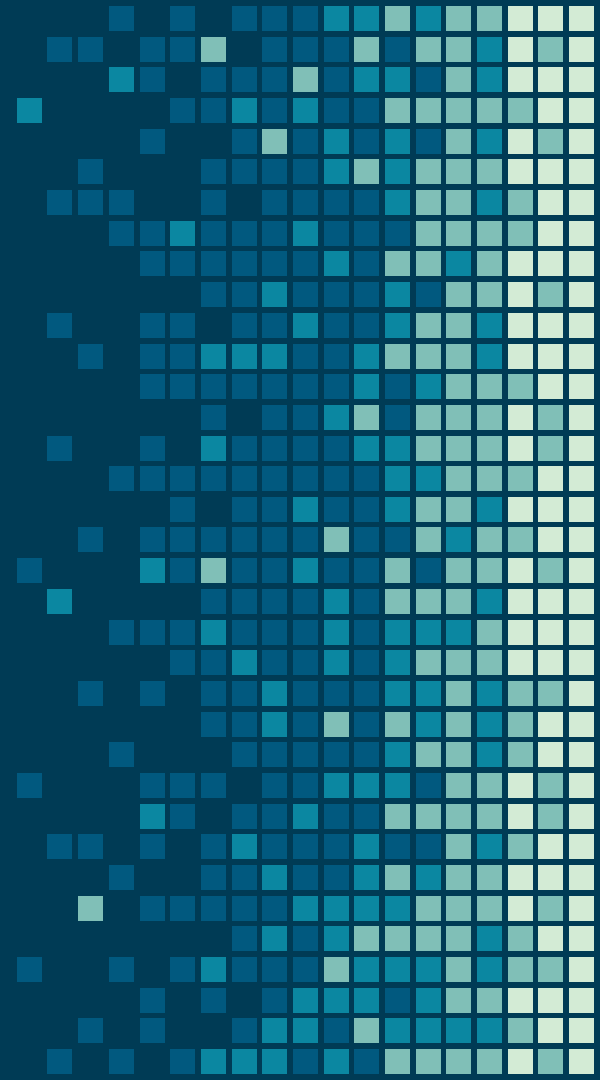



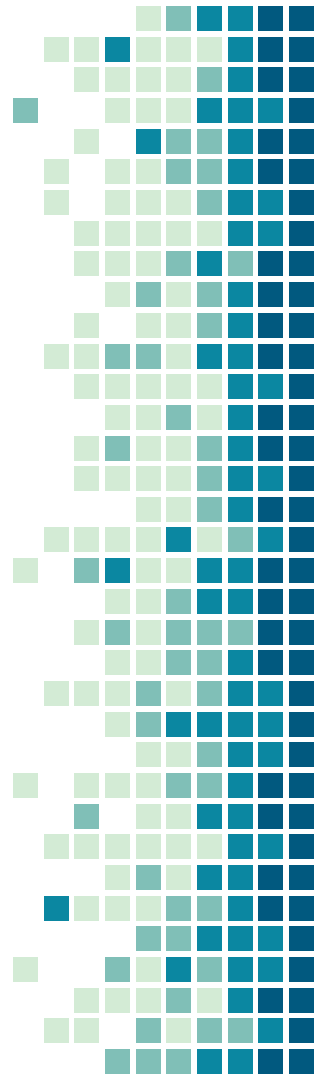
# Administración de información



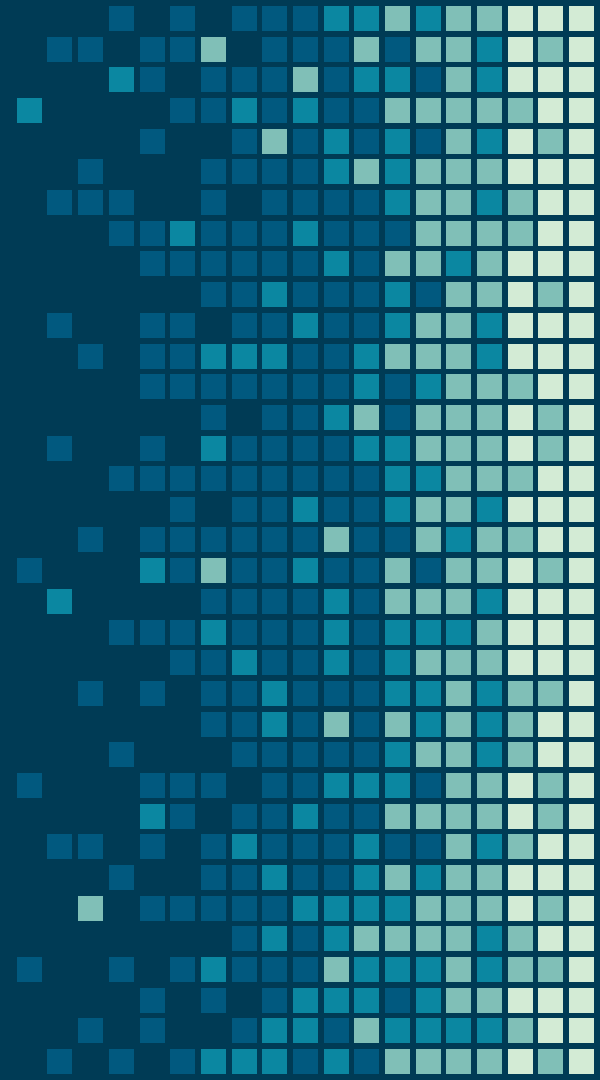
# Agenda

---

- 
- A vertical line with a blue circle at the top and a blue arrowhead at the bottom, pointing downwards.
- **Introducción**
  - **Operaciones con archivos y directorios**
  - **Estructuración**
  - **Almacenamiento**
  - **Estructuras de control del SO**



# Introducción



# Introducción

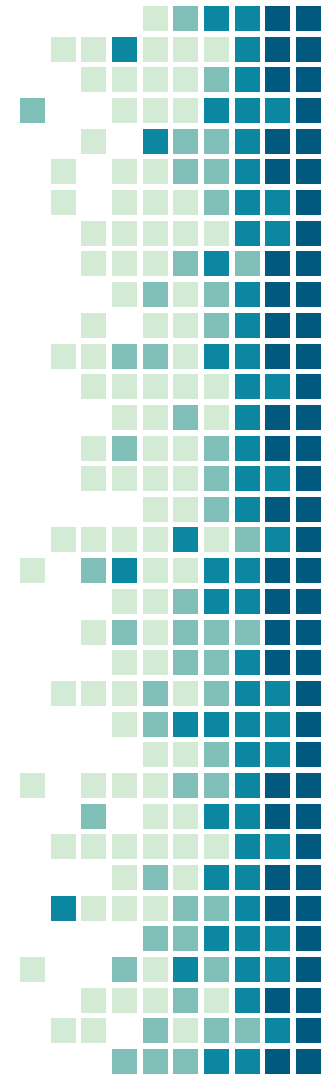
---

- Una función esencial de los sistemas es almacenar y recuperar información.
- Generalmente los sistemas deben lidiar con tres aspectos principales:

1
<ul style="list-style-type: none"><li>• Debe ser posible almacenar gran cantidad de información</li></ul>

2
<ul style="list-style-type: none"><li>• Los datos deben sobrevivir a la terminación de un proceso</li></ul>

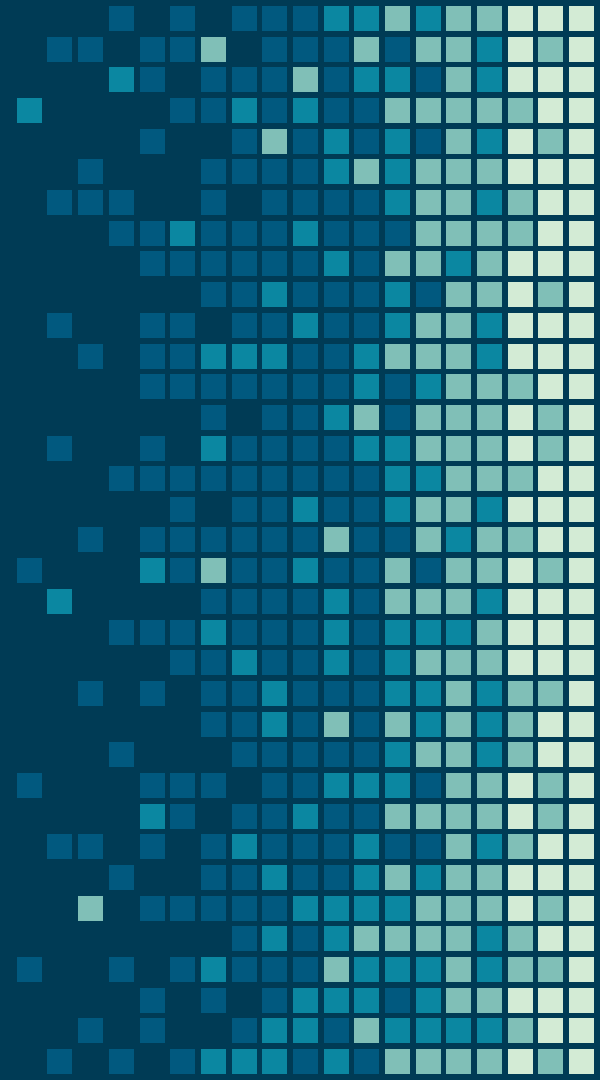
3
<ul style="list-style-type: none"><li>• Múltiples procesos pueden acceder a la información</li></ul>





- La idea fundamental es que el SO provea una abstracción para representar el conjunto de bytes (información).
- Así como el SO operativo brinda una la abstracción de proceso y memoria, también lo hace con la información, utilizando el concepto de archivos.

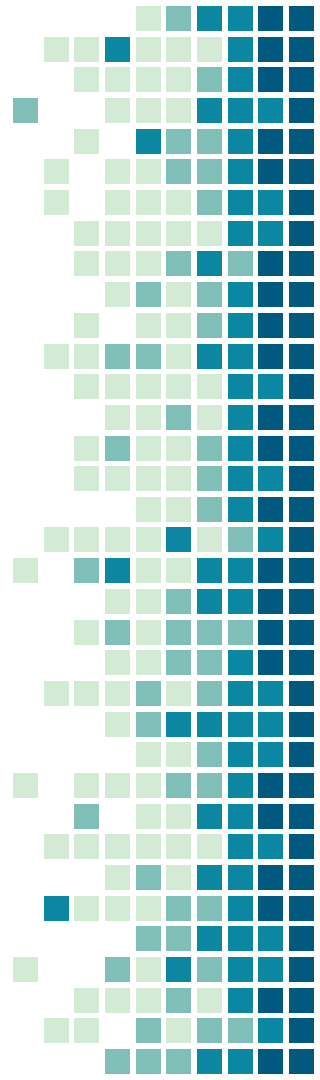
# Archivos



# Archivo

---

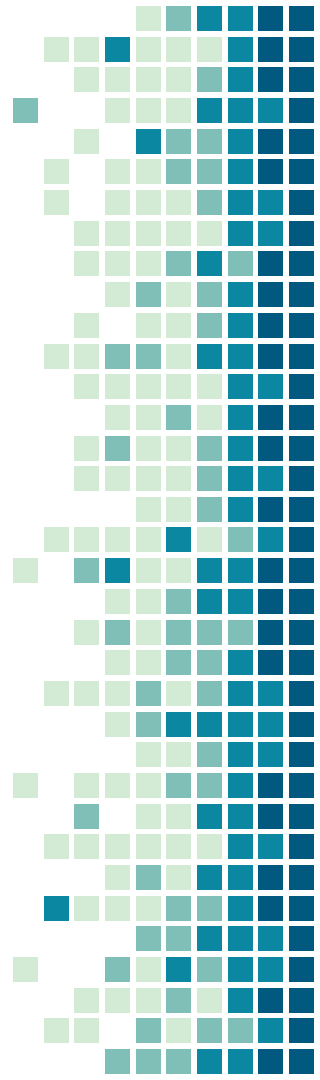
Son unidades lógicas de información creada por los procesos.  
El sistema operativo concibe un archivo como espacio de direcciones para crear un modelo similar al de la RAM.  
La información que se almacena en los archivos debe ser persistente.



# Nomenclatura de archivos

---

- La manera que se implementan los archivos en SO debe de ocultar los detalles de dónde y como está la información.
- Cuando un proceso crea un archivo le asigna un nombre y cuando termina dicho nombre se mantiene para su posterior utilización .
- Las reglas de nombrado las definen cada SO.

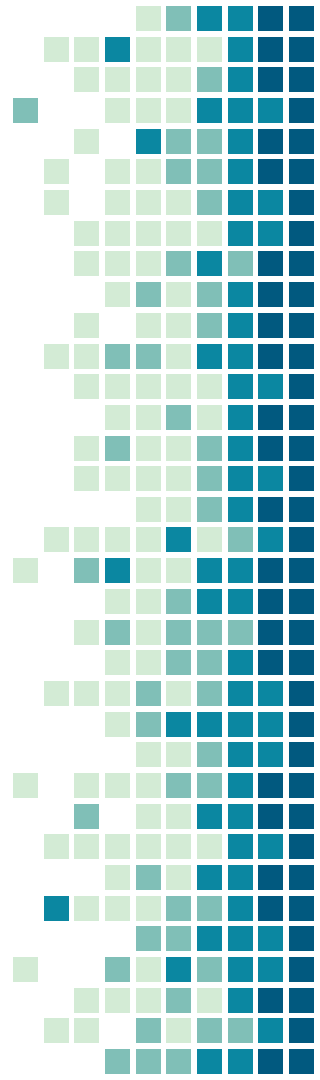




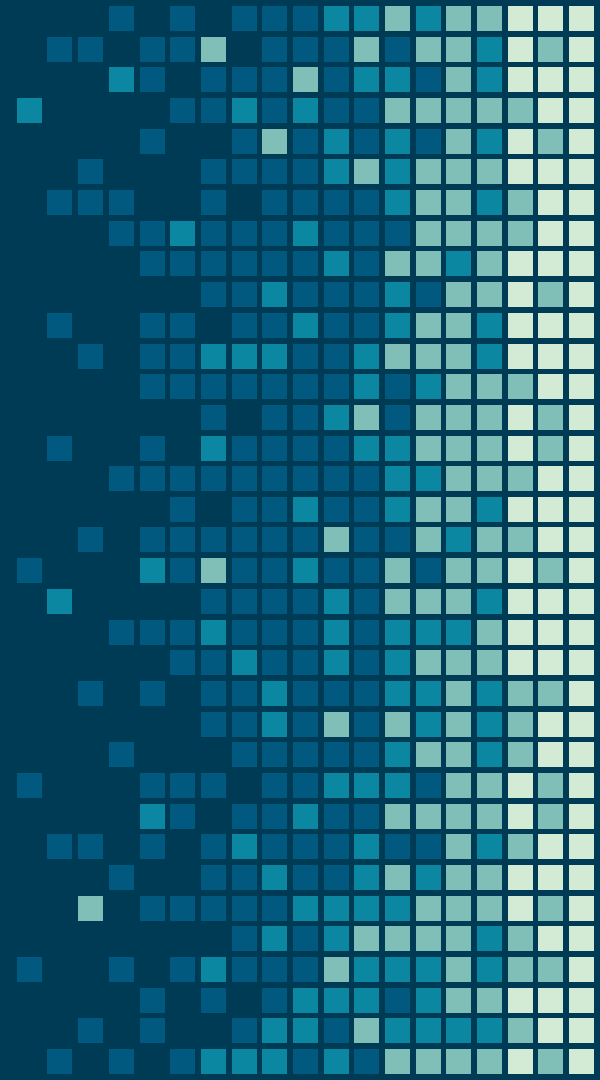
# Nomenclatura de archivos

---

Extensión	Significado
archivo.bak	Archivo de respaldo
archivo.c	Programa fuente en C
archivo.gif	Imagen en Formato de Intercambio de Gráficos de CompuServe
archivo.hlp	Archivo de ayuda
archivo.html	Documento en el Lenguaje de Marcación de Hipertexto de World Wide Web
archivo.jpg	Imagen fija codificada con el estándar JPEG
archivo.mp3	Música codificada en formato de audio MPEG capa 3
archivo.mpg	Película codificada con el estándar MPEG
archivo.o	Archivo objeto (producido por el compilador, no se ha enlazado todavía)
archivo.pdf	Archivo en Formato de Documento Portable
archivo.ps	Archivo de PostScript
archivo.tex	Entrada para el programa formateador TEX
archivo.txt	Archivo de texto general
archivo.zip	Archivo comprimido

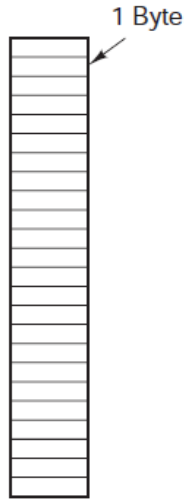


# Estructura de archivos

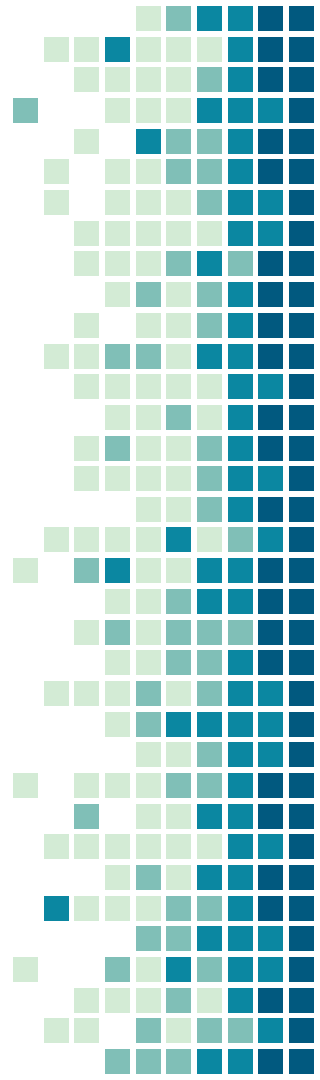


# Secuencia de bytes

---



- La manera más común es una secuencia de bytes, en donde el sistema operativo no sabe, ni le importa, qué hay en el archivo.



# Secuencia de bytes

---



- Brinda flexibilidad al permitir que los programas coloquen cualquier dato que quieran. El sistema operativo no ayuda, pero tampoco estorba. UNIX y Windows utilizan este sistema en combinación con otros.



# Secuencia de registros

---

Fue la primera manera estructurada de organizar la información en un archivo.



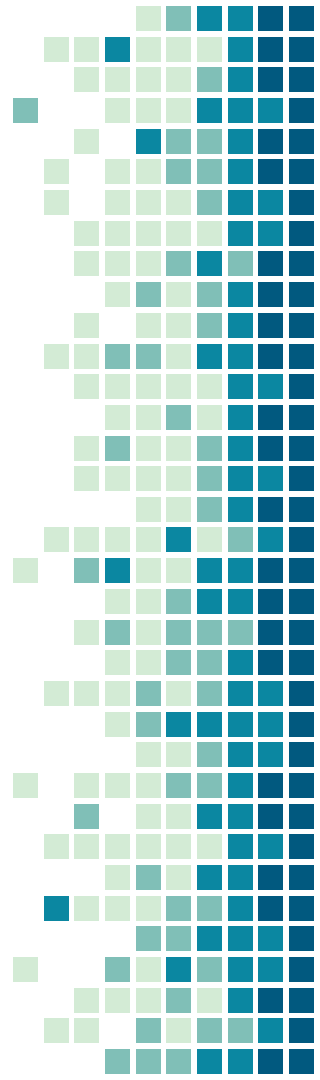
➡ La idea fundamental es que la operación read retorna un registro y la de write agrega uno o lo sobrescribe.

Fue muy utilizado en los mainframe porque las tarjetas siempre tenían el mismo formato, sin embargo en la actualidad es extraño que se utilice en algún SO de uso general

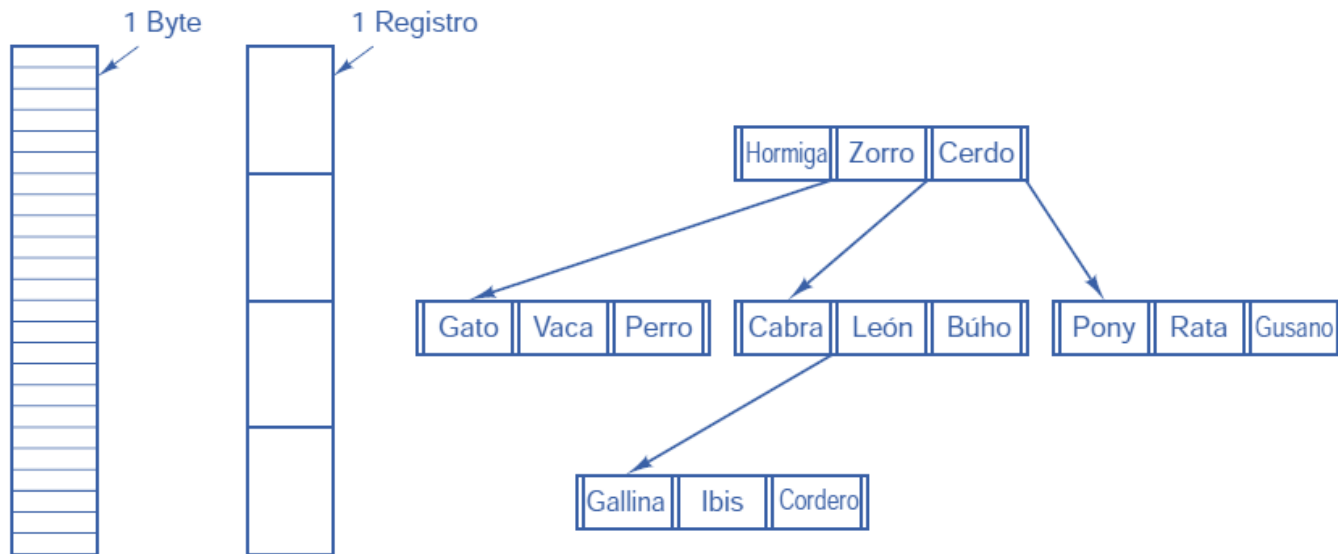
# Árbol

---

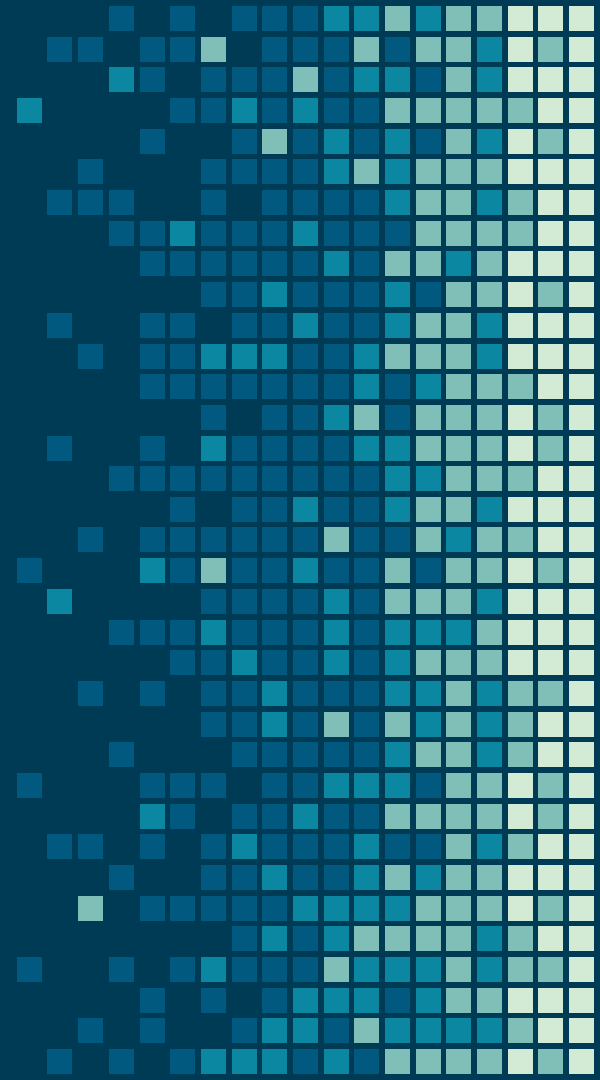
- La información se organiza en registros, donde cada uno de estos contiene una llave identificadora. El árbol se ordena según campo llave para permitir búsquedas eficientes.
- La idea fundamental es despreocuparse por la ubicación del dato y realizar las búsquedas por llave. Se utiliza de manera amplia en los sistemas mainframe actuales de compañías comerciales.



# Estructura de los archivos



# Tipos de archivos





# Tipos de archivos

---



1

Regulares: Son aquellos que contienen información de usuario.



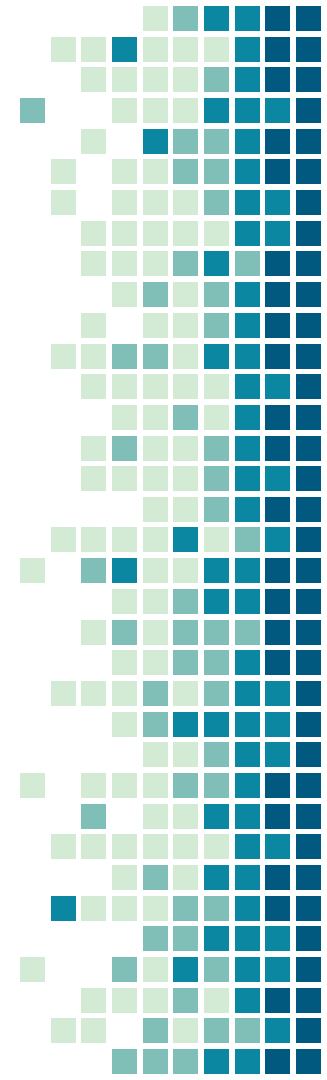
2

Especiales de caracteres: Están relacionados con E/S se utilizan para modelar dispositivos de E/S como impresoras.

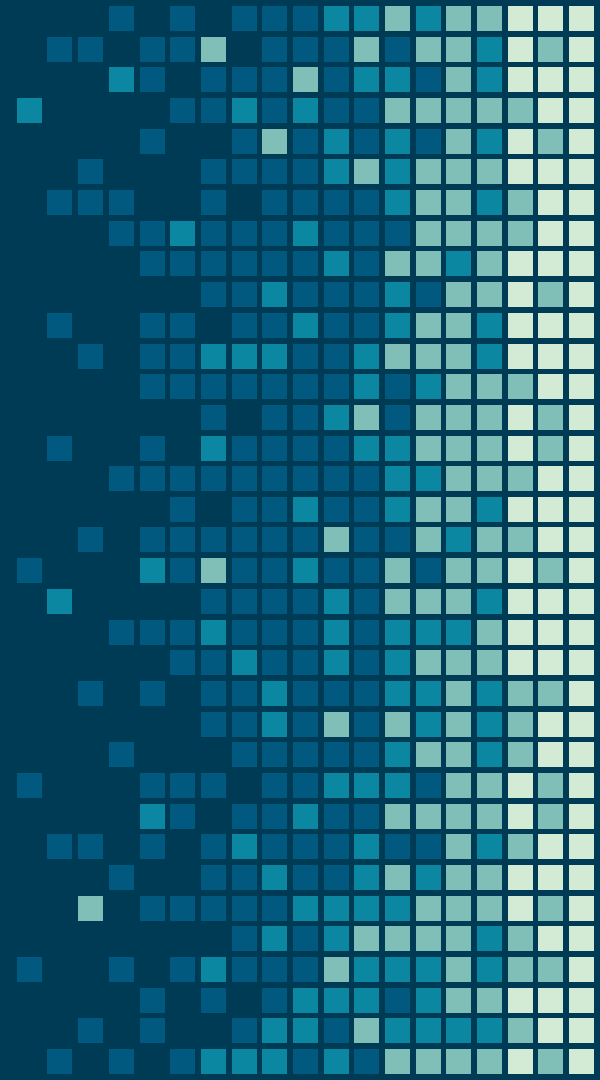


3

Especiales de bloque: Se utilizan para modelar discos.







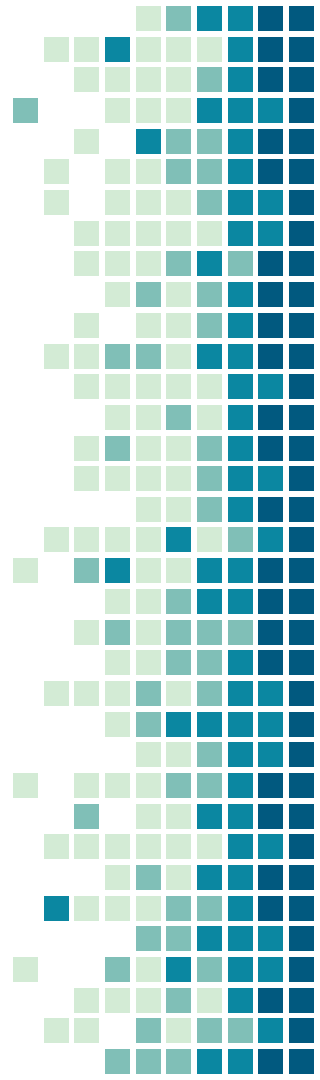
# Atributos de archivos



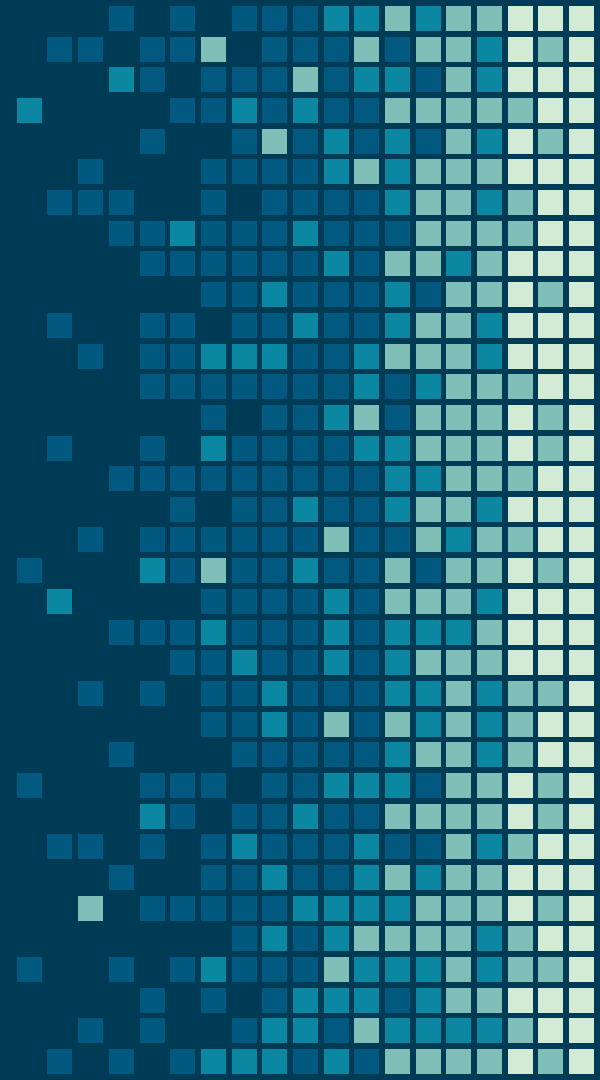
# Atributos de archivos

---

-  Todo archivo tiene un nombre y sus respectivos datos.
-  Generalmente los archivos asocian otra información como fecha y hora de última modificación.
-  A la información adicional se les conoce como atributos o metadatos.
-  Los atributos de los archivos dependen de cada sistema operativo.



# Operaciones con archivos

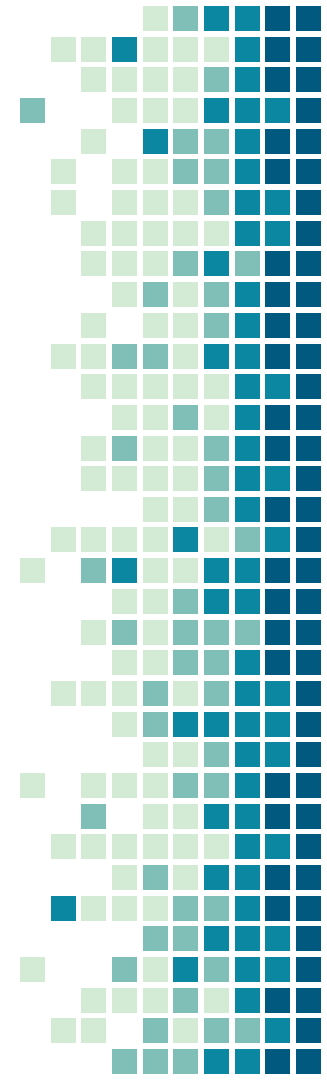


# Create

---



El archivo se crea sin datos. El propósito de la llamada es anunciar la llegada del archivo y establecer algunos de sus atributos.

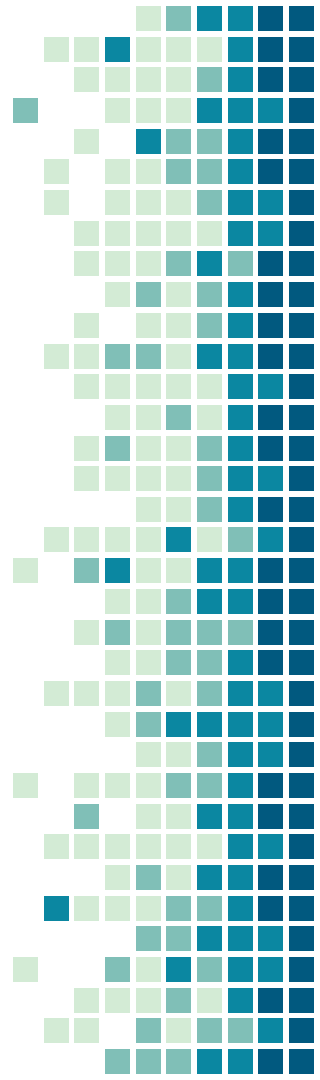


# Delete

---

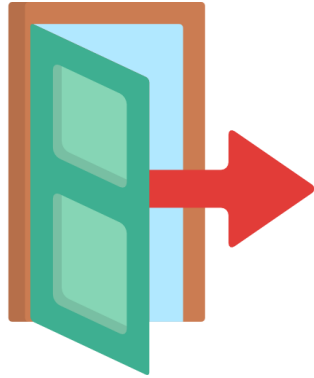


Cuando el archivo ya no se necesita, se tiene que eliminar para liberar espacio en el disco. Siempre hay una llamada al sistema para este propósito.

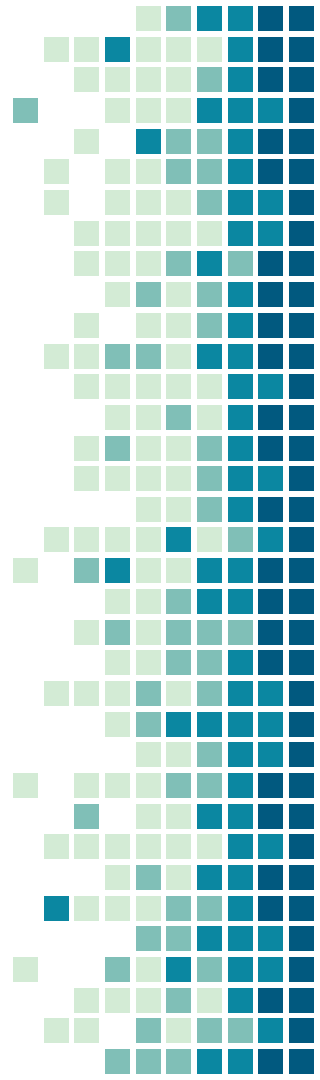


# Open

---



Antes de utilizar un archivo, un proceso debe abrirlo. El propósito de la llamada a open es permitir que el sistema lleve los atributos y la lista de direcciones de disco a memoria principal para tener un acceso rápido a estos datos.



# Close

---



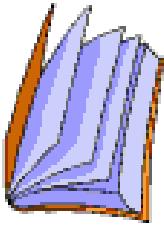
Cuando terminan todos los accesos, los atributos y las direcciones de disco ya no son necesarias, por lo que el archivo se debe cerrar para liberar espacio en la tabla interna. Muchos sistemas fomentan esto al imponer un número máximo de archivos abiertos en los procesos.



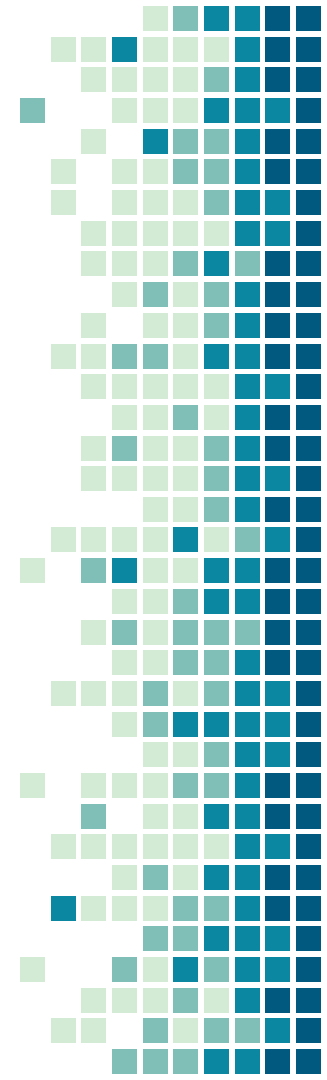


# Read

---



- Los datos se leen del archivo. Por lo general, los bytes provienen de la posición actual. La llamada al sistema necesita saber el archivo, el buffer, cantidad de datos.

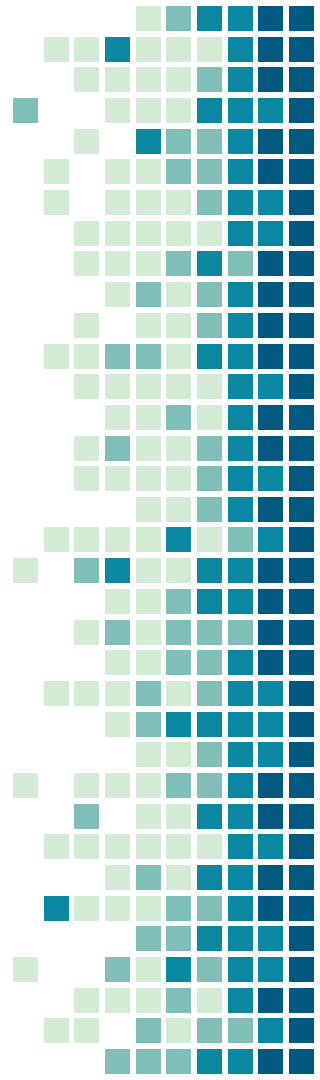


# Write

---



Los datos se escriben en el archivo, por lo general en la posición actual. Si la posición actual es al final del archivo, entonces éste aumentará e tamaño.



# Append

---



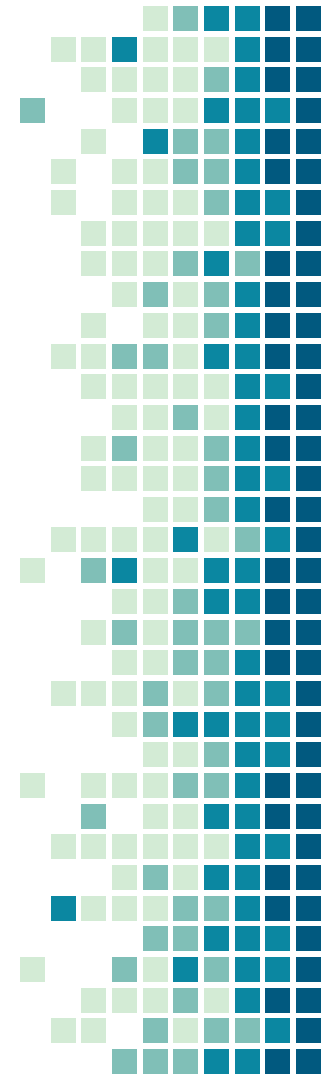
- ❏ Esta llamada es una forma restringida de write. Sólo puede agregar datos al final del archivo. Los sistemas que promueven pocas llamadas no implementan esta funcionalidad.

# Seek

---

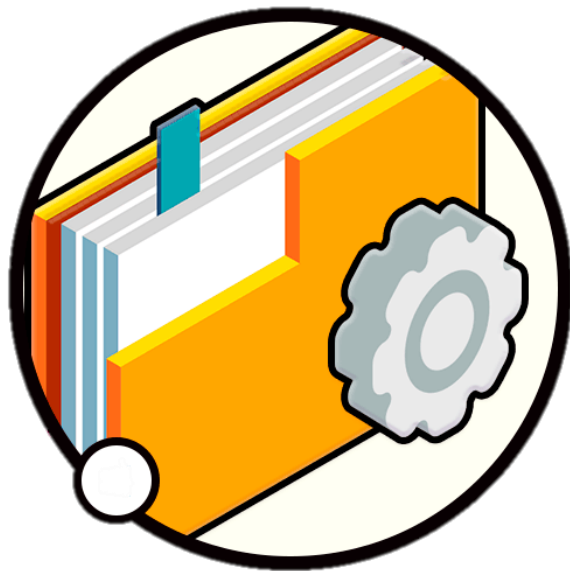


Reposiciona el apuntador del archivo en una posición específica del archivo. Después de esto se puede leer o escribir a partir de dicha posición.



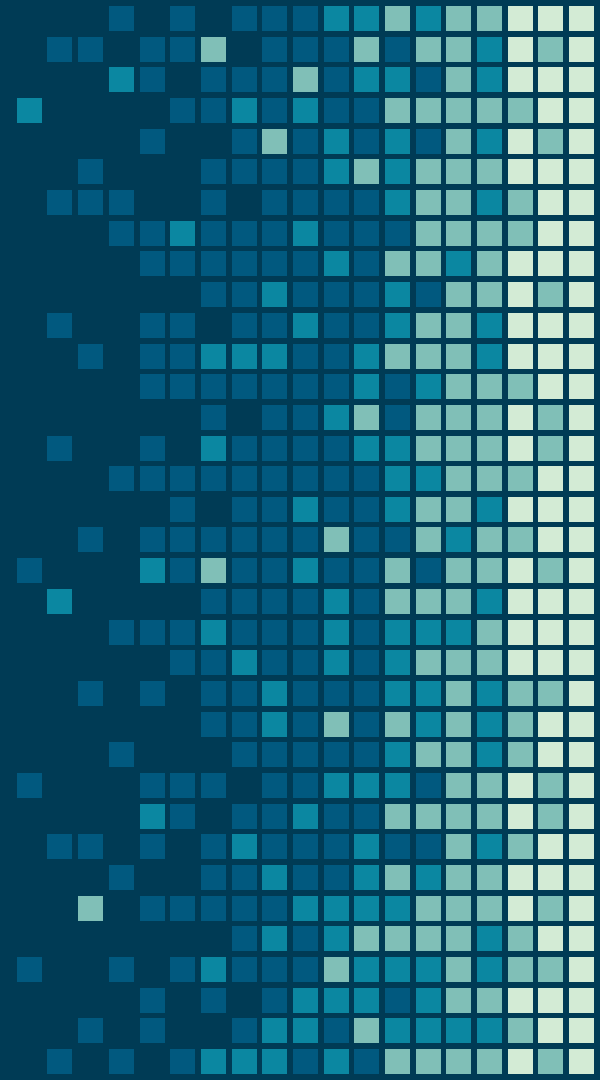
# Set y Get attributes.

---



Leen y escriben los atributos de archivos.

# Directorios

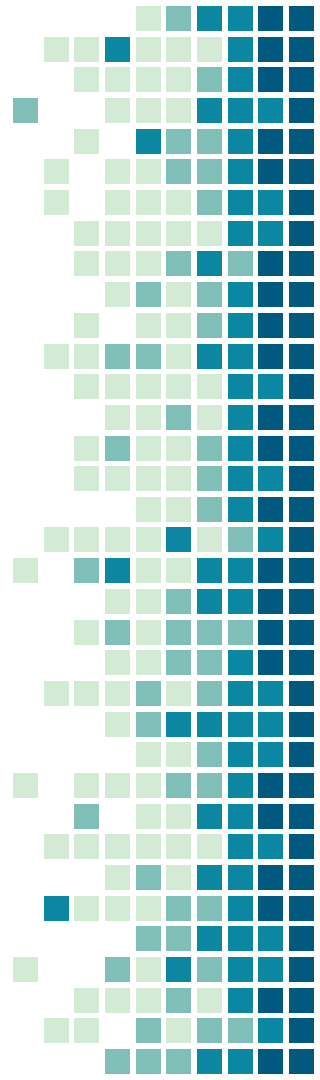


# Directorios

---



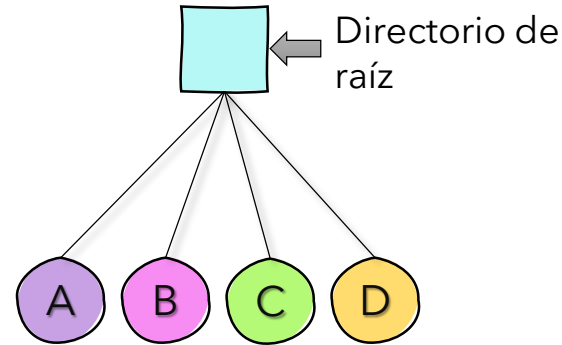
- Los sistemas de archivos actuales crean cierta estructura para la organización de los archivos, la cual la implementan por medio de directorios. Para muchos SO un directorio tiene un comportamiento similar al de un archivo.



# Sistemas de directorios de un sólo nivel

---

- Es la manera más simple de implementar un sistemas de directorios, ya que sólo se tiene un directorio que contiene todos los archivos.
- Es simple de implementar.
- La localización de los archivos se vuelve rápida, ya que hay un único lugar donde buscar.



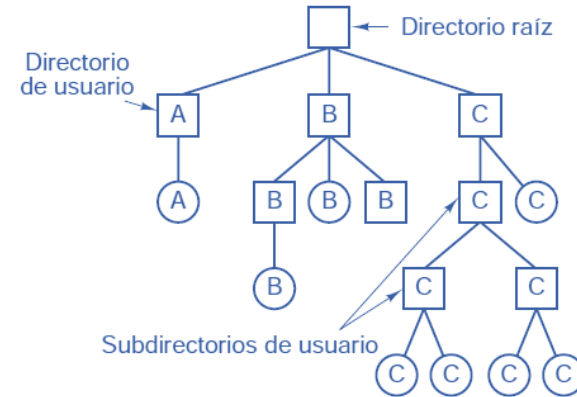


# Sistemas de directorios jerárquico

En caso de que los usuarios administren gran cantidad de archivos, un sólo directorio se vuelve insuficiente.

La idea fundamental de este tipo de sistema es crear un árbol de directorio, donde se pueda organizar mejor los archivos.

La principal ventaja que provee este sistema es que la estructuración de los archivos con N niveles de profundidad.



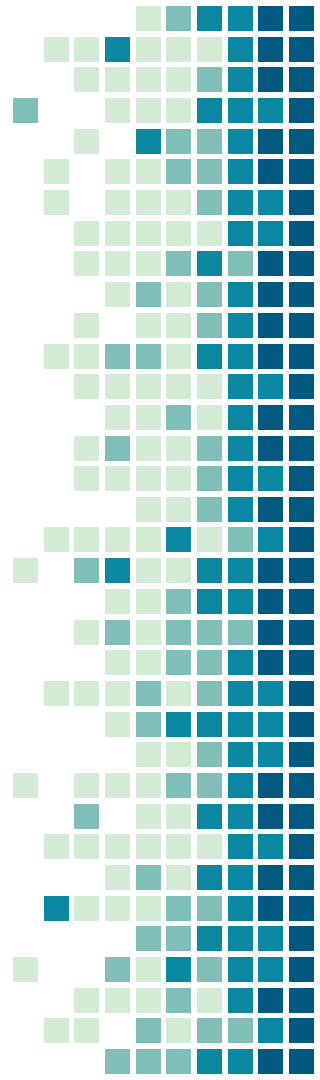
# Nombres de rutas

---

Cuando el sistema de archivos es implementado por medio de un árbol de directorios, es necesario especificar los nombres de los archivos en su ubicación.

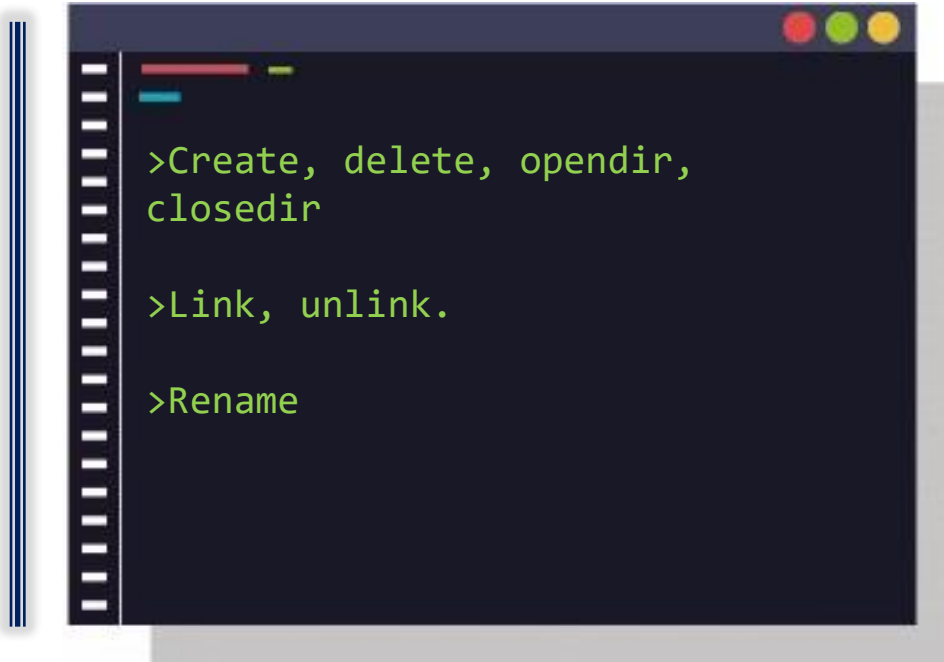
*Ruta absoluta:* consiste en el anidamiento de los directorios desde la raíz.

*Ruta relativa:* consiste en el anidamiento de directorios a partir del directorio actual.

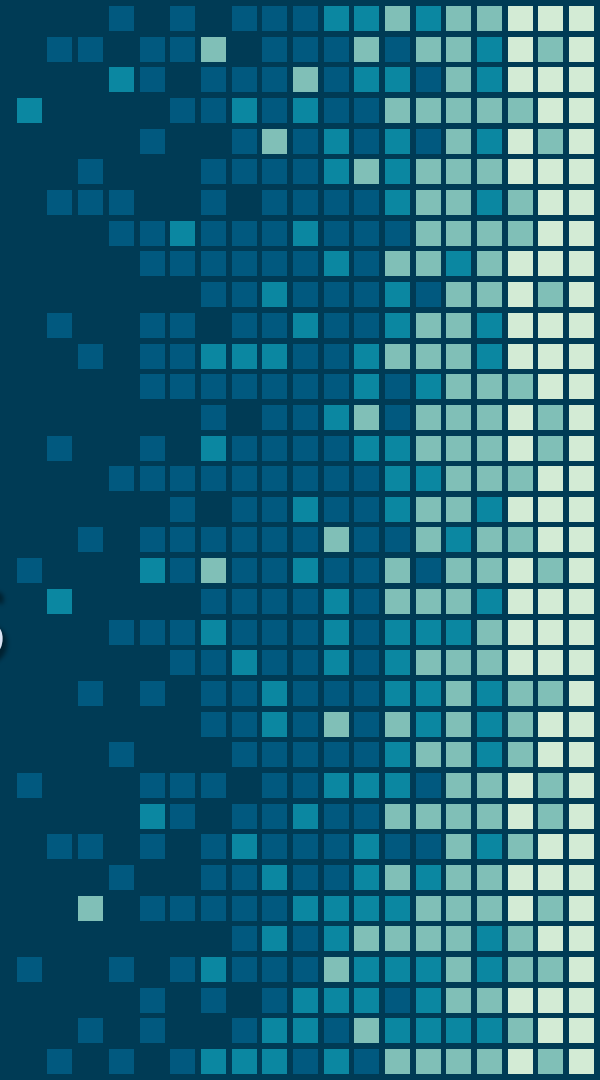


# Operaciones de directorios

---



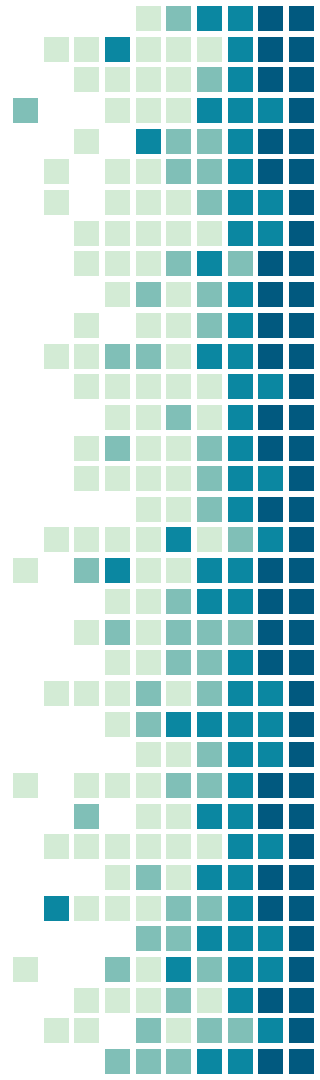
# Implementación de sistemas de archivos



# Distribución del sistema de archivos

---

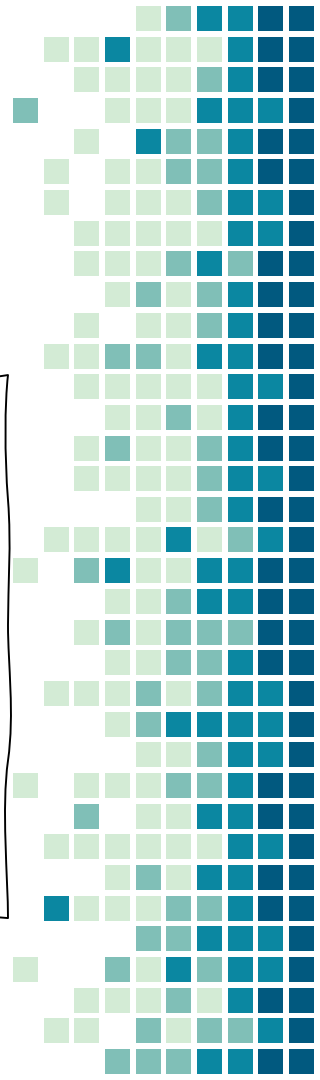
- Los sistemas de archivos se almacenan en discos.
- En el sector cero se encuentra la información de arranque
- El disco puede contener varias particiones.



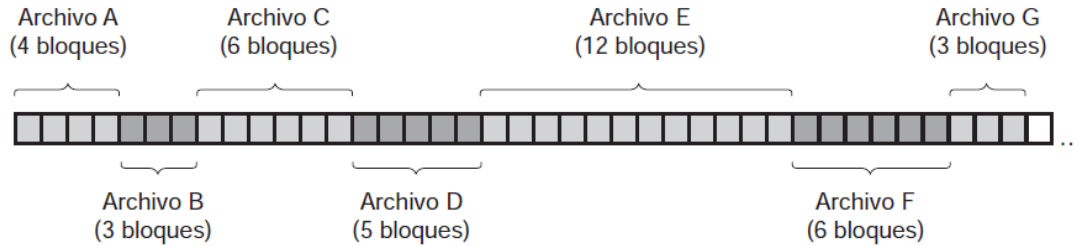
# Implementación de archivos

---

- Por la naturaleza de los archivos se tienen que almacenar en memoria secundaria.
- El reto más importante es llevar el control de los bloques de memoria que están asignados a los archivos.
- No hay sólo una manera de implementarlo, sino que cada SO lo implementa según la necesidad.

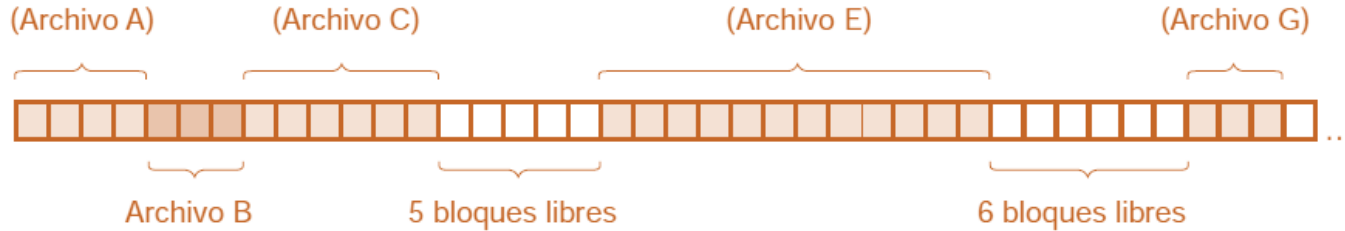


# Asignación contigua



- 💡 Es la manera más simple de almacenar la información.
- 💡 Consiste en que un archivo se almacena en un conjunto de bloques consecutivos.
- 💡 Es sencilla de implementar y tiene buen rendimiento.
- 💡 Problema de fragmentación.

# Desventaja de asignación contigua



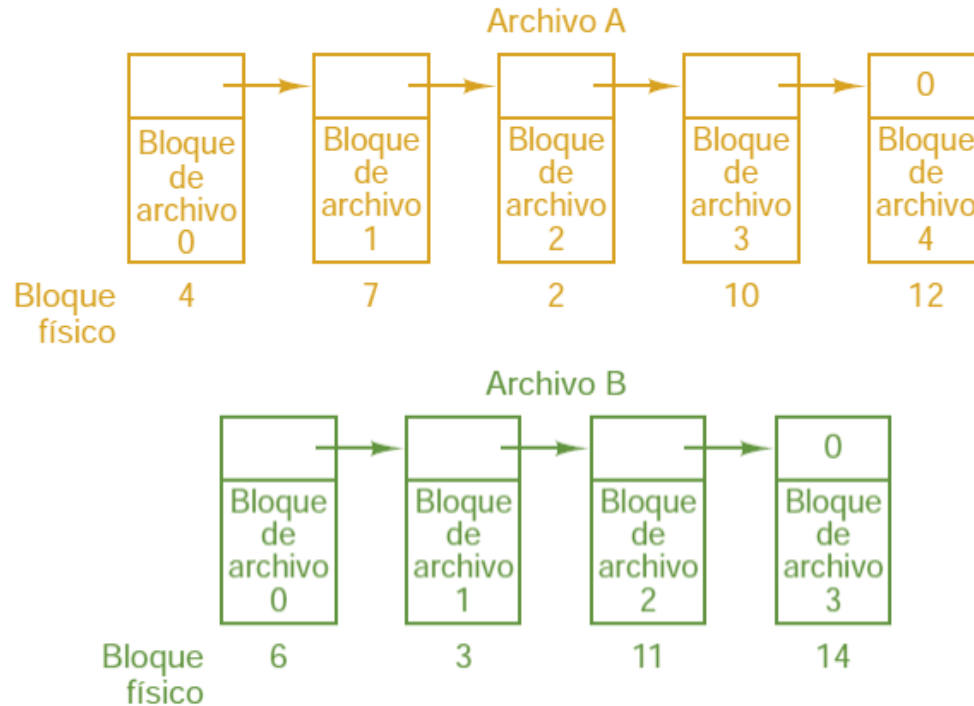


# Asignación de lista enlazada

---

- ✓ Consiste en que cada bloque se comporta como un nodo de una lista enlazada.
- ✓ La primera sección se utiliza como apuntador al siguiente bloque.
- ✓ No se pierde espacio por fragmentación externa.
- ✓ La lectura es muy lenta al consultar todos los nodos y hay desperdicio de memoria por el puntero.

# Asignación de lista enlazada



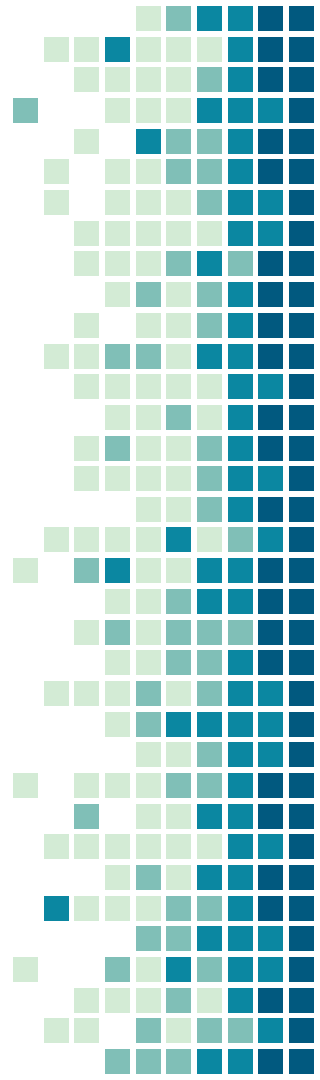
# Asignación de lista enlazada con tabla en memoria

---

Consisten en la misma idea de la lista simple, pero los apuntadores se colocan en memoria para aumentar velocidad y no desperdiciar memoria del bloque en el puntero.

Esta tabla es llamada FAT (File Allocation Table).

No es práctico para discos grandes porque ocupa mucho espacio en memoria RAM.



# Asignación de lista enlazada

Bloque físico		
0		
1		
2	10	
3	11	
4	7	← El archivo A empieza aquí
5		
6	3	← El archivo B empieza aquí
7	2	
8		
9		
10	12	
11	14	
12	-1	
13		
14	-1	
15		← Bloque sin utilizar

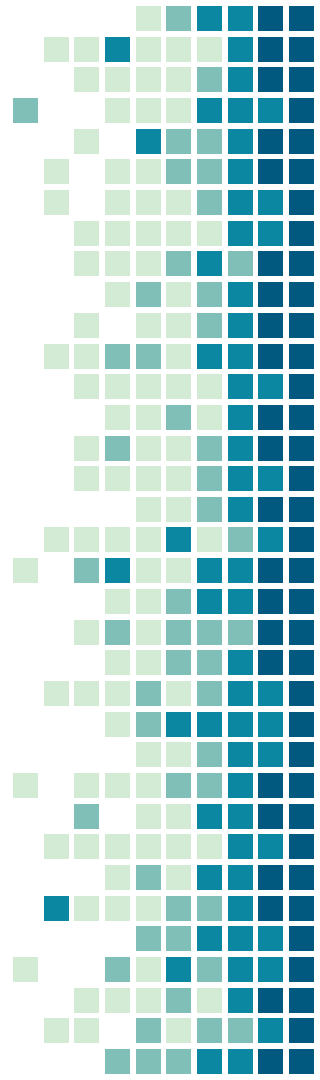
# I-Nodos

---

Es una estructura de datos, la cual lista los atributos y direcciones de discos de los bloques del archivo.

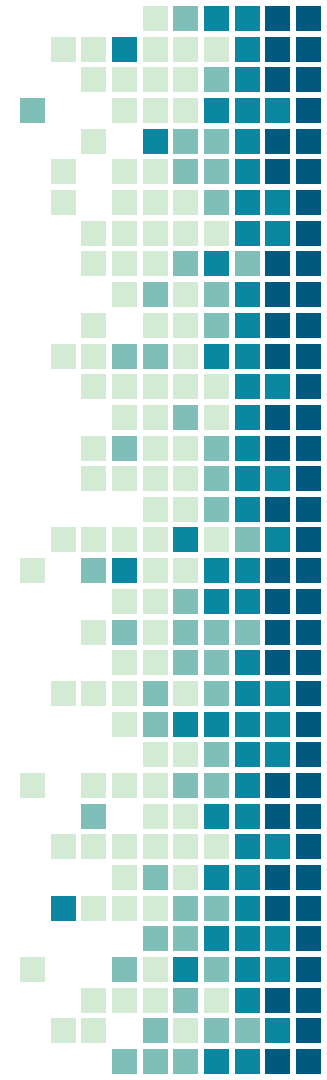
I-Nodo sólo está en memoria si el archivo está abierto.

Lo que se tiene en memoria es un arreglo con  $n$  I-Nodos y corresponden a los  $n$  archivos.



# Referencias

- ▶ Tanenbaum, A. S. (2015).  
*Sistemas operativos modernos. Pearson Educación*
- ▶ Stallings, W. (2008).  
*Sistemas operativos. Martin Iturbide.*
- ▶ CPerlman, R. (2005, December).  
*File system design with assured delete. In Third IEEE International Security in Storage Workshop (SISW'05) (pp. 6-pp). IEEE.*



# ¿Preguntas?

Realizado por: Jason Leitón Jiménez.

---

Tecnológico de Costa Rica

Ingeniería en Computadores

2024

TEC