

Pensamiento Computacional

Archivos

archivos

Los archivos te permiten almacenar datos

permanentemente

¿Por qué archivos?



Los programas usan datos de forma temporal.



Cuando un programa termina su ejecución, los datos que usó desaparecen.



La forma en que podemos almacenar datos de forma permanente es usando archivos



Los archivos se dividen generalmente en dos tipos

Binarios
De texto

Recorrido por el sistema operativo

- Los archivos se guardan directorios
- para llegar a ellos se debe seguir un camino desde el directorio raíz
 - PATH
 - C:\Users\AnaRaquel\Presentaciones\ManejoArchivos.pptx

Ruta hacia el archivo

Nombre del
archivo

Extensión (tipo)

Cosas a cuidar

- Para acceder correctamente a un archivo tiene que:
 - Estar en nuestro directorio de trabajo actual
 - `archivo = open("archivo.txt", "r")`
 - Usar la ruta completa desde la raíz
 - `archivo = open("C:\\Usuarios\\Thonny\\archivo.txt","r")`
 - Usar una ruta relativa desde el directorio actual
 - `archivo = open("../files/archivo.txt", "r")`
- Las librerías `os` y `sys` nos dan algunas herramientas para trabajar con directorios

Entrada y salida de texto

- OJO: La realidad es que todo está almacenado en binario (bytes) pero existe una clase de entrada y salida de texto que nos facilita la vida para usar archivos
- Para abrir un archivo
`fileVariable = open(filename, mode)`
- fileName es el nombre del archivo con todo y ruta
- mode indica qué vamos a hacer con este archivo

Modos de acceso

<i>Mode</i>	<i>Description</i>
"r"	Opens a file for reading.
"w"	Opens a new file for writing. If the file already exists, its old contents are destroyed.
"a"	Opens a file for appending data from the end of the file.
"rb"	Opens a file for reading binary data.
"wb"	Opens a file for writing binary data.

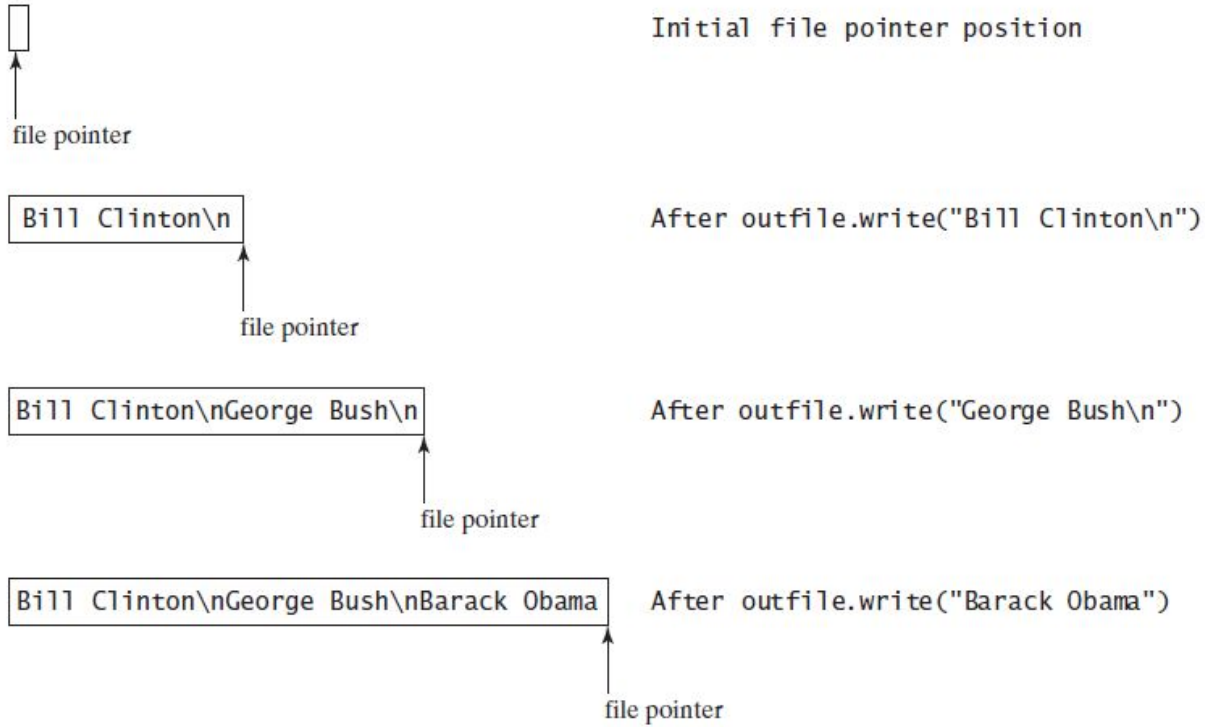
Ejemplo

- Abrir el archivo “Scores.txt” que está en mi directorio de trabajo
- `inFile = open(“Scores.txt”, “r”)`
- Abrir el archivo “Scores.txt” usando la ruta completa
- `inFile = open(“c:\\usuarios\\Thonny\\Scores.txt”, “r”)`
- OJO: nótese el uso de los caracteres especiales

Ejemplo escribiendo en un archivo.

```
def main():  
    # Open file for output  
    outfile = open("../files/Presidents.txt", "w")  
  
    # Write data to the file  
    outfile.write("Bill Clinton\n")  
    outfile.write("George Bush\n")  
    outfile.write("Barack Obama")  
  
    outfile.close() # Close the output file  
  
main() # Call the main function
```

Ejemplo



Cosas a cuidar

- Cuando usamos la función `print()` automáticamente se añade un salto de línea al final
- Cuando usamos el método `write` para escribir en un archivo, los saltos de línea se deben añadir de forma explícita.
- **Cuando abrimos un archivo en modo “w”, si el archivo ya existe, el contenido se pierde.**

Recuerda:

Formas de abrir un archivo

r	w	a
Solo lectura	Escritura	Agregar

Dos cosas que NO se pueden hacer con archivos en Python

Uno

No se puede acceder a una línea en particular.

La lectura es secuencial, elemento por elemento.

Dos

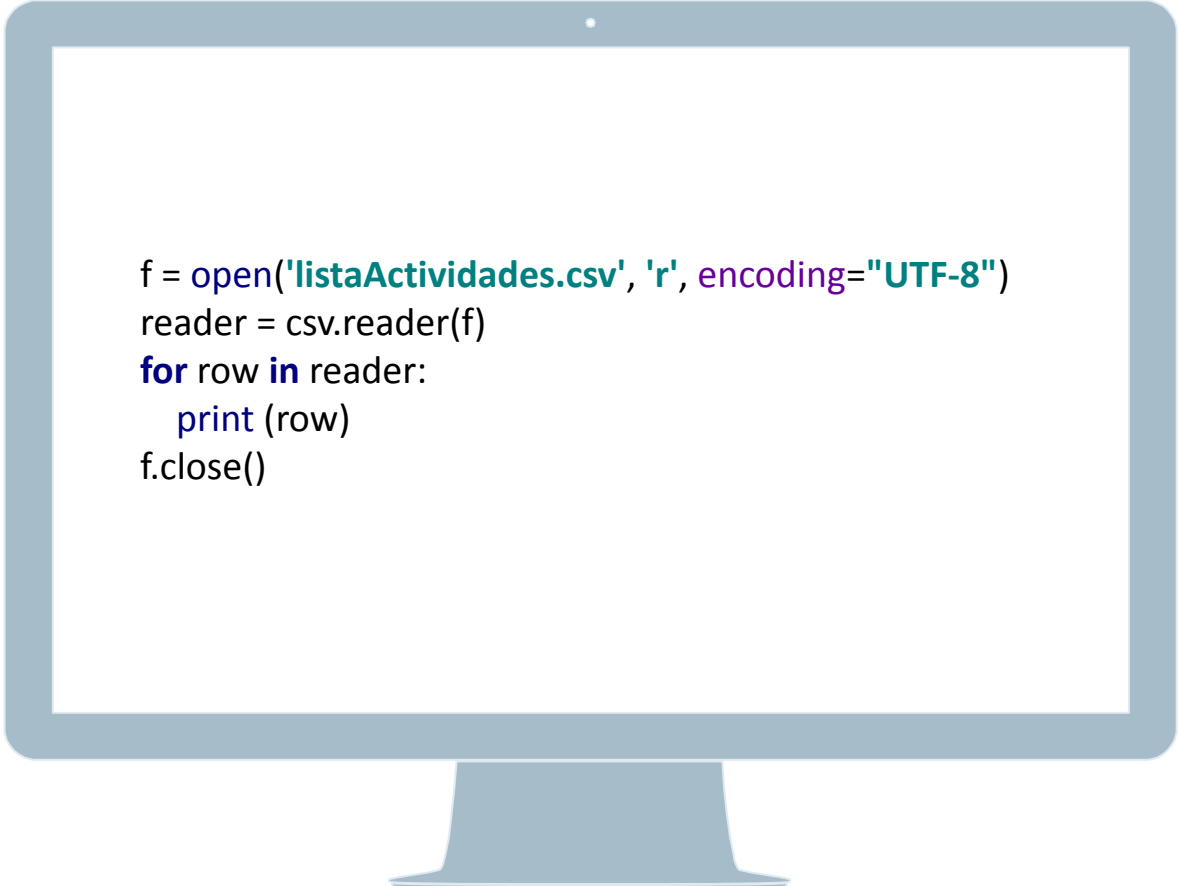
No se puede borrar o modificar una línea en particular del archivo.

Abrimos el
archivo en modo
lectura 'r'

El encoding es
para que pueda
leer acentos y
caracteres raros.
Úsalo sólo si tu
archivo de origen
tiene acentos o
caracteres raros.

Creamos un
“lector” de
archivo csv.

Al abrirlo se
recorre por
 renglones.



```
f = open('listaActividades.csv', 'r', encoding="UTF-8")
reader = csv.reader(f)
for row in reader:
    print (row)
f.close()
```

Nos conviene generar listas con los renglones del archivo.

- Recordando: la forma general de un ciclo `for` es la siguiente:

`for elemento in` Lista:
acciones a realizar

`elemento` es sólo una variable que nos sirve para ir tomando en cada iteración un valor de la lista.

Si recorremos una lista de listas, tendremos que usar referencia a dos posiciones del elemento, es decir, tener un índice para los renglones y otro para las columnas.

Ejemplo:



Aquí estoy generando
Una lista de listas con
el contenido del archivo.

```
f = open('listaMini.csv', 'r', encoding="UTF-8")
reader = csv.reader(f)
#creo la lista de listas
lista=[]
for row in reader:
    print (row)
    lista=lista+[row]
print (lista)

#reviso el número de elementos en la lista
tam=len(lista)
```


Matriz: Lista de listas

- Como recordarás Python no maneja el concepto de matriz, al igual que otros lenguajes, una matriz resulta la interpretación de una lista que contiene listas.
- Cada lista es un renglón de la matriz.
- Accedes al contenido de cada lista, pensando en el número de lista a *accesar* y luego en el número de elemento a utilizar de dicha lista.

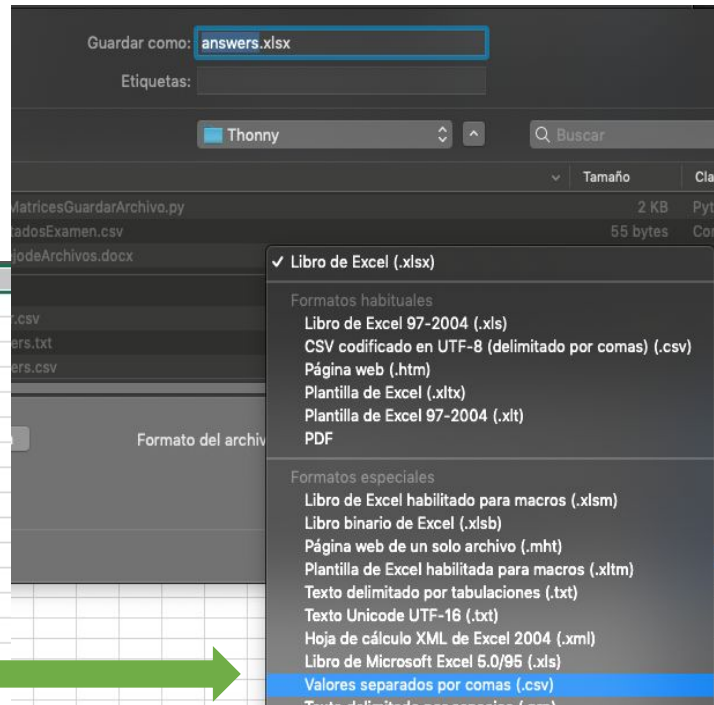
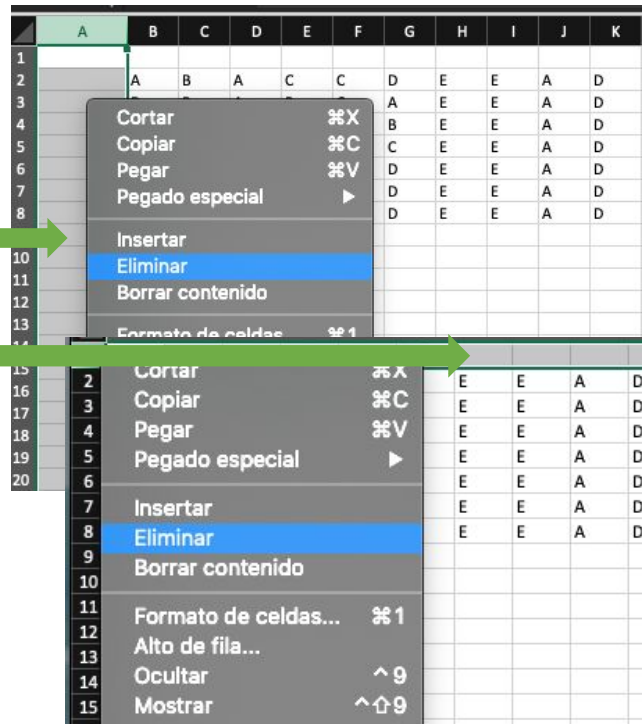
Recuerdas este
reto de listas de
listas:

```
def main():  
    # Students' answers to the questions  
    answers = [  
        ['A', 'B', 'A', 'C', 'C', 'D', 'E', 'E', 'A', 'D'],  
        ['D', 'B', 'A', 'B', 'C', 'A', 'E', 'E', 'A', 'D'],  
        ['E', 'D', 'D', 'A', 'C', 'B', 'E', 'E', 'A', 'D'],  
        ['C', 'B', 'A', 'E', 'D', 'C', 'E', 'E', 'A', 'D'],  
        ['A', 'B', 'D', 'C', 'C', 'D', 'E', 'E', 'A', 'D'],  
        ['B', 'B', 'E', 'C', 'C', 'D', 'E', 'E', 'A', 'D'],  
        ['B', 'B', 'A', 'C', 'C', 'D', 'E', 'E', 'A', 'D'],  
        ['E', 'B', 'E', 'C', 'C', 'D', 'E', 'E', 'A', 'D']]  
  
    # Key to the questions  
    keys = ['D', 'B', 'D', 'C', 'C', 'D', 'A', 'E', 'A', 'D']  
  
    # Grade all answers  
    for i in range(len(answers)):  
        # Grade one student  
        correctCount = 0  
        for j in range(len(answers[i])):  
            if answers[i][j] == keys[j]:  
                correctCount += 1  
  
        print("Student", i, "'s correct count is", correctCount)  
  
main() # Call the main function
```

varios archivos automatizados en un archivo que tenga las respuestas de los alumnos

Limpieza de datos:

1. Abre el archivo answers.xls
2. Elimina las columnas y renglones de más
3. Guarda el archivo en el mismo directorio que el proyecto pero con la extensión .csv



```

1  """
2  CALIFICACIONES
3  """
4  #abrir archivo de respuestas
5  f=open("answers.csv","r")
6  #Con el archivo abierto, voy a guardarlo en una lista para poder manipularlo
7  respuestas=[]
8  for row in f:
9      print(repr(row)) #chechar qué tiene el archivo renglón x renglón
10     respuestas.append(row) #guardo renglón por renglón en la lista <respuestas>
11 f.close()
12 #

```

Shell

```

>>> %Run RetoMatricesGuardarArchivo.py

```

```

'A,B,A,C,C,D,E,E,A,D\n'
'D,B,A,B,C,A,E,E,A,D\n'
'E,D,D,A,C,B,E,E,A,D\n'
'C,B,A,E,D,C,E,E,A,D\n'
'A,B,D,C,C,D,E,E,A,D\n'
'B,B,E,C,C,D,E,E,A,D\n'
'E,B,E,C,C,D,E,E,A,D\n'

```

```

answers = [
    ['A', 'B', 'A', 'C', 'C', 'D', 'E', 'E', 'A', 'D'],
    ['D', 'B', 'A', 'B', 'C', 'A', 'E', 'E', 'A', 'D'],
    ['E', 'D', 'D', 'A', 'C', 'B', 'E', 'E', 'A', 'D'],
    ['C', 'B', 'A', 'E', 'D', 'C', 'E', 'E', 'A', 'D'],
    ['A', 'B', 'D', 'C', 'C', 'D', 'E', 'E', 'A', 'D'],
    ['B', 'B', 'E', 'C', 'C', 'D', 'E', 'E', 'A', 'D'],
    ['B', 'B', 'A', 'C', 'C', 'D', 'E', 'E', 'A', 'D'],
    ['E', 'B', 'E', 'C', 'C', 'D', 'E', 'E', 'A', 'D']]

```

Como podemos ver, la lista todavía no tiene los datos como los necesitamos para la comparación. Necesitan ser strings independientes dentro de cada lista.

Agregamos
.strip().split(",")
Para que en cada
coma, sea interpretada
como un separador de
strings.

- El resultado es la
lista de listas de
strings (letras)
que necesito para
comparar.

```
1  """
2  CALIFICACIONES
3  """
4  #abrir archivo de respuestas
5  f=open("answers.csv","r")
6  #Con el archivo abierto, voy a guardarlo en una lista para poder manipularlo
7  respuestas=[]
8  for row in f:
9      #print(repr(row)) #chechar qué tiene el archivo renglón x renglón
10     respuestas.append(row) #guardo renglón por renglón en la lista <respuestas>
11 f.close() #se debe cerrar el archivo.
12
13 lista=[]
14 for recorre in respuestas:
15     lista.append(recorre.strip().split(","))
16 print(lista) #revisa cómo se imprime el archivo convertido a lista
17 #gracias al strip, split, los datos ya están almacenados como strings dentro de listas
18
```

Shell >

Python 3.7.2 (bundled)

>>> %Run RetoMatricesGuardarArchivo.py

```
[[['A', 'B', 'A', 'C', 'C', 'D', 'E', 'E', 'A', 'D'], ['D', 'B', 'A', 'B', 'C', 'A', 'E',
'E', 'A', 'D'], ['E', 'D', 'D', 'A', 'C', 'B', 'E', 'E', 'A', 'D'], ['C', 'B', 'A', 'E',
'D', 'C', 'E', 'E', 'A', 'D'], ['A', 'B', 'D', 'C', 'C', 'D', 'E', 'E', 'A', 'D'], ['B', 'A
'B', 'E', 'C', 'C', 'D', 'E', 'E', 'A', 'D'], ['E', 'B', 'E', 'C', 'C', 'D', 'E', 'E', 'A
', 'D']]
```


A photograph of a dense urban skyline at night. Numerous high-rise buildings are visible, their windows glowing with warm yellow and orange light. Some buildings have distinctive architectural features, like a tall, thin tower with a series of vertical light bars. The sky is dark, and the overall scene conveys a sense of a bustling, modern city.

El resto es simple programación...

```

5 #print(lista) #revisa cómo se imprime el archivo convertido a lista
7 #gracias al strip, split, los datos ya están almacenados como strings dentro de listas
8
9 #llave de las respuestas
10 llave= ["D", "B", "D", "C", "C", "D", "A", "E", "A", "D"]
11 #Imprimo un título de columna
12 print("\t Número de Estudiante \t\t\t Número de aciertos")
13 print("\t -----")
14 #CALIFICAR LAS respuestas
15 for i in range (len(lista)):
16     #voy estudiante por estudiante
17     resultados="" #usaremos este <string> para guardar los resultados
18     respuestasCorrectas=0
19     for j in range(len(lista[i])):
20         if lista[i][j]==llave[j]:
21             respuestasCorrectas+=1
22     #imprimo resultados de CADA estudiante
23     print (f"\t\t {i+1}\t\t\t {respuestasCorrectas}")
24

```

Todo se resuelve en
una simple
comparación

Número de Estudiante	Número de aciertos
-----	-----
1	7
2	6
3	5
4	4
5	8
6	7
7	7

Observa lo que se debe agregar al programa:

- 1- Creamos el encabezado del archivo en un string.
- 2- usamos with para indicar con qué archivo trabajaremos con el atributo "w". Y es nuestro archivo de salida: outputFile
- 3- Lo primero que escribo en el archivo es el encabezado que yo mismo cree.
- 4- En el mismo código que ya tenemos para contabilizar los aciertos, agregamos el string <resultados> que acumulará los dos valores, el número de estudiante y sus respuestas correctas para cada estudiante.
- 5- Cerramos el archivo.

```
print("\t Numero de Estudiante \t\t Numero de aciertos")
print("\t -----")
#Codigo para ESCRIBIR en un archivo los resultados obtenidos:
encabezadoResultados="No Estudiante, No Aciertos\n"
with open("../Thonny/resultadosExamen.csv", "w") as outputFile:
    outputFile.write(encabezadoResultados)
    for i in range (len(lista)):
        #voy estudiante por estudiante
        resultados="" #usaremos este <string> para guardar los resultados
                        #para luego GRABARLOS en un archivo
        respuestasCorrectas=0
        for j in range(len(lista[i])):
            if lista[i][j]==llave[j]:
                respuestasCorrectas+=1
        #imprimo resultados de CADA estudiante
        print (f"\t\t {i+1}\t\t\t{respuestasCorrectas}")
        resultados+=str(i)+", "+str(respuestasCorrectas)+"\n"
        outputFile.write(resultados)
#
#como podrás observar, la salida quedó sólo en pantalla.
#Si queremos guardar los resultados en un archivo hay que guardarlos en una lista
#y cuidaremos tener un encabezado. Para ello en el código agregamos un string en blanco
#<resultados>, donde vamos a ir guardando los resultados por alumno
#
```




Ejercicios

- Realiza este proyecto y logra guardar el archivo con los resultados.
- Prueba abrir tu archivo en EXCEL
- Si lo logras, ¡¡prueba superada!!!

Qué aprendimos...



archivos

Me permite la manipulación de archivos de texto o formato csv (coma separated values).



Modos

Existen tres formas de “abrir” el archivo dependiendo de qué haremos con él.



recorrido

Todo archive será manipulado como lista de listas en python y el recorrido es secuencial.