
                datos = linea.replace('\n','').split(',')
                if datos[1]==bus:
                    sw=True
                    break
    except Exception:
        pass
    return (sw,pos)
#
#  Se implementaborrado físico. Esta operación merece discutirse
#
def delcli():
    try:
        sal = '\nError al eliminar los datos.'
        bus = input("Digite el número de documento                    : ")
        (sw,enc) = fincli(bus)
        if not sw:
            sal = "\nNo se encontró ningun registro que coincida con el crit
erio de búsqueda"
        else:
            with open(nombreArchivo,'r') as f:
                datos = f.readlines()
            linea = datos[enc].replace('\n','').split(',')
            print('*** DATOS DEL CLIENTE A ELIMINAR ***')
            print(f"\tTipo de Identificación       : {linea[0]}")
            print(f"\tNúmero de identificación     : {linea[1]}")
            print(f"\tNombre                       : {linea[2]}")
            print(f"\tNúmero de teléfono           : {linea[3]}")
            print(f"\tDirección                    : {linea[4]}")
            op = input('Confirma la eliminación del cliente [S/N] (NOTA: Est
e proceso es irreversible) : ').lower()
            if op=='s':
                datos.pop(enc)
                with open(nombreArchivo,'w') as f:
                    f.writelines(datos)
                sal = '\nDatos eliminados exitosamente.'
            else:
                sal = '\nOperación cancelada'
```

```
    except Exception:
        pass
    return sal

def updcli():
    try:
        sal = '\nError al actualizar los datos.'
        bus = input("Digite el número de documento                    : ")
        (sw,enc) = fincli(bus)
        if not sw:
            sal = "\nNo se encontró ningun registro que coincida con el crit
erio de búsqueda"
        else:
            tid=input("Por favor digite el tipo de Identificación       : ")
            nid=input("Por favor digite el número de identificación     : ")
            nom=input("Por favor digite el nombre                       : ")
            tel=input("Por favor digite el número de teléfono           : ")
            dcn=input("Por favor digite la dirección                    : ")
            with open(nombreArchivo,'r') as f:
                datos = f.readlines()
            datos[enc] = f'{tid},{nid},{nom},{tel},{dcn}\n'
            with open(nombreArchivo,'w') as f:
                f.writelines(datos)
            sal = '\nDatos guardados exitosamente.'
    except Exception:
        pass
    return sal

def main():
    op = "1"
    while op!="S":
        os.system("cls")
        print("\tOPCIONES DISPONIBLES\n")
        print("\t1. Adicionar Cliente")
        print("\t2. Buscar Cliente")
        print("\t3. Borrar Cliente")
        print("\t4. Actualizar Cliente")
        print("\t5. Listar Clientes")
        print("\tS. Salir")
        op = input("\n\tSeleccione su opción : ").upper()
        if op=="1":
            esperar(msg=addcli())
        elif op=="2":
            esperar(msg=seacli())
        elif op=="3":
```

```
            esperar(msg=delcli())
        elif op=="4":
            esperar(msg=updcli())
        elif op=="5":
            esperar(6,shocli())
        elif op!="S":
            esperar(2,"\n\tSeleccione una opción valida")
        else:
            print("\tHasta la vista ...")

main()
```