

# SISTEMAS OPERACIONAIS

## AULA 06: SISTEMA DE ARQUIVOS

18 de junho de 2025

Prof. Me. José Paulo Lima

IFPE Garanhuns



**INSTITUTO  
FEDERAL**  
Pernambuco

# SUMÁRIO I



## Sistema de Arquivos

### Arquivos

- Nomeação de Arquivos

- Estrutura de arquivos

- Tipos de Arquivos

- Acesso aos Arquivos

- Atributos de Arquivos

- Operações com arquivos

### Diretórios

- Nomes de Caminhos

- Operações com Diretórios

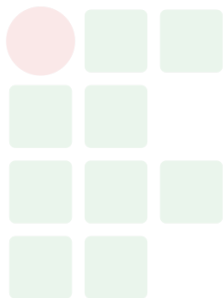
### Implementação do Sistema de Arquivos

- Alocação de Arquivo

- Implementação de Diretório

## Referências

**SISTEMA DE ARQUIVOS**



**INSTITUTO  
FEDERAL**  
Pernambuco

# SISTEMA DE ARQUIVOS



- ▶ Todas as operações precisam armazenar e recuperar informações;
- ▶ Desafios:
  1. Os processos podem armazenar uma quantidade limitada de informação, restrita ao tamanho do espaço de endereçamento virtual;
  2. Manter a informação dentro do espaço de endereçamento do processo onde o processo pode finalizar sua execução e a informação ser perdida;
  3. Múltiplos processos podem ter acesso à mesma informação ao mesmo tempo.
- ▶ A solução é tornar a própria informação independente de qualquer processo.

# SISTEMA DE ARQUIVOS



- ▶ Temos três requisitos essenciais para o armazenamento de informação por longo prazo:
  1. Deve ser possível armazenar uma quantidade muito grande de informação;
  2. A informação deve sobreviver ao término do processo que a usa;
  3. Múltiplos processos devem ser capazes de acessar a informação concorrentemente.

# SISTEMA DE ARQUIVOS

## ARQUIVOS



- ▶ Arquivos são unidades lógicas de informações criadas por processos;
- ▶ Os arquivos também são uma espécie de espaço de endereçamento, mas eles são usados para modelar o disco e não a memória;
- ▶ A informação armazenada em arquivos deve ser persistentes, isto é, não pode ser afetada pela criação e término de um processo;
- ▶ Arquivos são gerenciados pelo sistema operacional.
  - ▶ O sistema operacional gerencia o modo como os arquivos são estruturados, nomeados, acessados, usados, protegidos e implementados.

# SISTEMA DE ARQUIVOS

## NOMEAÇÃO DE ARQUIVOS



- ▶ Arquivo é um mecanismo de abstração que oferece meios de armazenar e ler informações no disco;
- ▶ A nomeação deve ser feita de um modo que isole do usuário os detalhes sobre como e onde a informação está armazenada e como os discos na verdade funcionam;
- ▶ Quando um processo cria um arquivo, ele dá um nome a esse arquivo e ao terminar sua execução, o arquivo continua existindo e outros processos podem ter acesso a ele através do nome.
  - ▶ Exemplo: “Notas.xls”.

# SISTEMA DE ARQUIVOS

## NOMEAÇÃO DE ARQUIVOS



- ▶ As regras para se dar nome a um arquivo variam de sistemas para sistemas;
  - ▶ Em geral todos os sistemas operacionais permitem uma cadeia de caracteres de até 255 caracteres;
  - ▶ Existe algumas limitações de caracteres especiais dependendo do sistemas operacional.
    - ▶ Distinção de letras maiúsculas de minúsculas:
      - UNIX: *Case sensitive*;
      - MS-DOS: Não é *case sensitive*.



# SISTEMA DE ARQUIVOS

## NOMEAÇÃO DE ARQUIVOS



- ▶ Muitos sistemas operacionais permitiram nomes de arquivos com duas partes separadas por um ponto:
  - ▶ A parte que segue o ponto é chamada de extensão do arquivo e indica alguma informação sobre o arquivo;
    - ▶ Exemplo: “apresentacao.ppt”.
  - ▶ No MS-DOS os nomes dos arquivos tem 8 caracteres mais uma extensão opcional de 1-3 caracteres;
  - ▶ No UNIX, a extensão, se houver, fica a critério do usuário, e um arquivo pode conter até mesmo duas ou mais extensões.
    - ▶ Exemplo: “documentos.doc.zip”

# SISTEMA DE ARQUIVOS

## ESTRUTURA DE ARQUIVOS



- ▶ Os arquivos podem ser estruturados de várias formas:
  1. Uma sequência desestruturada de bytes:
    - ▶ O sistema operacional não sabe o que o arquivo contém, só o que ele vê são bytes;
    - ▶ O significado dos bytes são dados pelos programas usuários.
  2. Uma sequência de registros de tamanho fixo:
    - ▶ Cada um com alguma estrutura interna;
    - ▶ A ideia central de ter um arquivo como uma sequência de registros;
    - ▶ A operação de leitura retorna um registro e a escrita sobrepõe ou anexa um registro.
  3. Constituído de uma árvore de registros:
    - ▶ Não necessariamente todos de mesmo tamanho, cada um contendo um campo chave em uma posição fixa no registro;
    - ▶ A árvore é ordenada pelo campo-chave para que a busca seja mais rápida.

# SISTEMA DE ARQUIVOS

## ESTRUTURA DE ARQUIVOS

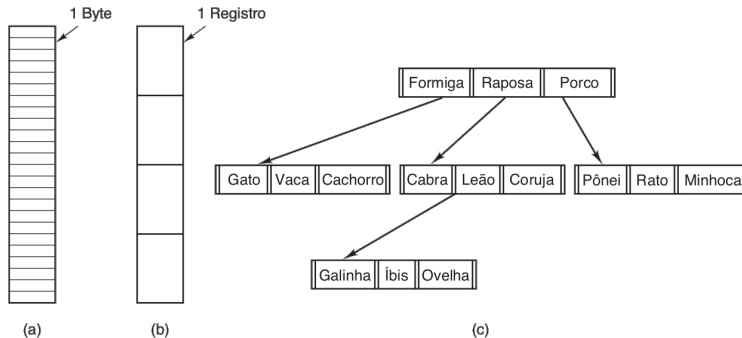


Figura extraída de Tanenbaum e Bos (2016, p. 184).

- (a) Sequência de bytes;
- (b) Sequência de registros;
- (c) Árvore.

# SISTEMA DE ARQUIVOS

## TIPOS DE ARQUIVOS



### 1. Arquivos regulares:

- ▶ São aqueles que contêm informação do usuário, podendo, geralmente, ser de dois tipos:
- ▶ ASCII:
  - ▶ São constituídos de linhas de texto;
  - ▶ Cada linha termina com um caractere especial;
  - ▶ São apresentados a impressora;
  - ▶ Podem ser editados por qualquer editor de texto.
- ▶ Binários:
  - ▶ São os arquivos não ASCII;
  - ▶ Possuem uma estrutura interna conhecida pelos programas que os usam.

# SISTEMA DE ARQUIVOS

## TIPOS DE ARQUIVOS



### 2. Diretórios:

- ▶ São arquivos do sistema que mantêm a estrutura do sistema de arquivos.

### 3. Arquivos especiais de caracteres:

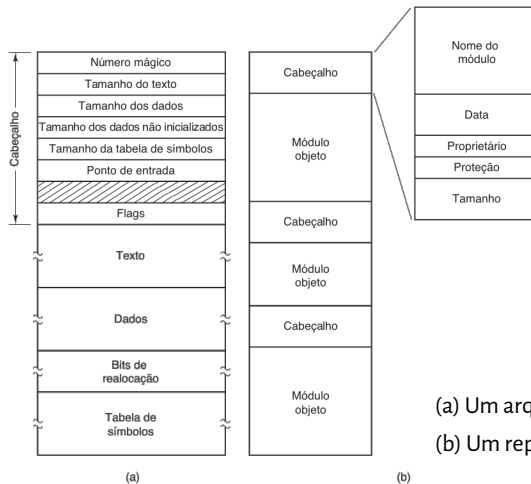
- ▶ São relacionados a E/S e usados para modelar dispositivos de E/S.
  - ▶ Exemplos: Impressoras e redes.

### 4. Arquivos especiais de bloco:

- ▶ São usados para modelar discos.

# SISTEMA DE ARQUIVOS

## TIPOS DE ARQUIVOS



(a) Um arquivo executável;  
(b) Um repositório (*archive*).

# SISTEMA DE ARQUIVOS

## ACESSO AOS ARQUIVOS



- ▶ Acesso sequencial:
  - ▶ Presente nos primeiros sistemas operacionais, que forneciam apenas este tipo de acesso aos arquivos;
  - ▶ Apenas podia ler todos os bytes ou registros de um arquivo, partindo do início em sequência.
- ▶ Acesso aleatório:
  - ▶ Arquivos cujos bytes ou registros possam ser lidos em qualquer ordem;
  - ▶ Com o armazenamento nos discos possibilitou a leitura dos bytes ou registros de um arquivo fora de ordem em que apareciam no disco;
  - ▶ Arquivos de acesso aleatório são essenciais em aplicações como sistemas de banco de dados;
  - ▶ Métodos podem ser usados para especificar a partir de onde a leitura começa:
    - ▶ *Read*: Indica a posição do arquivo em que se inicializa a leitura;
    - ▶ *Seek*: Indica a posição atual. Depois de um *seek*, o arquivo pode ser lido sequencialmente a partir de sua posição atual.

# SISTEMA DE ARQUIVOS

## ATRIBUTOS DE ARQUIVOS

- ▶ Todo arquivo possui um nome e seus dados;
- ▶ Além disso, os sistemas operacionais associam outras informações em cada arquivo chamadas atributos (metadados).

Atributo	Significado
Proteção	Quem tem acesso ao arquivo e de que modo
Senha	Necessidade de senha para acesso ao arquivo
Criador	ID do criador do arquivo
Proprietário	Proprietário atual
Flag de somente leitura	0 para leitura/escrita; 1 para somente leitura
Flag de oculto	0 para normal; 1 para não exibir o arquivo
Flag de sistema	0 para arquivos normais; 1 para arquivos de sistema
Flag de arquivamento	0 para arquivos com backup; 1 para arquivos sem backup
Flag de ASCII/binário	0 para arquivos ASCII; 1 para arquivos binários
Flag de acesso aleatório	0 para acesso somente sequencial; 1 para acesso aleatório
Flag de temporário	0 para normal; 1 para apagar o arquivo ao sair do processo
Flag de travamento	0 para destravados; diferente de 0 para travados
Tamanho do registro	Número de bytes em um registro
Posição da chave	Posição da chave em cada registro
Tamanho da chave	Número de bytes na chave
Momento de criação	Data e hora de criação do arquivo
Momento do último acesso	Data e hora do último acesso do arquivo
Momento da última alteração	Data e hora da última modificação do arquivo
Tamanho atual	Número de bytes no arquivo
Tamanho máximo	Número máximo de bytes no arquivo

Tabela extraída de Tanenbaum e Bos (2016, p. 187).



# SISTEMA DE ARQUIVOS

## OPERAÇÕES COM ARQUIVOS



- ▶ Create: O arquivo é criado sem dados;
- ▶ Delete: Exclui o arquivo que não é mais necessário;
- ▶ Open: Antes de usar um arquivo, um processo deve abri-lo;
- ▶ Close: Quando o acesso termina, o arquivo deve ser fechado;
- ▶ Read: Dados são lidos do arquivo;
- ▶ Write: Dados são escritos no arquivo;
- ▶ Append: Dados são escritos apenas no final dos arquivos;
- ▶ Seek: Reposiciona o ponteiro de arquivo para outro local;
- ▶ Get Attributes: Lê os atributos de um arquivo;
- ▶ Set Attributes: Alteração de atributos pelo usuário;
- ▶ Rename: Renomear o arquivo.

# SISTEMA DE ARQUIVOS

## DIRETÓRIOS



- ▶ Para controlar os arquivos os sistemas utilizam, em geral, diretórios ou pastas;
- ▶ A organização pode ser feita das seguintes maneiras:
  1. Nível único;
  2. Hierárquicos.

# SISTEMA DE ARQUIVOS

## DIRETÓRIOS

### 1. Sistemas de diretório em nível único:

- ▶ Um diretório único contendo todos os arquivos;
- ▶ Chamado de diretório-raiz;
- ▶ Nos primeiros computadores pessoais, esse sistema era comum;
- ▶ Vantagens:
  - ▶ Simplicidade;
  - ▶ Localização rápida de arquivos.
- ▶ Utilizados em sistemas embarcados, câmeras digitais, telefones, etc.

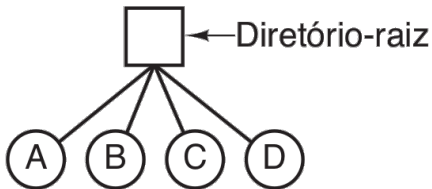


Figura extraída de Tanenbaum e Bos (2016, p. 191).

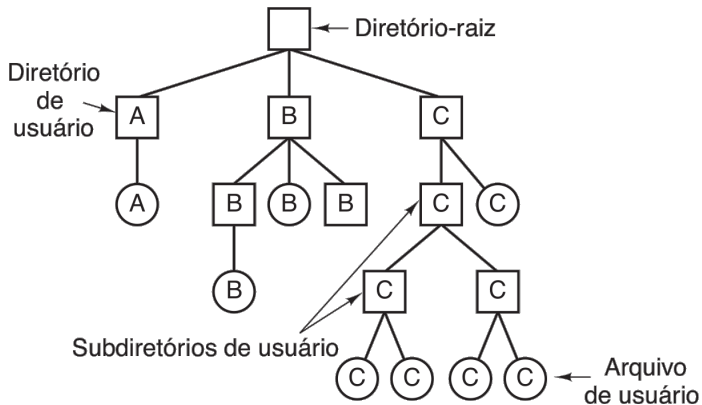
# SISTEMA DE ARQUIVOS

## DIRETÓRIOS



### 2. Sistemas de diretório hierárquicos:

- ▶ A ideia é encontrar uma maneira de agrupar os arquivos relacionados em um mesmo local;
- ▶ Os usuários podem ter tantos diretórios quanto necessários para agrupar os arquivos.



# SISTEMA DE ARQUIVOS

## NOMES DE CAMINHOS



- ▶ Quando o sistema de arquivos é organizado como uma árvore de diretórios, é preciso algum modo de especificar o nome do arquivo;

### 1. Caminho absoluto:

- ▶ Formado pelo caminho entre o diretório-raiz e o arquivo;
  - Windows: \usr\ast\caixapostal;
  - UNIX: /usr/ast/caixapostal.

### 2. Caminho relativo:

- ▶ Ele é usado com o conceito de diretório de trabalho;
- ▶ Todos os nomes que não comecem no diretório-raiz são assumidos como relativos ao diretório de trabalho;
- ▶ Por exemplo, se o diretório de trabalho atual é /usr/ast, então o arquivo cujo caminho absoluto é /usr/ast/caixapostal pode ser referenciado somente como caixapostal.

```
cp /usr/ast/caixapostal /usr/ast/caixapostal.bak  
cp caixapostal caixapostal.bak
```

# SISTEMA DE ARQUIVOS

## OPERAÇÕES COM DIRETÓRIOS



- ▶ Create: Cria um diretório;
- ▶ Delete: Remove um diretório;
- ▶ Opendir: Permite a leitura de diretórios;
- ▶ Closedir: Fecha os diretórios abertos;
- ▶ Readdir: Devolve a próxima entrada em um diretório aberto;
- ▶ Rename: Renomear o nome dos diretórios;
- ▶ Link: É uma técnica que possibilita a um arquivo aparecer em mais de um diretório;
- ▶ Unlik: Se estiver em apenas um diretório, o arquivo será removido, caso esteja em vários diretórios, somente o nome do seu caminho especificado será apagado.

# SISTEMA DE ARQUIVOS

## IMPLEMENTAÇÃO DO SISTEMA DE ARQUIVOS

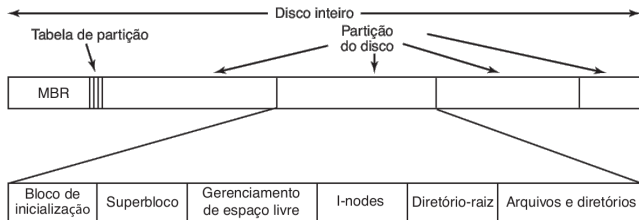


- ▶ Na implementação, o sistema de arquivo está interessado em:
  - ▶ Como os arquivos e diretórios são armazenados;
  - ▶ Como o espaço em disco é gerenciado;
  - ▶ Como fazer todo o sistema funcionar de maneira eficiente e confiável;

# SISTEMA DE ARQUIVOS

## IMPLEMENTAÇÃO DO SISTEMA DE ARQUIVOS

- ▶ Os sistemas de arquivos são armazenados em discos;
- ▶ A maioria dos discos é dividida em uma ou mais partições, com sistemas de arquivos independentes para cada partição;
- ▶ O setor 0 do disco é chamado MBR (*Master Boot Record*, Registro Mestre de Inicialização);
  - ▶ O fim do MRB contém a tabela de partição que indica os endereços iniciais e finais de cada partição;
  - ▶ Quando o computador é inicializado, a BIOS lê e executa o MBR.





# SISTEMA DE ARQUIVOS

## ALOCÇÃO DE ARQUIVO



- ▶ São usados vários métodos em diferentes sistemas operacionais:
  1. Alocação Contígua;
  2. Alocação por lista encadeada;
  3. Alocação por lista encadeada usando uma tabela na memória;
  4. I-nodes.

# SISTEMA DE ARQUIVOS

## ALOCAÇÃO DE ARQUIVO: ALOCAÇÃO CONTÍGUA



### 1. Alocação Contígua:

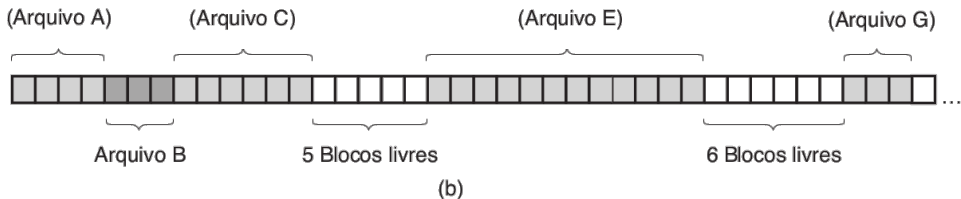
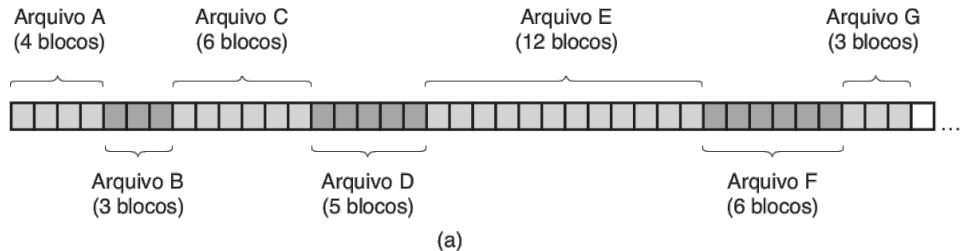
- ▶ Esquema mais simples de alocação onde os blocos são armazenados em blocos contíguos de disco;
- ▶ Exemplo:
  - ▶ Em um disco com blocos de 1KB, um arquivo com 50KB seria necessário 50 blocos.
- ▶ Vantagem:
  - ▶ É simples de implementar e o arquivo pode ser lido com um acesso sequencial ao disco.
- ▶ Desvantagem:
  - ▶ Com o tempo o disco fica fragmentado.

# SISTEMA DE ARQUIVOS

## ALOCAÇÃO DE ARQUIVO: ALOCAÇÃO CONTÍGUA



25



# SISTEMA DE ARQUIVOS

## ALOCAÇÃO DE ARQUIVO: ALOCAÇÃO POR LISTA ENCADEADA

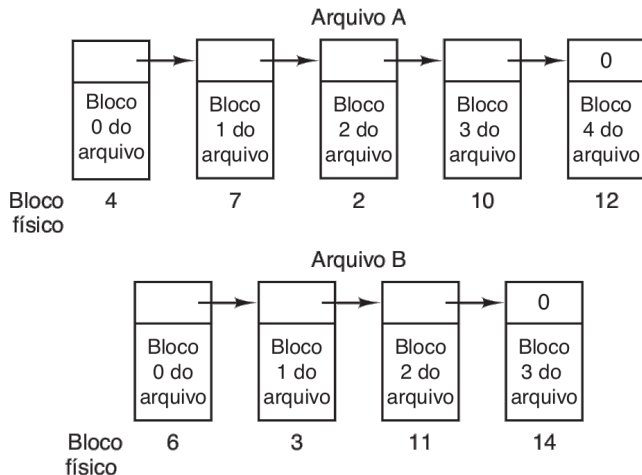


### 2. Alocação por lista encadeada:

- ▶ A primeira palavra de cada bloco é um ponteiro para o bloco seguinte;
- ▶ O restante do bloco fica responsável para armazenar os dados;
- ▶ Apenas o endereço do primeiro bloco do arquivo é armazenado no disco;
- ▶ Vantagem:
  - ▶ Todo bloco de disco é usado, nenhum espaço é perdido pela fragmentação;
  - ▶ É suficiente armazenar apenas o endereço em disco do primeiro bloco.
- ▶ Desvantagem:
  - ▶ O acesso aleatório é extremamente lento;
  - ▶ A quantidade de dados que um bloco pode armazenar não é mais um potência de 2 porque os ponteiros ocupam alguns bytes.

# SISTEMA DE ARQUIVOS

## ALOCAÇÃO DE ARQUIVO: ALOCAÇÃO POR LISTA ENCADEADA



# SISTEMA DE ARQUIVOS

## ALOCAÇÃO DE ARQUIVO: ALOCAÇÃO POR LISTA ENCADEADA USANDO UMA TABELA NA MEMÓRIA



### 3. Alocação por lista encadeada usando uma tabela na memória:

- ▶ As desvantagens por lista encadeada podem ser eliminadas colocando-se as palavras do ponteiro de cada bloco em uma tabela na memória;
- ▶ Essa tabela na memória principal é chamada FAT (*File Allocation Table*), ou tabela de alocação de arquivos;
- ▶ Vantagem:
  - ▶ Todo bloco fica disponível para os dados;
  - ▶ O acesso aleatório se torna muito mais fácil.
- ▶ Desvantagem:
  - ▶ Toda tabela deve estar na memória o tempo todo;
  - ▶ Exemplo: um disco de 200GB e blocos de 1KB, a tabela precisará de 200 milhões de entradas.

# SISTEMA DE ARQUIVOS

## ALOCAÇÃO DE ARQUIVO: ALOCAÇÃO POR LISTA ENCADEADA USANDO UMA TABELA NA MEMÓRIA

Bloco físico

0		
1		
2	10	
3	11	
4	7	← O arquivo A começa aqui
5		
6	3	← O arquivo B começa aqui
7	2	
8		
9		
10	12	
11	14	
12	-1	
13		
14	-1	
15		← Bloco sem uso

Figura extraída de Tanenbaum e Bos (2016, p. 197).

# SISTEMA DE ARQUIVOS

## ALOCAÇÃO DE ARQUIVO: I-NODES



### 4. I-nodes:

- ▶ Cada arquivos pertence a uma estrutura de dados chamada de i-nodes (*index-node*, nó-índice);
- ▶ Essa estrutura relaciona os atributos e os endereços em disco dos blocos de arquivos;
- ▶ Um i-node é, na realidade, uma estrutura de dados que possui informações sobre um determinado arquivo ou diretório como, por exemplo, dono, grupo, tipo e permissões de acesso;
- ▶ É possível encontrar todos os blocos do arquivo;
- ▶ Se cada i-node ocupar  $n$  bytes e um máximo de  $k$  arquivos puderem estar abertos ao mesmo tempo, a memória ocupada pelo arranjo contendo o i-node para os arquivos abertos é de  $kn$  bytes;
- ▶ Vantagem:
  - ▶ O i-node só precisa estar na memória quando o arquivo correspondente se encontrar aberto.
- ▶ Desvantagem:
  - ▶ O tamanho do arquivo pode aumentar muito: a solução é reservar o último endereço para outros endereços de blocos.



# SISTEMA DE ARQUIVOS

## ALOCAÇÃO DE ARQUIVO: I-NODES

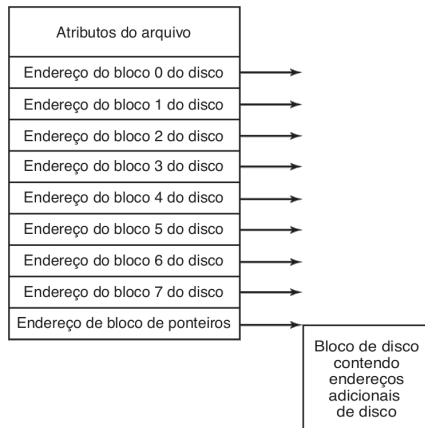


Figura extraída de Tanenbaum e Bos (2016, p. 197).

# SISTEMA DE ARQUIVOS

## IMPLEMENTAÇÃO DE DIRETÓRIO



1. Os atributos dos arquivos devem ser armazenados;
  - ▶ Todo sistema de arquivos mantém os atributos, como proprietário e a hora de sua criação. Eles devem ser armazenados em algum lugar:
    - ▶ Armazená-los diretamente na entrada do diretório:
      - ▶ O diretório consiste em uma lista de entradas de tamanho fixo, um por arquivo, contendo um nome de arquivo (de tamanho fixo), uma estrutura de atributos do arquivo e um ou mais endereços de disco.
    - ▶ Armazená-los nos i-nodes:
      - ▶ Nesse caso, a entrada de diretório pode ser menor, apenas um nome de arquivo e um número de i-node.

# SISTEMA DE ARQUIVOS

## IMPLEMENTAÇÃO DE DIRETÓRIO

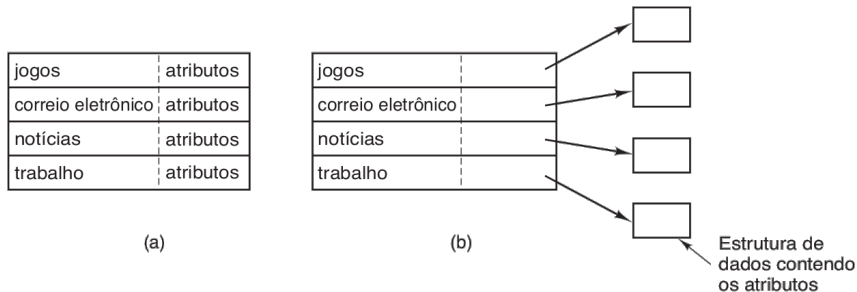


Figura extraída de Tanenbaum e Bos (2016, p. 198).

- (a) Um diretório simples contendo entradas de tamanho fixo com endereços de disco e atributos na entrada do diretório.
  - Windows.
- (b) Um diretório onde cada entrada refere-se a apenas um i-node.
  - Unix.

# SISTEMA DE ARQUIVOS

## IMPLEMENTAÇÃO DE DIRETÓRIO



### 2. Como dar suporte a nomes de arquivos variáveis?

#### ► Uma maneira simples:

- Determinar um limite para o tamanho do nome do arquivo, normalmente em 255 caracteres;
- Estratégia simples;
- Desvantagem que consome uma grande quantidade de espaço no diretório porque poucos arquivos terão nomes longos.

#### ► Mais eficiente:

- É desistir da ideia de que todas as entradas de diretório sejam do mesmo tamanho. Tratar o nome dos arquivos de tamanho variável.

# SISTEMA DE ARQUIVOS

## IMPLEMENTAÇÃO DE DIRETÓRIO

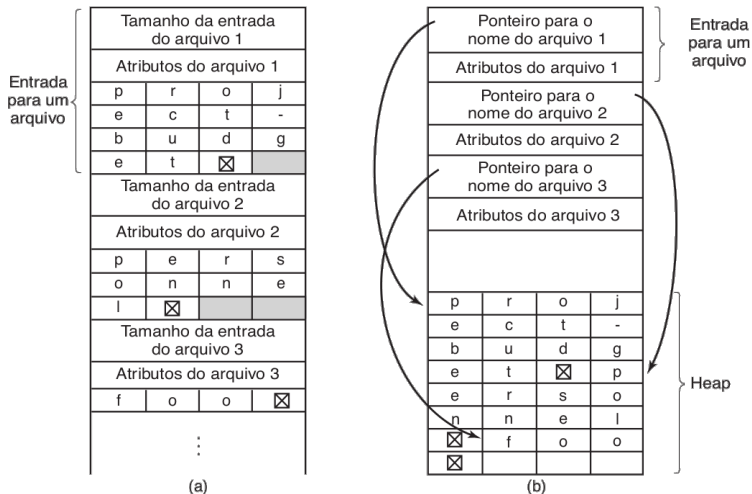


### 3. Tamanho do nome do arquivo variável:

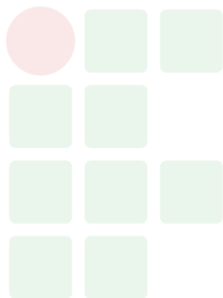
- ▶ Cada entrada de diretório passa a conter uma parte fixa, em geral, começando com o tamanho da entrada;
  - ▶ Seguido dos dados de formato fixo: dados de criação, informações sobre proteção etc.
  - ▶ Esse cabeçalho de tamanho fixo é seguido pelo nome real do arquivo quanto longo for;
  - ▶ Desvantagem é que quando um arquivo é removido, fica um lacuna de tamanho variável no diretório e o próximo arquivo a entrar poderá não caber nela.
- ▶ Outro modo de lidar com tamanho do nome do arquivo variável:
  - ▶ É fazer com que as entradas do diretório sejam todas de tamanho fixo e mantendo ao nomes de arquivos juntos em uma área temporária (*heap*);
  - ▶ Vantajoso quando uma entrada é removida pois o próximo arquivo a entrar sempre caberá.

# SISTEMA DE ARQUIVOS

## IMPLEMENTAÇÃO DE DIRETÓRIO




## REFERÊNCIAS



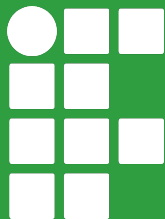
INSTITUTO  
FEDERAL  
Pernambuco

# REFERÊNCIAS I



 TANENBAUM, Andrew S.; BOS, Herbert. **Sistemas Operacionais Modernos**. 4. ed. São Paulo: Pearson Education do Brasil, 2016.





**INSTITUTO FEDERAL**

Pernambuco