

Programación
Licenciatura en Agroinformática
Licenciatura en Bioinformática

Tema 07: Archivos

Almacenamiento de datos

- Las variables y datos utilizados en un programa se almacenan en la memoria RAM.
- El acceso a estos datos es muy rápido, lo cual favorece la eficiencia del programa.
- Almacenar en memoria tiene dos limitaciones:
 - Tamaño de la memoria RAM.
 - Temporalidad.

Flujo de datos

- Durante su ejecución, un programa puede recibir y devolver datos del usuario o del sistema.
- Este movimiento se conoce como **flujo** de datos.
- Hasta el momento, la recepción y devolución de datos fue, principalmente, por consola.
- Estos flujos se conocen como **entrada y salida estándar** (STDIN y STDOUT).

Archivos

- Un archivo o fichero es una colección de datos almacenada en la memoria secundaria (por ej: disco).
- Una vez escritos, los datos almacenados en un archivo **no se pierden** como sucede en la memoria RAM.
- Los datos en un archivo **deben** estar **relacionados entre sí y organizados** (por ej, datos en filas, cada columna es un campo distinto).

Tipos de archivos

- En programación, se suele trabajar con dos tipos de archivos:
 - Archivos de texto:
 - Almacenan caracteres ASCII.
 - Son legibles por las personas
 - Archivos binarios:
 - Almacenan bytes.
 - Solo son interpretables por computadoras.
 - Se utilizan para almacenar programas, imágenes, audios, ...

Archivo binario

```
Environments
artic-ncov2019
medaka
pangolin
qiime2-2022.2
trycycler
```

ANICALculator_v1
ANVio
FALite.pm
FastANI
FastTree
FastTreeDbL
FastTreeMP
GATK
GTDBTk
KEGG_mapping
KrakenTools
KronaTools-2.7
MaxBin-2.2.5
Prodigal

189PNG
1A
000000
IHDR000000S000000D9000000008F90J000000
d880000000pHYs0000.#0000.#01x\A?
v000000019tEXtSoftware00www.inkscape.org98EE
00IDATx9C\AC}\EB\BA兜
ED08\A9^\97\F3\FEO\BAW\F7L0C:? \A4!
EC\BD\F0v})g:600!t019!
F2\F8\A7\AA*0015\CD0C\8A\F5\DCY\FC*0000~015@`0F\F
E5\F9\83R\FE\8Ek"Y\B0\940017\88\C4G\F9\984\FB\BD\F
Z\A00F\88b\8C7\CE\F1\C68\DFx\9Fo\9C\E7\CB>\E3\C48C
DA\DF\C4\DF\D901F\F8\C701F0F\E0\8F?\FE\81?
FE\F8003\BF~\FD011\E7\E7\F30017~\FD\FA\85\E7\F3017ZkFY
00T015\E3<q\9EV\EF\F3\FD\C6y\BEq\BE\DFx\BF\DF8\DF
8E\E3\C00F^\E7\BF~\E1\F9x\E0\F9|
EO\E94007F\F4\8E\A301Fh\AD\B4@601EE\D69x/
AE\EF\F7^00\91\80†\F4\D22F\F8\9D\84\BA\FB[U\83/
D\8C017[k\F1\F7]\B9\9F\EA%0C#\82018/,A7y\DS\EF>m\F

Archivos de texto: acceso secuencial

- En un archivo de texto, los datos se escriben y acceden consecutivamente.
- Para acceder al registro n es necesario pasar por los $n-1$ registros anteriores.
- Al final, el sistema añade una marca que indica que no hay más datos (End Of File o EOF)

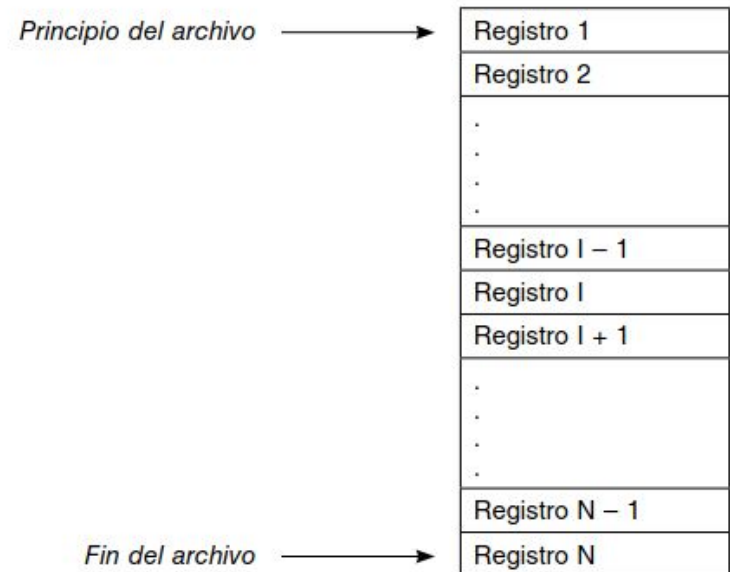


Figura 9.6. Organización secuencial.

Operaciones con archivos

- Creación.
- Consulta.
- Actualización (altas, bajas, modificación, consulta)
- Destrucción (borrado).
- Fusión y separación.

Creación

- Es la primera operación que se realiza sobre un archivo.
- La creación exige definir la localización del archivo y reservar el espacio en el soporte de almacenamiento.
- No es excluyente, pero es buena práctica saber la organización y estructura que tendrán los datos.

Consulta

- Implica acceder al archivo y chequear su contenido.
- No se realiza ninguna modificación sobre los datos.

Actualización

- Se modifica el contenido de un archivo.
- Las modificaciones más comunes incluyen:
 - Agregar registros.
 - Eliminar registros.
 - Cambiar un campo o un registro completo.

Destrucción



- Es la operación inversa a la creación.
- El archivo es eliminado del disco y el espacio que ocupaba liberado.
- Dependiendo del sistema operativo y de la forma de eliminación empleada, este proceso puede ser irreversible.

Fusión y separación

- Son operaciones que involucran más de un archivo.
- En la reunión, varios archivos se unen en uno solo.
- En la separación, se crean nuevos archivos con subsets de datos del archivo original.

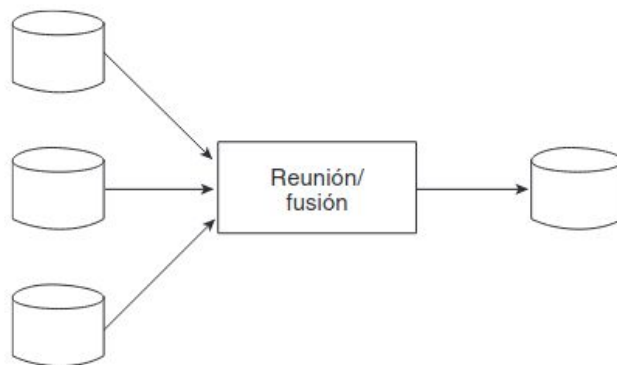


Figura 9.14. Fusión de archivos.

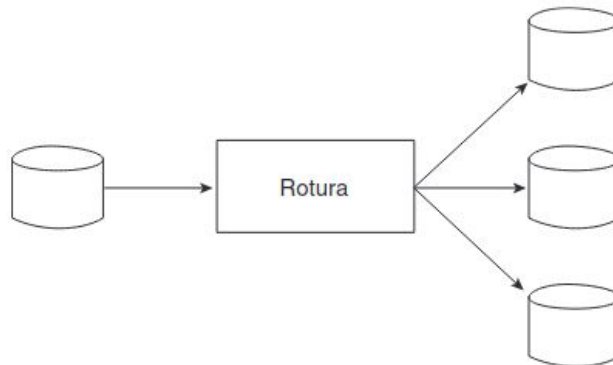


Figura 9.15. Rotura de un archivo.

Abriendo archivos en Python

- La función `open` permite abrir archivos.
- El primer parámetro es el nombre de un archivo (como texto o almacenado en una variable).
- El segundo parámetro es el modo en que se abrirá el archivo.

'r': read (lectura) *

't': texto *

'w': write (escritura)

'b': binario

'a': append (agregar)

'x': create (crear)

Abriendo archivos en Python

- Existen dos formas de abrir un archivo:

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

```
...
```

```
archivo.close()
```

```
with open("archivo.txt", "r") as archivo:
```

```
...
```

Leyendo archivos en Python

- Hay 3 alternativas para leer un archivo:
 - La función `readline`, que lee solo una línea
 - La función `readlines`, que lee todas las líneas del archivo.
 - Recorrer la variable “archivo” con un bucle `for`.

Leyendo archivos de texto

```
l = archivo.readline() # lee una línea
```

```
todas_las_lineas = archivo.readlines()
```

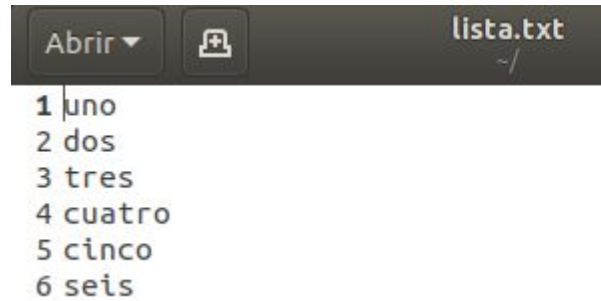
```
for l in archivo:
```

```
    # lee cada línea en archivo y la guarda en "l"
```

Leyendo archivos en Python

- Cuando se abre un archivo, el sistema crea una variable del tipo **puntero** que indica que línea se está leyendo.
- Cada vez que se lee una línea (con cualquiera de las alternativas), es imposible volver atrás.
- Si se usan la función `readlines` o el bucle `for`, se alcanza el final del archivo

Leyendo archivos en Python



A screenshot of a text editor window. The title bar shows 'Abrir' with a dropdown arrow, a file icon, and the filename 'lista.txt' with a tilde '~/' indicating the current directory. The text area contains a numbered list from 1 to 6, each followed by a Spanish word: '1 uno', '2 dos', '3 tres', '4 cuatro', '5 cinco', and '6 seis'.

```
1 uno  
2 dos  
3 tres  
4 cuatro  
5 cinco  
6 seis
```

```
with open("lista.txt", "r") as archivo:  
    l = archivo.readline()  
    print(l)  
    archivo.readlines()  
    l = archivo.readline()  
    print(l)
```



A terminal window showing the execution of a Python script. The prompt is '(base) matias@rfgenom002:~\$' and the command 'python test.py' has been entered. The output of the script is 'uno'.

```
(base) matias@rfgenom002:~$ python test.py  
uno
```

Lidiando con fines de líneas..

```
with open("lista.txt", "r") as archivo:  
    for l in archivo:  
        print(l)
```

```
(base) matias@rfgenom002:~$ python test.py  
uno  
  
dos  
  
tres  
  
cuatro  
  
cinco  
  
seis
```

Lidiando con fines de líneas..

```
with open("lista.txt", "r") as archivo:  
    for l in archivo:  
        print(l)
```

```
with open("lista.txt", "r") as archivo:  
    lineas = archivo.readlines()  
    print(lineas)
```

```
(base) matias@rfgenom002:~$ python test.py  
uno  
  
dos  
  
tres  
  
cuatro  
  
cinco  
  
seis  
  
['uno\n', 'dos\n', 'tres\n', 'cuatro\n', 'cinco\n', 'seis\n']
```

Lidiando con fines de líneas...

```
with open("lista.txt", "r") as archivo:
    for l in archivo:
        l = l.rstrip()
        print(l)
```

```
(base) matias@rfgenom002:~$ python test.py
uno
dos
tres
cuatro
cinco
seis
```

Escribiendo en un archivo

```
with open("archivo.txt", "w") as archivo:  
    archivo.write("Programacion.\n")  
  
var = "Biologia."  
archivo.write(var)
```

Valores separados por comas (CSV)


- Un formato de archivo de texto muy utilizado para almacenar datos el CSV.
- Estos archivos representan tablas, donde cada columna está delimitada por un carácter (por ej.: comas).
- Para leer un archivo en este formato es necesario leer cada línea y separarla según el carácter delimitador.

Valores separados por comas (CSV)

```
with open("archivo.csv", "r") as archivo:  
    for l in archivo:  
        l = l.rstrip()  
        valores = l.split(",")  
        ...
```

Ejemplo 1

Dado el siguiente archivo, ¿de qué raza es la vaca con mayor aumento en el número de células somáticas?



```
1 Caravana,Raza,CS_Anterior,CS
2 6441,H0,160,43
3 5995,XB,82,236
4 6084,H0,136,19
5 5646,H0,27,227
6 6270,H0,112,36
7 6409,XB,44,174
8 6230,XB,63,33
9 6239,H0,27,16
```

Ejemplo 1

Inicializar mayor en -inf y raza como ""

Leer el archivo, línea por línea.

Separar la línea por comas.

Restar la columna 4 menos la columna 3.

Si la resta es mayor al valor máximo guardado:

 raza = columna 2; mayor = resta.

Devolver resta.

Ejemplo 1

```
mayor = float("-inf")
raza = ""

with open("celulas.txt", "r") as archivo:
    archivo.readline()
    for linea in archivo:
        linea = linea.rstrip()
        valores = linea.split(",")
        if (valores[3] - valores[2]) > mayor:
            mayor = valores[3] - valores[2]
            raza = valores[1]

print ("La raza del animal con mayor aumento de CS es:", raza)
```

Ejemplo 2

“casos.tsv” es un archivo separado por tabulaciones que presenta una lista de casos de COVID en distintos países, cada millón de habitantes. Construya un gráfico de barras a partir de los datos brindados

```
1 Brasil 161472
2 Argentina 211118
3 Colombia 122457
4 Chile 242034
5 Peru 123176
6 Bolivia 92458
7 Ecuador 55543
8 Uruguay 282482
9 Paraguay 98146
10 Venezuela 18622|
```

Ejemplo 2

Crear lista de países y de cantidad de casos

Cargar archivo y leer por líneas.

Separar línea por tabulación.

Agregar primer campo a la lista de países

Agregar segundo campo a la lista de cantidades

Graficar

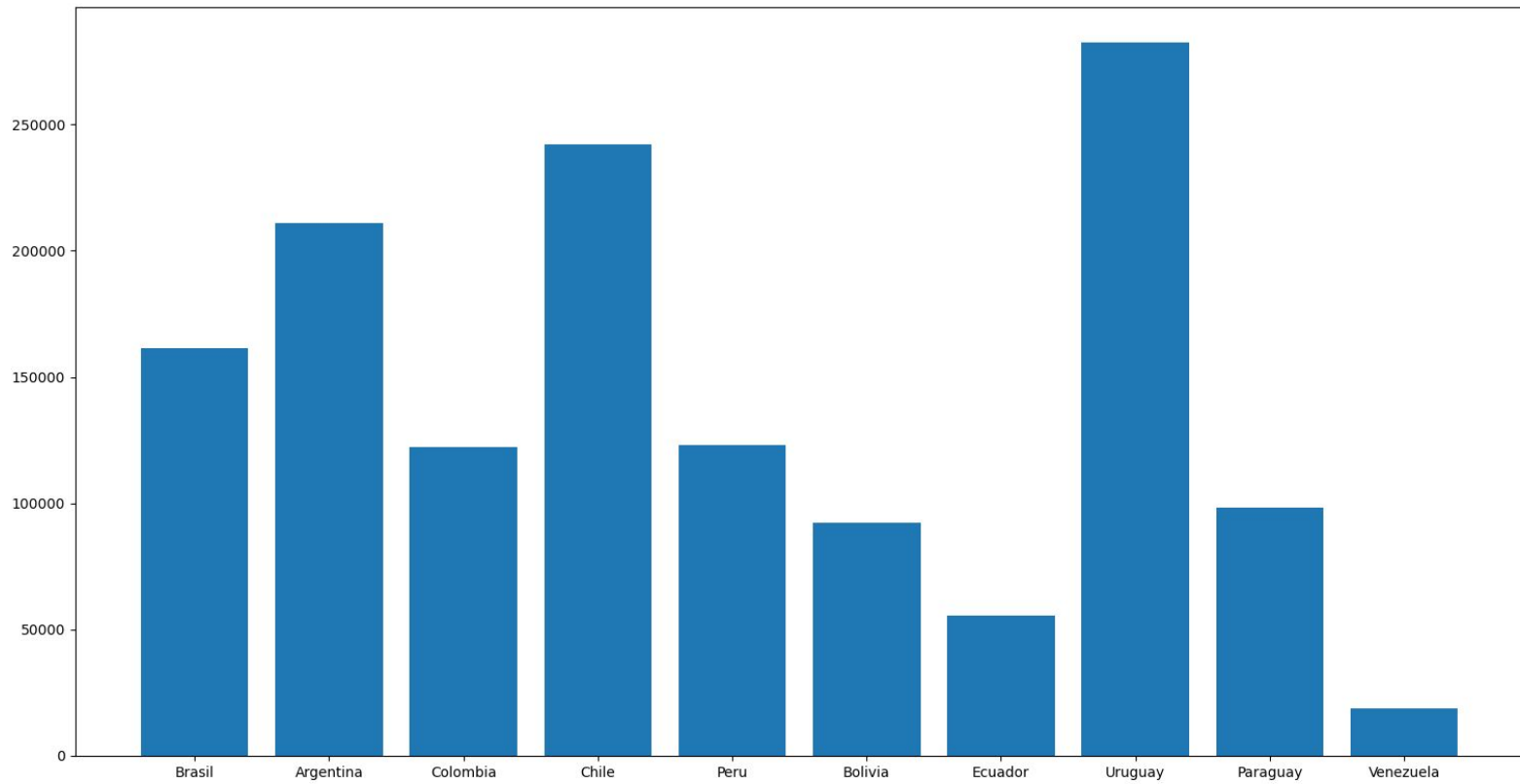
Ejemplo 2

```
import matplotlib.pyplot as plt

países = []
casos = []
with open("casos.tsv", "r") as archivo:
    for linea in archivo:
        linea = linea.rstrip()
        campos = linea.split("\t")
        países.append(campos[0])
        casos.append(int(campos[1]))

plt.bar(países, casos)
plt.show()
```

Ejemplo 2



Ejemplo 3

A partir del archivo del ejemplo anterior, guardar en un archivo nuevo el promedio de casos en todos los países listados.

Ejemplo 3

Inicializar suma=0 y cantidad=0

Cargar archivo y leer por líneas.

Separar línea por tabulación.

Sumar la segunda columna a suma y sumar 1 a cantidad

Abrir archivo de salida

Escribir (suma/cantidad)

Ejemplo 3

```
suma = 0
cantidad = 0
with open("casos.tsv", "r") as entrada:
    for linea in entrada:
        linea = linea.rstrip()
        campos = linea.split("\t")
        suma += int(campos[1])
        cantidad += 1

promedio = suma/cantidad
with open("promedio.txt", "w") as salida:
    salida.write(str(promedio))
```