Lectura y escritura de archivos

Lógica y Algoritmia

Imagen tomada de: https://goo.gl/whLYqt

Archivos (1)

- Son una alternativa para guardar datos que perduren incluso después de cerrar el programa.
- Para abrir un archivo se utiliza la siguiente instrucción:

```
archivo = open("datos.txt","propiedad")
```

En donde “datos.txt” es la ruta y nombre del archivo y “propiedad” se reemplaza por una de las propiedades de la tabla siguiente.

Archivos (2)

- Los modos de apertura se seleccionan en función al uso que se le dará al archivo.

	r	r+	w	w+	a	a+
Leer	*	*		*		
Escribir		*	*	*	*	*
Crear			*	*	*	*
Truncar			*	*		
Ubicado al inicio	*	*	*	*		
Ubicado al final					*	*

Ejemplo:

```
archivo = open("datos.txt","r")
```

Archivos (3)

- Descripción:

Leer	Permite leer los datos del archivo
Escribir	Permite escribir datos en el archivo
Crear	Crea el archivo en caso de que no exista
Truncar	Al abrir el archivo todo el contenido es eliminado
Ubicado al inicio	La posición inicial será en el comienzo del archivo
Ubicado al final	La posición inicial será al final del archivo

Archivos (4)

- Se considera una buena práctica cerrar los archivos una vez se termine de usarlos.
- Un archivo que esté abierto en un programa no podrá ser modificado en otro hasta que no sea cerrado.
- Para cerrar un archivo se utiliza la siguiente instrucción en la variable que contiene el archivo:

```
.close()
```

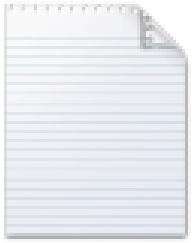
Archivos:
Ejemplo 1 (1) crear archivo

Código:

```
archivo = open("datos.txt","w")
archivo.close()
```

Ejecución:

Se crea el archivo vacío junto al programa:



datos.txt

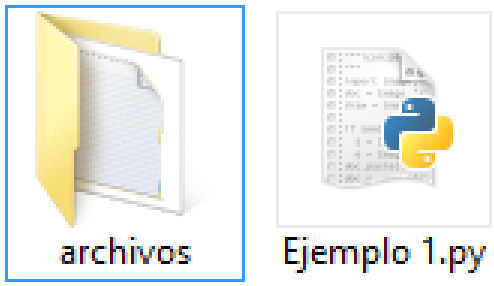
Archivos:
Ejemplo 1 (2) archivo con ruta

Código:

```
archivo = open("archivos/datos.txt","w")
archivo.close()
```

Ejecución:

Se crea el archivo vacío en la ruta establecida respecto al programa (la carpeta debe crearse antes de intentar crear el archivo):



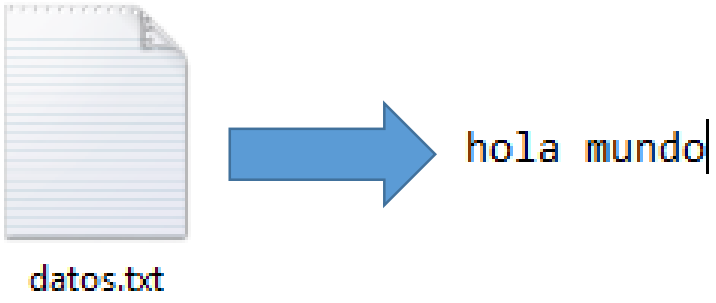
Archivos:
Ejemplo 2 escribir archivo con ruta

Código:

```
archivo = open("archivos/datos.txt","w")
texto="hola mundo"
archivo.write(texto)
archivo.close()
```

Ejecución:

Se crea el archivo vacío en la ruta establecida respecto al programa, se escribe en el archivo el texto ingresado y se cierra el archivo:



Archivos:
Ejemplo 3 leer archivo con ruta

Código:

```
archivo = open("archivos/datos.txt","r")
texto=archivo.read()
archivo.close()
print(texto)
```

Ejecución:

Se abre el archivo creado en el ejemplo 2, se lee todo su contenido y se imprime:

```
>>>
hola mundo
>>>
```

Archivos (5)

- Para realizar saltos de línea en un archivo es necesario agregar la siguiente expresión justo después de cada línea:

```
"\n"
```

  Esta expresión también es utilizada para realizar saltos de línea al ejecutar el comando “print”.

- Se puede almacenar el contenido de una lista separado por cambios de línea haciendo uso de la anterior expresión y el ciclo repetitivo “for”.

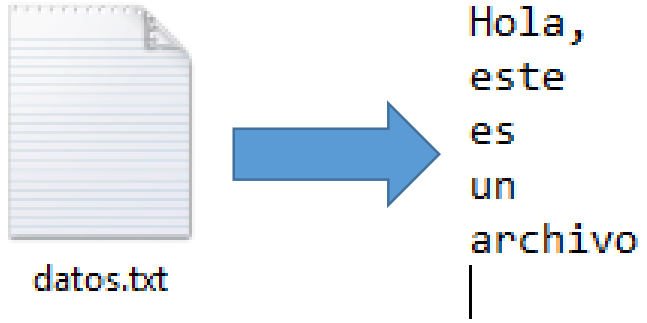
Archivos:
Ejemplo 4 escribir lista en archivo

Código:

```
archivo = open("archivos/datos.txt","w")
lista=["Hola,","este","es","un","archivo"]
for elemento in lista:
    archivo.write(elemento+"\n")
archivo.close()
```

Ejecución:

Se abre el archivo creado en el ejemplo 2, la propiedad "w" reemplaza todo el contenido del archivo al escribir:



5 líneas y 5 saltos de línea

Archivos:
Ejemplo 5 (1) leer lista de archivo

Código:

```
archivo = open("archivos/datos.txt","r")
lista=[]
for linea in archivo:
    lista.append(linea)
archivo.close()
print(lista)
```

Ejecución:

```
>>>
['Hola,\n', 'este\n', 'es\n', 'un\n', 'archivo\n']
>>>
```

Cada línea es almacenada incluyendo la expresión que representa salto de línea.

Archivos:
Ejemplo 5 (2) leer lista de archivo

Código:

```
archivo = open("archivos/datos.txt","r")
lista=[]
for linea in archivo:
    lista.append(linea.strip())
archivo.close()
print(lista)
```

Ejecución:

```
>>>
['Hola,', 'este', 'es', 'un', 'archivo']
>>>
```

Para solucionarlo se agrega la función “strip” a cada línea antes de almacenarla en la lista.

Archivos:
Ejemplo 6(1) notas de estudiantes

- Se entrega un archivo de texto llamado “notas.txt” el cual contiene las notas con valores de 0 a 5, de cuatro estudiantes en tres materias, cada línea representa una de las materias en donde se encuentran las notas de los estudiantes separadas por espacios:

	E1	E2	E3	E4
Cálculo	4.5	2.3	4.3	5.0
Geometría	3.3	3.6	1.6	4.5
Química	5.0	1.0	2.3	4.7

Archivos:
Ejemplo 6(2) notas de estudiantes

- El objetivo es almacenar la información en un arreglo bidimensional, calcular el promedio por materia y el promedio por estudiante.

	E1	E2	E3	E4	
Cálculo	4.5	2.3	4.3	5.0	X
Geometría	3.3	3.6	1.6	4.5	X
Química	5.0	1.0	2.3	4.7	X
	X	X	X	X	

Archivos:
Ejemplo 6(3) notas de estudiantes

- Para separar por espacios cada uno de los elementos del arreglo se utiliza la función “Split”:

```
import numpy as np
archivo = open("archivos/notas.txt","r")
arreglo=np.zeros(shape=(3,4))
fila=0 #cantidad de líneas
for linea in archivo:
    elementos=linea.split()
    arreglo[fila]=elementos
    fila=fila+1
archivo.close()
```

Archivos:
Ejemplo 6(4) notas de estudiantes

- Cada una de las filas representa una materia, se deben recorrer una por una y obtener el promedio.

```
#arreglo es el arreglo bidimensional
print("Promedio por materias")
for materia in arreglo:
    total=0 #suma total de notas
    cNotas=0 #cantidad de notas
    for nota in materia:
        total=total+nota
        cNotas=cNotas+1
    promedio=total/cNotas
    print(promedio)
```

Archivos:
Ejemplo 6(5) notas de estudiantes

- Para obtener el promedio de las columnas se toma como opción obtener la transpuesta del arreglo bidimensional y realizar la misma tarea anterior:

Arreglo de Notas				
	E1	**E2**	**E3**	**E4**
Cálculo	**4.5**	**2.3**	**4.3**	**5.0**
Geometría	**3.3**	**3.6**	**1.6**	**4.5**
Química	**5.0**	**1.0**	**2.3**	**4.7**

Arreglo de notas transpuesto			
	Cálculo	**Geometría**	**Química**
E1	**4.5**	**3.3**	**5.0**
E2	**2.3**	**3.6**	**1.0**
E3	**4.3**	**1.6**	**2.3**
E4	**5.0**	**4.5**	**4.7**

Archivos:
Ejemplo 6(4) notas de estudiantes

- Cada una de las filas representa un estudiante, se deben recorrer una por una y obtener el promedio.

```
#arreglo.transpose() es el transpuesto
print("Promedio por estudiante")
for estudiante in arreglo.transpose():
    total=0 #suma total de notas
    cNotas=0 #cantidad de notas
    for nota in estudiante:
        total=total+nota
        cNotas=cNotas+1
    promedio=total/cNotas
    print(promedio)
```

Archivos:
Ejemplo 6(5) notas de estudiantes

- Los resultados de la ejecución muestran los promedios en el orden esperado:

Cálculo
Geometría
Química

Estudiante 1
Estudiante 2
Estudiante 3
Estudiante 4

```
>>>
Promedio por materias
4.025
3.25
3.25
Promedio por estudiante
4.26666666667
2.3
2.73333333333
4.73333333333
>>>
```

Referencias Bibliográficas

[1] Zelle, John M. Python Programming an Introduction to Computer Science. -- 2nd ed. -- Washington : Franklin, Beedle & Associates Inc, 2010.

[2] Dawson, Michael. Python Programming for the Absolute Beginner. -- 3th ed. -- Australia : Cengage Learning, 2010.

[3] Diana Teresa Gómez Forero. Presentación de clase: Lectura y escritura de archivos. Universidad Pontificia Bolivariana Seccional Bucaramanga. 2015.