

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto dos windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do *primeiro* bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de enfileiramento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.

Supondo por exemplo um sistema cujo bloco tenha 20.000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510×2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500 × 500 blocos (do endereço 12), + 500 × 500 × 500 (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

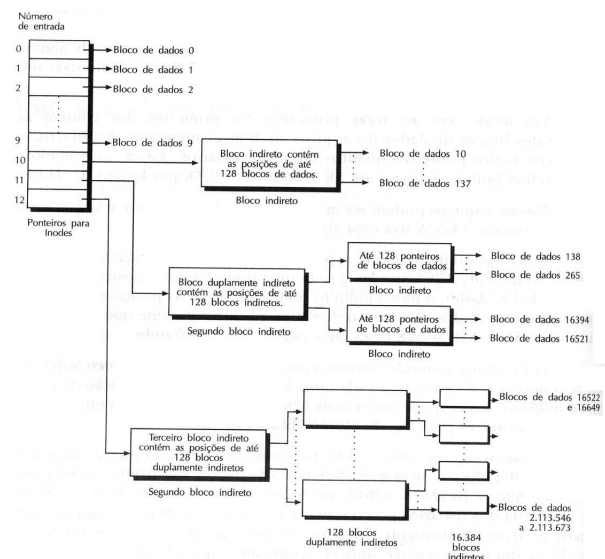


Figura **Acessos UNIX aos blocos de dados**

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
- A área de diretório é separada, e contém:
o nome do arquivo;
espaço para 5 blocos de alocação;
tamanho do arquivo.
- Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
- Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

```
Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expande o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expande o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expande o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expande o arquivo A em mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)
CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A-----A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expande o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D-AA
(20,21,22,23)AAAA
```

👉 Para você fazer

- 1-Cria arquivo A com 100 bytes
- 2-Cria arquivo B com 100 bytes
- 3-Cria arquivo C com 300 bytes
- 4-Cria arquivo D com 500 bytes
- 5-Apaga arquivo A
- 6-Apaga arquivo D
- 7-Expand (se existir) ou Cria (se nao) arquivo A em/com 100 bytes
- 8-Expand (se existir) ou Cria (se nao) arquivo A em/com 500 bytes
- 9-Expand (se existir) ou Cria (se nao) arquivo A em/com 400 bytes
- 10-Cria arquivo H com 500 bytes
- 11-Cria arquivo F com 100 bytes
- 12-Expand (se existir) ou Cria (se nao) arquivo A em/com 500 bytes
- 13-Expand (se existir) ou Cria (se nao) arquivo D em/com 300 bytes
- 14-Cria arquivo E com 500 bytes
- 15-Expand (se existir) ou Cria (se nao) arquivo B em/com 100 bytes
- 16-Expand (se existir) ou Cria (se nao) arquivo B em/com 200 bytes

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

13 35 38 45

Para ajudar, segue aqui o rascunho do superbloco

[illegible]

Dica: nas 4 respostas, duas são cruces e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76506 - ga/ a

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto dos windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do *primeiro* bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de endereçamento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.

Supondo por exemplo um sistema cujo bloco tenha 2000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510 x 2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500×500 blocos (do endereço 12), + $500 \times 500 \times 500$ (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

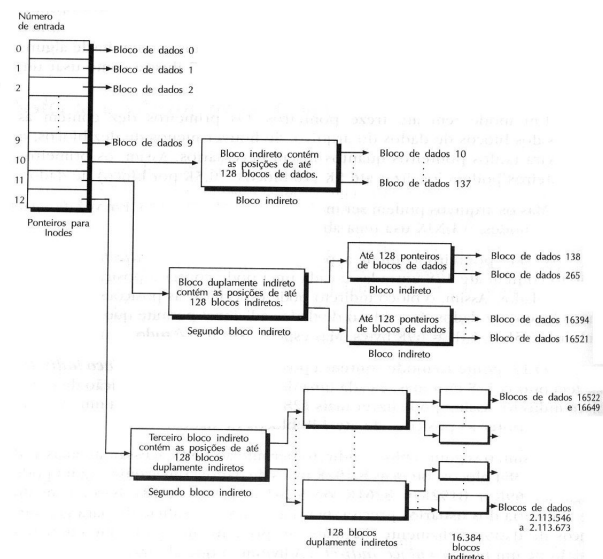


Figura **Acessos UNIX aos blocos de dados**

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
- A área de diretório é separada, e contém:
o nome do arquivo;
espaço para 5 blocos de alocação;
tamanho do arquivo.
- Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
- Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

```
Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expande o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expande o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expande o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expande o arquivo A em mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)
CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A-----A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expande o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AADAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D--AA
(20,21,22,23)AAAA
```

 Para você fazer

- 1-Cria arquivo A com 100 bytes
- 2-Cria arquivo B com 300 bytes
- 3-Cria arquivo C com 200 bytes
- 4-Cria arquivo D com 200 bytes
- 5-Apaga arquivo A
- 6-Expande (se existir) ou Cria (se nao) arquivo C em/com 400 bytes
- 7-Expande (se existir) ou Cria (se nao) arquivo A em/com 500 bytes
- 8-Cria arquivo G com 400 bytes
- 9-Expande (se existir) ou Cria (se nao) arquivo A em/com 500 bytes
- 10-Apaga arquivo D
- 11-Expande (se existir) ou Cria (se nao) arquivo B em/com 500 bytes
- 12-Apaga arquivo B
- 13-Expande (se existir) ou Cria (se nao) arquivo B em/com 100 bytes
- 14-Expande (se existir) ou Cria (se nao) arquivo A em/com 100 bytes
- 15-Cria arquivo H com 300 bytes
- 16-Expande (se existir) ou Cria (se nao) arquivo A em/com 100 bytes

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

14 17 26 27

Para ajudar, segue aqui o rascunho do superbloco

[illegible]

Dica: nas 4 respostas, duas são cruces e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76663 - ga/ a

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto do windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do primeiro bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de endereçamento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.

Supondo por exemplo um sistema cujo bloco tenha 2000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510 x 2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500×500 blocos (do endereço 12), + $500 \times 500 \times 500$ (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

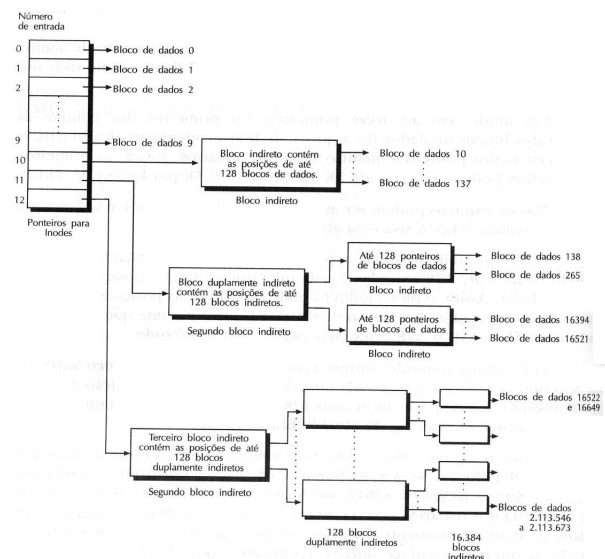


Figura 1. Acessos UNIX aos blocos de dados.

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
- A área de diretório é separada, e contém:
 - o nome do arquivo;
 - espaço para 5 blocos de alocação;
 - tamanho do arquivo.
- Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
- Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expando o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expando o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expando o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expando o arquivo A em mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A-----A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expando o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AAAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D--AA(20,21,22,23)AAAA

Para você fazer

- 1-Cria arquivo A com 500 bytes
- 2-Cria arquivo B com 100 bytes
- 3-Cria arquivo C com 300 bytes
- 4-Cria arquivo D com 500 bytes
- 5-Cria arquivo E com 400 bytes
- 6-Expando (se existir) ou Cria (se nao) arquivo C em/com 400 bytes
- 7-Apaga arquivo B
- 8-Expando (se existir) ou Cria (se nao) arquivo B em/com 300 bytes
- 9-Apaga arquivo A
- 10-Cria arquivo F com 200 bytes
- 11-Apaga arquivo D
- 12-Expando (se existir) ou Cria (se nao) arquivo A em/com 200 bytes
- 13-Cria arquivo G com 200 bytes
- 14-Expando (se existir) ou Cria (se nao) arquivo C em/com 300 bytes
- 15-Apaga arquivo C

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

8	15	20	30
---	----	----	----

Para ajudar, segue aqui o rascunho do superbloco

nome	tamanho	D ₁	D ₂	I ₁	I ₂	I ₃

Dica: nas 4 respostas, duas são cruces e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76513 - ga/ a

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto dos windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do *primeiro* bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de endereçamento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.

Supondo por exemplo um sistema cujo bloco tenha 2000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510 x 2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500×500 blocos (do endereço 12), + $500 \times 500 \times 500$ (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

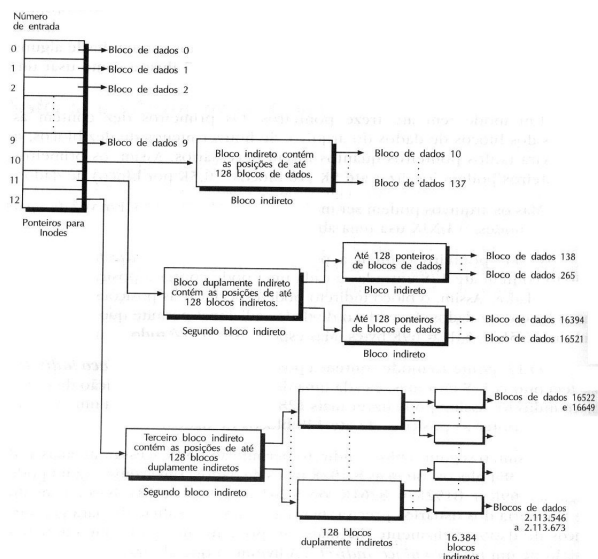


Figura 1 Acessos UNIX aos blocos de dados

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
- A área de diretório é separada, e contém:
o nome do arquivo;
espaço para 5 blocos de alocação;
tamanho do arquivo.
- Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
- Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

```
Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expande o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expande o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expande o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expande o arquivo A com mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)
CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A-----A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expande o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AADAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D--AA
(20,21,22,23)AAAA
```

👉 Para você fazer

- 1-Cria arquivo A com 300 bytes
- 2-Cria arquivo B com 400 bytes
- 3-Cria arquivo C com 500 bytes
- 4-Cria arquivo D com 100 bytes
- 5-Apaga arquivo D
- 6-Expande (se existir) ou Cria (se nao) arquivo B em/com 200 bytes
- 7-Cria arquivo E com 100 bytes
- 8-Expande (se existir) ou Cria (se nao) arquivo B em/com 200 bytes
- 9-Apaga arquivo C
- 10-Expande (se existir) ou Cria (se nao) arquivo A em/com 300 bytes
- 11-Expande (se existir) ou Cria (se nao) arquivo A em/com 100 bytes
- 12-Apaga arquivo B
- 13-Expande (se existir) ou Cria (se nao) arquivo A em/com 200 bytes
- 14-Cria arquivo F com 100 bytes
- 15-Expande (se existir) ou Cria (se nao) arquivo C em/com 100 bytes

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

2 3 12 13

• _____

• _____

• _____

• _____

Para ajudar, segue aqui o rascunho do superbloco

[illegible]

Dica: nas 4 respostas, duas são cruces e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76520 - ga/ a

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto dos windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do *primeiro* bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de endereçamento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.

Supondo por exemplo um sistema cujo bloco tenha 20.000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510×2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500 x 500 blocos (do endereço 12), + 500 x 500 x 500 (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

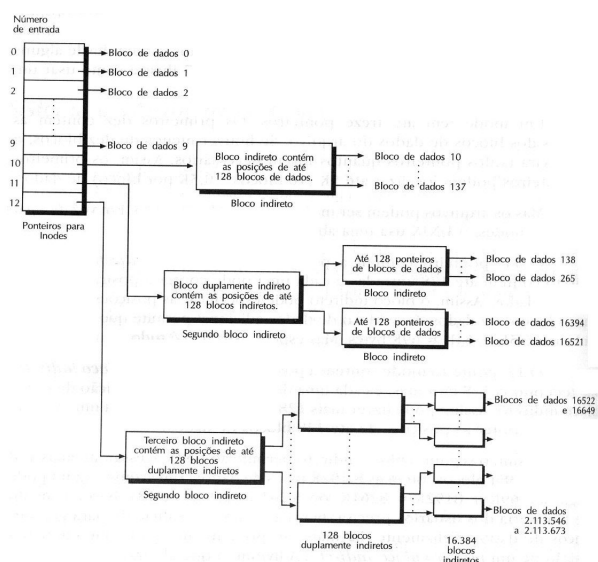


Figura 1 Acessos UNIX aos blocos de dados

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
- A área de diretório é separada, e contém:
o nome do arquivo;
espaço para 5 blocos de alocação;
tamanho do arquivo.
- Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
- Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

```
Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expande o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expande o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expande o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expande o arquivo A em mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)
CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A-----A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expande o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AAAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D-----AA
(20,21,22,23)AAAA
```

👉 Para você fazer

- 1-Cria arquivo A com 100 bytes
- 2-Cria arquivo B com 200 bytes
- 3-Cria arquivo C com 400 bytes
- 4-Cria arquivo D com 400 bytes
- 5-Cria arquivo E com 100 bytes
- 6-Cria arquivo G com 500 bytes
- 7-Cria arquivo F com 100 bytes
- 8-Expande (se existir) ou Cria (se nao) arquivo A em/com 100 bytes
- 9-Expande (se existir) ou Cria (se nao) arquivo A em/com 200 bytes
- 10-Expande (se existir) ou Cria (se nao) arquivo B em/com 100 bytes
- 11-Expande (se existir) ou Cria (se nao) arquivo A em/com 500 bytes
- 12-Expande (se existir) ou Cria (se nao) arquivo C em/com 200 bytes
- 13-Expande (se existir) ou Cria (se nao) arquivo C em/com 100 bytes
- 14-Expande (se existir) ou Cria (se nao) arquivo A em/com 400 bytes
- 15-Apaga arquivo B

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

1	30	33	37
---	----	----	----

Para ajudar, segue aqui o rascunho do superbloco

[illegible]

Dica: nas 4 respostas, duas são cruces e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76537 - ga/ a

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto do windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do *primeiro* bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de endereçamento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.

Supondo por exemplo um sistema cujo bloco tenha 2000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510×2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500 × 500 blocos (do endereço 12), + 500 × 500 × 500 (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

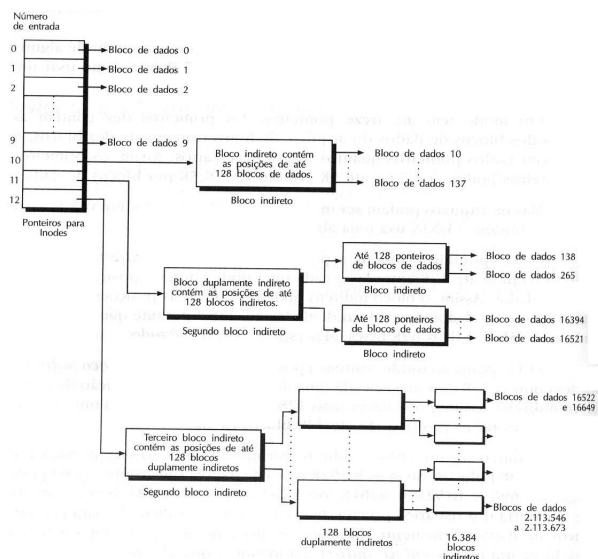


Figura **Acessos UNIX aos blocos de dados**

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
- A área de diretório é separada, e contém:
o nome do arquivo;
espaço para 5 blocos de alocação;
tamanho do arquivo.
- Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
- Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

```
Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expande o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expande o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expande o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expande o arquivo A em mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)
CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A-----A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expande o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AAAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D--AA
(20,21,22,23)AAAA
```

👉 Para você fazer

- 1-Cria arquivo A com 300 bytes
- 2-Cria arquivo B com 200 bytes
- 3-Cria arquivo C com 400 bytes
- 4-Cria arquivo D com 100 bytes
- 5-Cria arquivo E com 500 bytes
- 6-Expande (se existir) ou Cria (se nao) arquivo A em/com 200 bytes
- 7-Cria arquivo G com 100 bytes
- 8-Expande (se existir) ou Cria (se nao) arquivo B em/com 100 bytes
- 9-Expande (se existir) ou Cria (se nao) arquivo B em/com 200 bytes
- 10-Apaga arquivo D
- 11-Apaga arquivo A
- 12-Expande (se existir) ou Cria (se nao) arquivo C em/com 500 bytes
- 13-Expande (se existir) ou Cria (se nao) arquivo C em/com 400 bytes
- 14-Expande (se existir) ou Cria (se nao) arquivo A em/com 100 bytes
- 15-Cria arquivo F com 500 bytes
- 16-Expande (se existir) ou Cria (se nao) arquivo B em/com 300 bytes

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

4 10 12 34

Para ajudar, segue aqui o rascunho do superbloco

[illegible]

Dica: nas 4 respostas, duas são cruces e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76544 - ga/ a

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto do windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do primeiro bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de endereçamento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.

Supondo por exemplo um sistema cujo bloco tenha 2000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510×2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500×500 blocos (do endereço 12), + $500 \times 500 \times 500$ (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

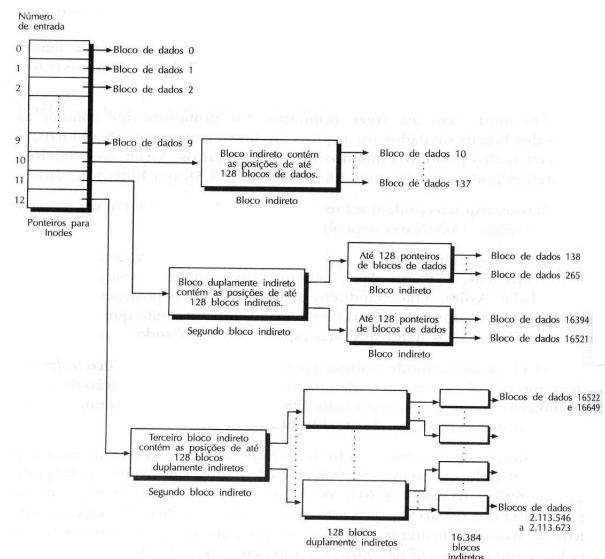


Figura 1. Acessos UNIX aos blocos de dados.

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
- A área de diretório é separada, e contém:
 - o nome do arquivo;
 - espaço para 5 blocos de alocação;
 - tamanho do arquivo.
- Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
- Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

```
Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expando o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expando o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expando o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expando o arquivo A em mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)
CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A-----A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expando o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AAAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D--AA
(20,21,22,23)AAAA
```

Para você fazer

- 1-Cria arquivo A com 300 bytes
- 2-Cria arquivo B com 200 bytes
- 3-Cria arquivo C com 300 bytes
- 4-Cria arquivo D com 200 bytes
- 5-Expando (se existir) ou Cria (se nao) arquivo B em/com 500 bytes
- 6-Expando (se existir) ou Cria (se nao) arquivo A em/com 200 bytes
- 7-Apaga arquivo B
- 8-Expando (se existir) ou Cria (se nao) arquivo A em/com 300 bytes
- 9-Apaga arquivo D
- 10-Apaga arquivo C
- 11-Expando (se existir) ou Cria (se nao) arquivo C em/com 400 bytes
- 12-Cria arquivo E com 300 bytes
- 13-Cria arquivo H com 200 bytes
- 14-Apaga arquivo A
- 15-Cria arquivo F com 100 bytes
- 16-Cria arquivo G com 200 bytes

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

7 9 11 17

Para ajudar, segue aqui o rascunho do superbloco

nome	tamanho	D ₁	D ₂	I ₁	I ₂	I ₃

Dica: nas 4 respostas, duas são cruces e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76551 - ga/ a

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto dos windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do *primeiro* bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de endereçamento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.

Supondo por exemplo um sistema cujo bloco tenha 2000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510 x 2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500×500 blocos (do endereço 12), + $500 \times 500 \times 500$ (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

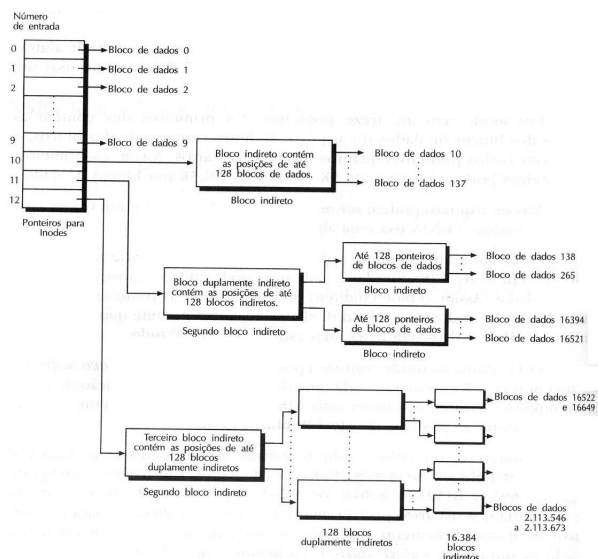


Figura 1 Acessos UNIX aos blocos de dados

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
- A área de diretório é separada, e contém:
o nome do arquivo;
espaço para 5 blocos de alocação;
tamanho do arquivo.
- Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
- Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

```
Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expande o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expande o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expande o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expande o arquivo A com mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)
CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A-----A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expande o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AADAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D--AA
(20,21,22,23)AAAA
```

 Para você fazer

- 1-Cria arquivo A com 500 bytes
- 2-Cria arquivo B com 200 bytes
- 3-Cria arquivo C com 500 bytes
- 4-Cria arquivo D com 400 bytes
- 5-Expand (se existir) ou Cria (se nao) arquivo B em/com 300 bytes
- 6-Apaga arquivo B
- 7-Expand (se existir) ou Cria (se nao) arquivo C em/com 100 bytes
- 8-Expand (se existir) ou Cria (se nao) arquivo C em/com 400 bytes
- 9-Cria arquivo G com 500 bytes
- 10-Cria arquivo H com 300 bytes
- 11-Apaga arquivo D
- 12-Expand (se existir) ou Cria (se nao) arquivo B em/com 200 bytes
- 13-Expand (se existir) ou Cria (se nao) arquivo A em/com 300 bytes
- 14-Apaga arquivo A
- 15-Expand (se existir) ou Cria (se nao) arquivo B em/com 300 bytes
- 16-Cria arquivo E com 500 bytes

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

18	20	33	34
----	----	----	----

Para ajudar, segue aqui o rascunho do superbloco

[illegible]

Dica: nas 4 respostas, duas são cruces e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76568 - ga/ a

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto dos windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do *primeiro* bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de endereçamento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.

Supondo por exemplo um sistema cujo bloco tenha 2000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510 x 2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500×500 blocos (do endereço 12), + $500 \times 500 \times 500$ (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

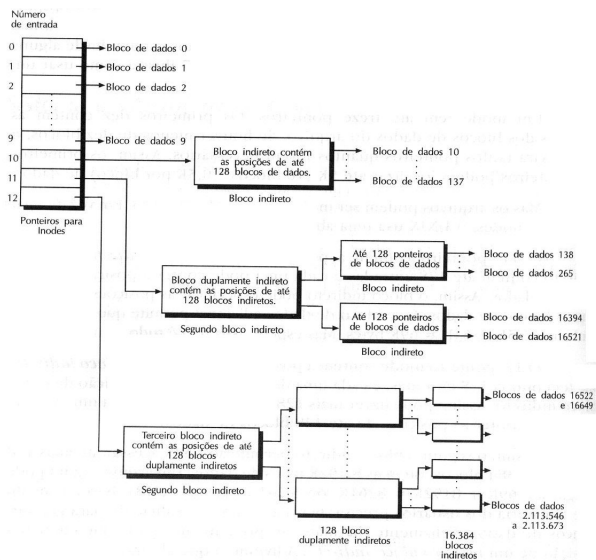


Figura **Acessos UNIX aos blocos de dados**

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
- A área de diretório é separada, e contém:
o nome do arquivo;
espaço para 5 blocos de alocação;
tamanho do arquivo.
- Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
- Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

```
Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expande o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expande o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expande o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expande o arquivo A em mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)
CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A-----A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expande o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AAAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D--AA
(20,21,22,23)AAAA
```

👉 Para você fazer

- 1-Cria arquivo A com 100 bytes
- 2-Cria arquivo B com 400 bytes
- 3-Cria arquivo C com 200 bytes
- 4-Cria arquivo D com 400 bytes
- 5-Expande (se existir) ou Cria (se nao) arquivo B em/com 200 bytes
- 6-Cria arquivo H com 500 bytes
- 7-Expande (se existir) ou Cria (se nao) arquivo A em/com 300 bytes
- 8-Expande (se existir) ou Cria (se nao) arquivo D em/com 200 bytes
- 9-Expande (se existir) ou Cria (se nao) arquivo A em/com 400 bytes
- 10-Expande (se existir) ou Cria (se nao) arquivo B em/com 300 bytes
- 11-Apaga arquivo A
- 12-Expande (se existir) ou Cria (se nao) arquivo B em/com 200 bytes
- 13-Apaga arquivo C
- 14-Expande (se existir) ou Cria (se nao) arquivo C em/com 500 bytes
- 15-Expande (se existir) ou Cria (se nao) arquivo C em/com 200 bytes

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

9 22 29 35

*** _____**

*** _____**

*** _____**

*** _____**

Para ajudar, segue aqui o rascunho do superbloco

[illegible]

Dica: nas 4 respostas, duas são cruces e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76575 - ga/ a

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto dos windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do *primeiro* bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de enfileiramento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.

Supondo por exemplo um sistema cujo bloco tenha 20.000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510×2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500 × 500 blocos (do endereço 12), + 500 × 500 × 500 (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

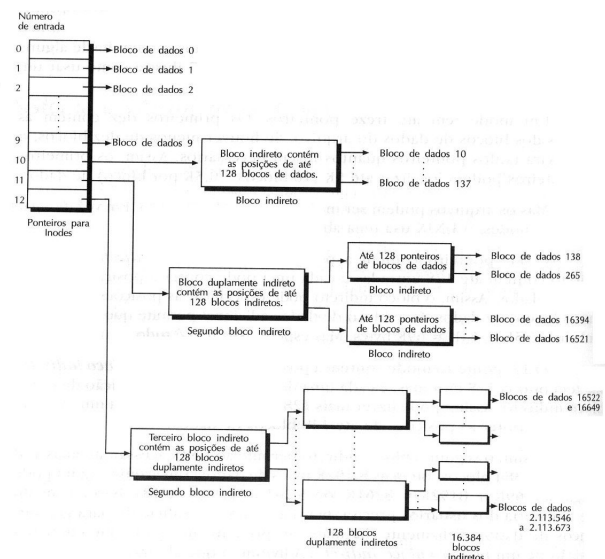


Figura **Acessos UNIX aos blocos de dados**

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
- A área de diretório é separada, e contém:
o nome do arquivo;
espaço para 5 blocos de alocação;
tamanho do arquivo.
- Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
- Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

```
Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expande o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expande o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expande o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expande o arquivo A em mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)
CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A-----A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expande o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D-AA
(20,21,22,23)AAAA
```

👉 Para você fazer

- 1-Cria arquivo A com 100 bytes
- 2-Cria arquivo B com 300 bytes
- 3-Cria arquivo C com 100 bytes
- 4-Cria arquivo D com 300 bytes
- 5-Expande (se existir) ou Cria (se nao) arquivo A em/com 100 bytes
- 6-Apaga arquivo A
- 7-Apaga arquivo C
- 8-Cria arquivo F com 100 bytes
- 9-Expande (se existir) ou Cria (se nao) arquivo B em/com 200 bytes
- 10-Expande (se existir) ou Cria (se nao) arquivo B em/com 300 bytes
- 11-Expande (se existir) ou Cria (se nao) arquivo A em/com 400 bytes
- 12-Apaga arquivo D
- 13-Expande (se existir) ou Cria (se nao) arquivo A em/com 400 bytes
- 14-Expande (se existir) ou Cria (se nao) arquivo D em/com 300 bytes
- 15-Expande (se existir) ou Cria (se nao) arquivo B em/com 500 bytes
- 16-Expande (se existir) ou Cria (se nao) arquivo C em/com 100 bytes

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

Transcreva a seguir, qual o conteúdo da cerâmica numerada.

4	19	33	35
.....			

Para ajudar, segue aqui o rascunho do superbloco

[illegible]

Dica: nas 4 respostas, duas são cruces e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76582 - ga/ a

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto do windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do *primeiro* bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de endereçamento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.

Supondo por exemplo um sistema cujo bloco tenha 2000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510×2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500 × 500 blocos (do endereço 12), + 500 × 500 × 500 (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

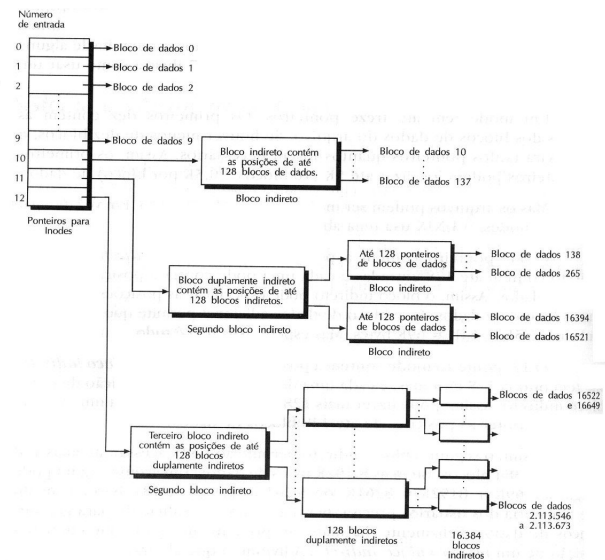


Figura **Acessos UNIX aos blocos de dados**

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
- A área de diretório é separada, e contém:
o nome do arquivo;
espaço para 5 blocos de alocação;
tamanho do arquivo.
- Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
- Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

```
Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expande o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expande o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expande o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expande o arquivo A com mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)
CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A-----A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expande o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AADAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D--AA
(20,21,22,23)AAAA
```

 Para você fazer

- 1-Cria arquivo A com 100 bytes
- 2-Cria arquivo B com 300 bytes
- 3-Cria arquivo C com 500 bytes
- 4-Cria arquivo D com 400 bytes
- 5-Expande (se existir) ou Cria (se nao) arquivo A em/com 200 bytes
- 6-Expande (se existir) ou Cria (se nao) arquivo D em/com 500 bytes
- 7-Expande (se existir) ou Cria (se nao) arquivo C em/com 300 bytes
- 8-Expande (se existir) ou Cria (se nao) arquivo B em/com 500 bytes
- 9-Apaga arquivo A
- 10-Expande (se existir) ou Cria (se nao) arquivo A em/com 400 bytes
- 11-Apaga arquivo C
- 12-Expande (se existir) ou Cria (se nao) arquivo B em/com 200 bytes
- 13-Apaga arquivo B
- 14-Cria arquivo F com 400 bytes
- 15-Expande (se existir) ou Cria (se nao) arquivo C em/com 300 bytes
- 16-Apaga arquivo D

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

4	18	21	38
_____	_____	_____	_____

Para ajudar, segue aqui o rascunho do superbloco

[illegible]

Dica: nas 4 respostas, duas são cruces e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76599 - ga/ a

302-76601 - ga/ a

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto do windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do primeiro bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de endereçamento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.

Supondo por exemplo um sistema cujo bloco tenha 2000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510 x 2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500×500 blocos (do endereço 12), + $500 \times 500 \times 500$ (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

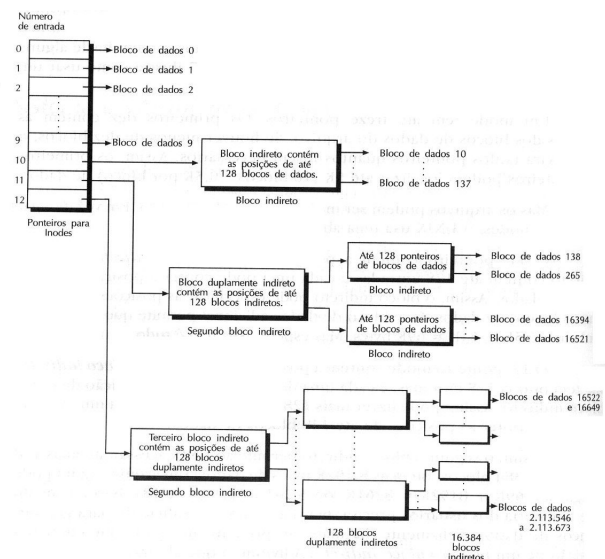


Figura 1. Acessos UNIX aos blocos de dados.

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
- A área de diretório é separada, e contém:
 - o nome do arquivo;
 - espaço para 5 blocos de alocação;
 - tamanho do arquivo.
- Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
- Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expando o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expando o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expando o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expando o arquivo A em mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)
CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A-----A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expando o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AAAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D--AA
(20,21,22,23)AAAA

Para você fazer

- 1-Cria arquivo A com 400 bytes
- 2-Cria arquivo B com 400 bytes
- 3-Cria arquivo C com 200 bytes
- 4-Cria arquivo D com 300 bytes
- 5-Cria arquivo E com 200 bytes
- 6-Expando (se existir) ou Cria (se nao) arquivo A em/com 400 bytes
- 7-Expando (se existir) ou Cria (se nao) arquivo B em/com 200 bytes
- 8-Expando (se existir) ou Cria (se nao) arquivo A em/com 500 bytes
- 9-Expando (se existir) ou Cria (se nao) arquivo B em/com 400 bytes
- 10-Expando (se existir) ou Cria (se nao) arquivo C em/com 300 bytes
- 11-Apaga arquivo D
- 12-Apaga arquivo B
- 13-Expando (se existir) ou Cria (se nao) arquivo D em/com 400 bytes
- 14-Expando (se existir) ou Cria (se nao) arquivo C em/com 200 bytes
- 15-Expando (se existir) ou Cria (se nao) arquivo A em/com 500 bytes
- 16-Cria arquivo H com 300 bytes

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

2 3 35 43

Para ajudar, segue aqui o rascunho do superbloco

nome	tamanho	D ₁	D ₂	I ₁	I ₂	I ₃

Dica: nas 4 respostas, duas são cruces e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76618 - ga/ a

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto do windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do *primeiro* bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de endereçamento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.

Supondo por exemplo um sistema cujo bloco tenha 2000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510×2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500 × 500 blocos (do endereço 12), + 500 × 500 × 500 (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

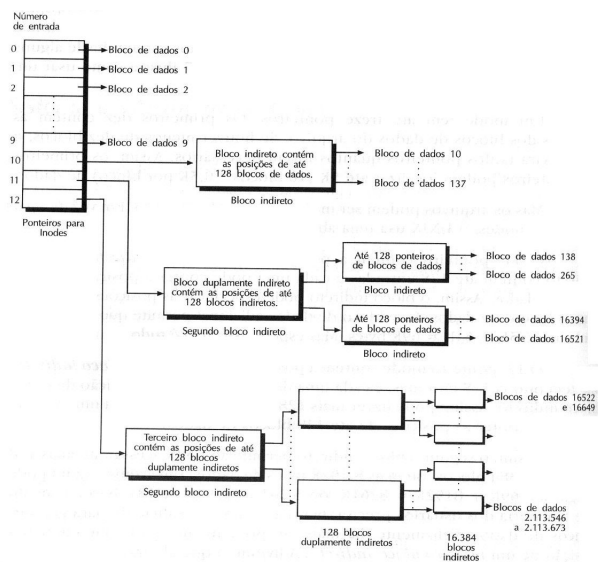


Figura 1 Acessos UNIX aos blocos de dados

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
- A área de diretório é separada, e contém:
o nome do arquivo;
espaço para 5 blocos de alocação;
tamanho do arquivo.
- Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
- Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

```
Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expande o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expande o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expande o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expande o arquivo A com mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)
CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A-----A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expande o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AAAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D--AA
(20,21,22,23)AAAA
```

 Para você fazer

- 1-Cria arquivo A com 200 bytes
- 2-Cria arquivo B com 400 bytes
- 3-Cria arquivo C com 300 bytes
- 4-Cria arquivo D com 500 bytes
- 5-Cria arquivo H com 400 bytes
- 6-Apaga arquivo A
- 7-Apaga arquivo D
- 8-Expande (se existir) ou Cria (se nao) arquivo A em/com 200 bytes
- 9-Expande (se existir) ou Cria (se nao) arquivo C em/com 300 bytes
- 10-Expande (se existir) ou Cria (se nao) arquivo D em/com 300 bytes
- 11-Apaga arquivo C
- 12-Expande (se existir) ou Cria (se nao) arquivo A em/com 500 bytes
- 13-Expande (se existir) ou Cria (se nao) arquivo B em/com 500 bytes
- 14-Expande (se existir) ou Cria (se nao) arquivo A em/com 400 bytes
- 15-Cria arquivo G com 200 bytes

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

5 7 27 30

• _____

• _____

• _____

• _____

Para ajudar, segue aqui o rascunho do superbloco

[illegible]

Dica: nas 4 respostas, duas são cruces e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76625 - ga/ a

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto do windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.	ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA Apaga o arquivo B A-----A(10,11,17,18)AACC(15,16,0,0)CCAA Expande o arquivo C em mais 300 bytes (total=700) ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA Expande o arquivo A em mais 400 bytes (total=1000)
---	--

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do *primeiro* bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de endereçamento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.	Cria o arquivo D com 300 bytes <pre>A(3,4,5,0)AAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D--AA(20,21,22,23)AAAA</pre>
--	---

Supondo por exemplo um sistema cujo bloco tenha 2000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510×2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500×500 blocos (do endereço 12), + $500 \times 500 \times 500$ (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

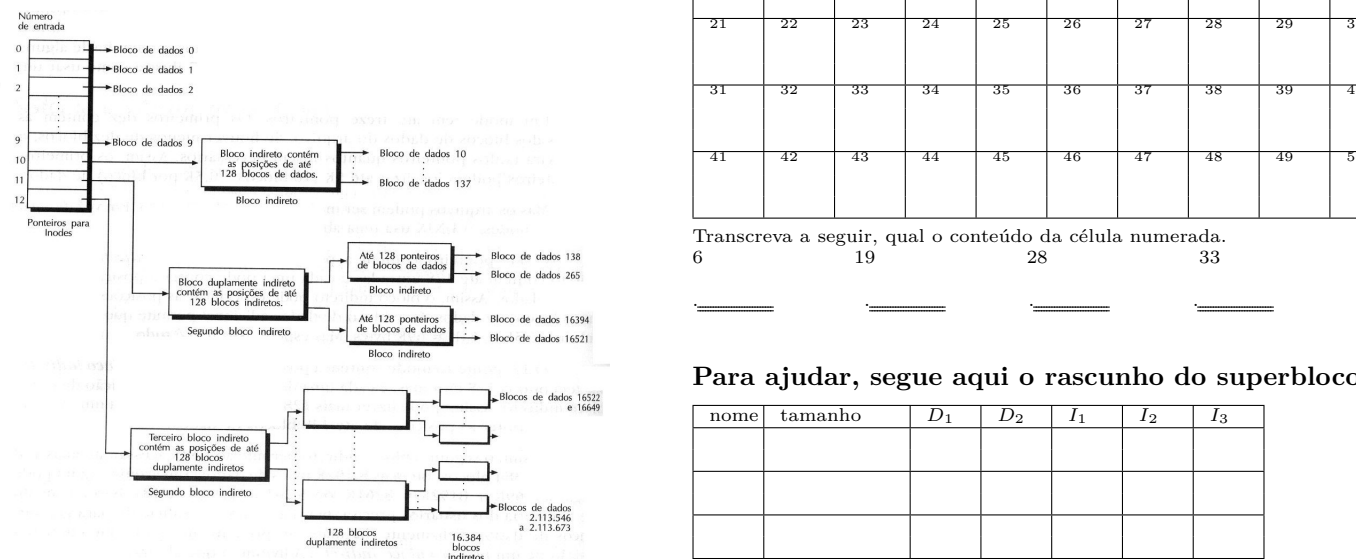


Figura 1. Acessos UNIX aos blocos de dados

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
 - A área de diretório é separada, e contém:
o nome do arquivo;
espaço para 5 blocos de alocação;
tamanho do arquivo.
 - Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
 - Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

```
Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expande o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expande o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expande o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expande o arquivo A em mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)
CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A-----A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expande o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-A(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AAAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D--AA
(20,21,22,23)AAAA
```

 Para você fazer

- 1-Cria arquivo A com 300 bytes
- 2-Cria arquivo B com 200 bytes
- 3-Cria arquivo C com 200 bytes
- 4-Cria arquivo D com 100 bytes
- 5-Expande (se existir) ou Cria (se nao) arquivo A em/com 300 bytes
- 6-Expande (se existir) ou Cria (se nao) arquivo C em/com 500 bytes
- 7-Expande (se existir) ou Cria (se nao) arquivo A em/com 300 bytes
- 8-Cria arquivo H com 200 bytes
- 9-Expande (se existir) ou Cria (se nao) arquivo B em/com 200 bytes
- 10-Cria arquivo E com 100 bytes
- 11-Expande (se existir) ou Cria (se nao) arquivo A em/com 100 bytes
- 12-Expande (se existir) ou Cria (se nao) arquivo C em/com 200 bytes
- 13-Expande (se existir) ou Cria (se nao) arquivo B em/com 300 bytes
- 14-Apaga arquivo A
- 15-Cria arquivo F com 200 bytes

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

6	19	28	33
---	----	----	----

Para ajudar, segue aqui o rascunho do superbloco

[illegible]

Dica: nas 4 respostas, duas são cruzeiros e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76632 - ga/ a

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto do windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do *primeiro* bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de endereçamento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.

Supondo por exemplo um sistema cujo bloco tenha 2000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510×2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500 × 500 blocos (do endereço 12), + 500 × 500 × 500 (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

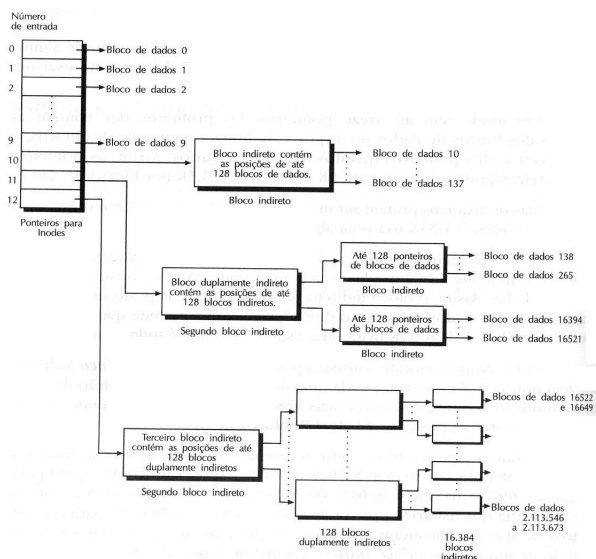


Figura 1 Acessos UNIX aos blocos de dados

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
- A área de diretório é separada, e contém:
o nome do arquivo;
espaço para 5 blocos de alocação;
tamanho do arquivo.
- Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
- Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

```
Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expande o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expande o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expande o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expande o arquivo A com mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)
CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A-----A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expande o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AAAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D--AA
(20,21,22,23)AAAA
```

👉 Para você fazer

- 1-Cria arquivo A com 100 bytes
- 2-Cria arquivo B com 300 bytes
- 3-Cria arquivo C com 100 bytes
- 4-Cria arquivo D com 400 bytes
- 5-Expand (se existir) ou Cria (se nao) arquivo C em/com 100 bytes
- 6-Expand (se existir) ou Cria (se nao) arquivo D em/com 500 bytes
- 7-Cria arquivo E com 100 bytes
- 8-Cria arquivo G com 200 bytes
- 9-Expand (se existir) ou Cria (se nao) arquivo B em/com 100 bytes
- 10-Expand (se existir) ou Cria (se nao) arquivo B em/com 400 bytes
- 11-Expand (se existir) ou Cria (se nao) arquivo C em/com 300 bytes
- 12-Expand (se existir) ou Cria (se nao) arquivo A em/com 100 bytes
- 13-Apaga arquivo C
- 14-Expand (se existir) ou Cria (se nao) arquivo A em/com 200 bytes
- 15-Apaga arquivo B

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

9 15 20 34

* _____

Para ajudar, segue aqui o rascunho do superbloco

[illegible]

Dica: nas 4 respostas, duas são cruces e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76649 - ga/ a

Algoritmo de organização do disco Unix/Linux

O sistema operacional UNIX utiliza um esquema completamente distinto dos windows para alocações em disco. Similar ao diretório do DOS, há uma área nos discos UNIX onde o sistema coloca a lista de arquivos existentes no disco. Ao lado do nome e dos atributos (quem pode ler, gravar e executar os arquivos) há espaço para a colocação de 13 endereços de blocos de alocação do disco associados a um único arquivo.

Os 10 primeiros blocos, apontam diretamente para o bloco de dados. Assim se um determinado arquivo contiver apenas até 10 blocos de dados, os endereços desses blocos estarão diretamente colocados na lista de arquivos do disco. (Em contraposição ao DOS que guarda apenas o endereço do *primeiro* bloco de dados). Para arquivos pequenos, este esquema favorece o acesso e a facilidade de endereçamento dos dados.

Quando o arquivo se expandir além dos 10 blocos, passa-se a usar o endereço número 11. Este não aponta para dados e sim aponta para um bloco que é subdividido em apontadores para blocos de dados. Este bloco tem portanto um nível de indireção quando aponta para os dados.

Supondo por exemplo um sistema cujo bloco tenha 2000 bytes e cujos números de bloco tenham 4 bytes. Neste sistema o endereçamento direto (10 blocos iniciais) permite usar 20.000 bytes. Se este espaço for insuficiente, alocar-se-á o bloco 11 e este permitirá endereçar mais 500 blocos de dados ($2000 \div 4 = 500$). Ao usar este endereço o arquivo pode chegar a 510 blocos (ou 1.020.000 bytes).

Se o arquivo continuar crescendo, alocar-se-á o endereço 12, que terá dois níveis de indireção. Portanto, o endereço 12 aponta para um bloco de endereços que apontam para blocos de endereços que apontam para blocos de dados.

Usando o mesmo exemplo anterior, ao usar o endereço 12, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + $500 \times 500 = 250.000$ blocos (do endereço 12), totalizando 250.510 blocos ou 250.510 x 2000 bytes.

Finalmente, se o arquivo se tornar imenso, alocar-se-á o endereço 13, que terá 3 níveis de indireção. Os endereços 1 a 10 continuam sendo usados de maneira direta, o 11 com 1 e o 12 com 2 níveis de indireção e finalmente o 13 com 3 níveis de indireção.

Ainda o mesmo exemplo anterior. Usando o endereço 13, o sistema passa a poder endereçar 10 blocos (dos endereços diretos) + 500 blocos (do endereço 11) + 500×500 blocos (do endereço 12), + $500 \times 500 \times 500$ (do endereço 13) totalizando 125.250.510 blocos ou 250.000.000.000 bytes.

Note-se que é uma estrutura encadeada, o que não obriga (nem isso seria possível) a contigüidade física dos dados.

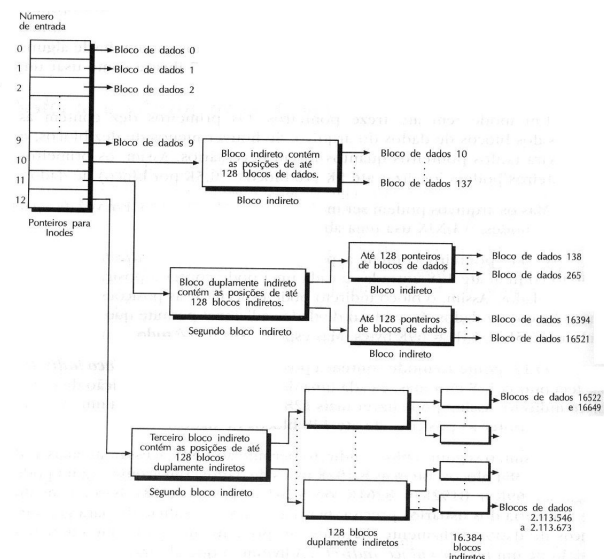


Figura 1 Acessos UNIX aos blocos de dados

Instruções:

- Para efeitos deste exercício, considere-se um disco contendo 80 blocos de 100 bytes cada um.
- A área de diretório é separada, e contém:
o nome do arquivo;
espaço para 5 blocos de alocação;
tamanho do arquivo.
- Desses 5 blocos, os 2 primeiros são diretos, o terceiro é indireto o quarto é duplamente indireto e o quinto é triplamente indireto.
- Os blocos que forem utilizados para apontadores de espaço terão lugar apenas para 4 apontadores (é como se cada apontador consumisse 25 bytes ($100 \div 4 = 25$)).

Antes de fazer o seu exercício, acompanhe este exemplo

```
Cria o arquivo A com 100 bytes
A
Cria o arquivo B com 500 bytes
ABB(5,6,7,0)BBB
Expande o arquivo A em mais 300 bytes (total=400)
ABB(5,6,7,0)BBBA(10,11,0,0)AA
Cria o arquivo C com 400 bytes
ABB(5,6,7,0)BBBA(10,11,0,0)AACC(15,16,0,0)CC
Expande o A em mais 200 bytes (total=600)
ABB(5,6,7,0)BBBA(10,11,17,18)AACC(15,16,0,0)CCAA
Apaga o arquivo B
A-----A(10,11,17,18)AACC(15,16,0,0)CCAA
Expande o arquivo C em mais 300 bytes (total=700)
ACC(5,0,0,0)(6,0,0,0)C-A(10,11,17,18)AACC(15,16,2,3)CCAA
Expande o arquivo A em mais 400 bytes (total=1000)
ACC(5,0,0,0)(6,0,0,0)C(19,0,0,0)A(10,11,17,18)AACC(15,16,2,3)
CCAA(20,21,22,23)AAAA
Apaga o arquivo C
A----- (19,0,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Expande o arquivo A com mais 300 (total=1300)
A(3,4,5,0)AAA-(19,2,0,0)A(10,11,17,18)AA-----AA(20,21,22,23)AAAA
Cria o arquivo D com 300 bytes
A(3,4,5,0)AAD(19,2,0,0)A(10,11,17,18)AAD(14,0,0,0)D--AA
(20,21,22,23)AAAA
```

👉 Para você fazer

- 1-Cria arquivo A com 300 bytes
- 2-Cria arquivo B com 100 bytes
- 3-Cria arquivo C com 100 bytes
- 4-Cria arquivo D com 100 bytes
- 5-Expand (se existir) ou Cria (se nao) arquivo A em/com 400 bytes
- 6-Expand (se existir) ou Cria (se nao) arquivo A em/com 400 bytes
- 7-Apaga arquivo C
- 8-Expand (se existir) ou Cria (se nao) arquivo D em/com 200 bytes
- 9-Apaga arquivo B
- 10-Expand (se existir) ou Cria (se nao) arquivo B em/com 300 bytes
- 11-Cria arquivo H com 300 bytes
- 12-Expand (se existir) ou Cria (se nao) arquivo A em/com 300 bytes
- 13-Expand (se existir) ou Cria (se nao) arquivo C em/com 300 bytes
- 14-Apaga arquivo A
- 15-Expand (se existir) ou Cria (se nao) arquivo B em/com 500 bytes
- 16-Cria arquivo G com 200 bytes

Use esta grade para responder quais os blocos resultantes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Transcreva a seguir, qual o conteúdo da célula numerada.

7	8	19	29
_____	_____	_____	_____

Para ajudar, segue aqui o rascunho do superbloco

[illegible]

Dica: nas 4 respostas, duas são cruces e duas não são.

Se faltar espaço, complemente no caderno, que a folha A4 é limitada.



302-76656 - ga/ a