Instituto Superior de Engenharia do Porto
Departamento de Engenharia Informática

SISTEMAS OPERATIVOS I

Texto de Apoio às Aulas Práticas

Ficheiros e Directórios

baseado no livro
"UNIX For Application Developers"
William A. Parrete

Abril de 2002

Lino Oliveira

Sugestões e participações de erros para: lino@dei.isep.ipp.pt

FICHEIROS E DIRECTÓRIOS

Índice

1 Ficheiros E Directórios 3

1.1 Directórios 3

1.1.1 Directórios Standard 3
1.1.2 Home Directory 4
1.1.3 Directório De Trabalho (Current Working Directory) 4
1.1.4 Pathnames 4
1.1.5 Directórios Especiais 4
1.1.6 Mudar De Directório 4
1.1.7 Criar Novos Directórios 5
1.1.8 Eliminar Directórios 5

1.2 Ficheiros 5

1.2.1 Listar Nome De Ficheiros 5
1.2.2 Determinar O Que Está Dentro De Um Ficheiro 6
1.2.3 Ver O Conteúdo De Um Ficheiro 6
1.2.4 Ver O Conteúdo De Um Ficheiro Página A Página 6
1.2.5 Ver A Parte Final Dum Ficheiro 7
1.2.6 Ver A Parte Inicial Dum Ficheiro 7
1.2.7 Copiar Um Ficheiro 8
1.2.8 Mover Ou Renomear Ficheiro 8
1.2.9 Criar Ou Alterar Nome Alternativo Para Ficheiro 8
1.2.10 Comparação Entre Cp, Mv E Ln 9
1.2.11 Eliminar Ficheiros 10
1.2.12 Utilizando *Pathnames* 11

1.3 Questões 11

1 FICHEIROS E DIRECTÓRIOS

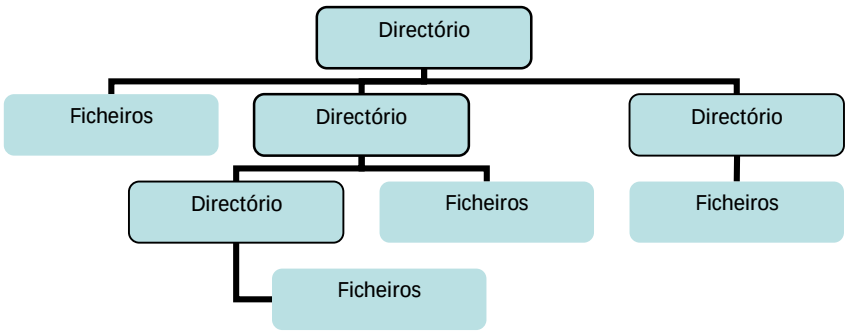
Os ficheiros e os directórios fazem parte do Sistema de Ficheiros:

- "Local" onde o sistema operativo e os utilizadores guardam e organizam os seus ficheiros
- o Unix impõe uma estrutura onde armazena os ficheiros o que facilita o armazenamento e a procura posterior

1.1 DIRECTÓRIOS

- É utilizado para guardar ficheiros relacionados num mesmo local
- Analogia com as pastas
- Pode conter ficheiros ou outros directórios (seguindo a analogia, podemos ter pastas com documentos e/ou outras pastas que, por sua vez podem ter documentos e/ou outras pastas, e assim sucessivamente)
- É tratado pelo Unix com um tipo especial de ficheiro
- Não há limite para o número de níveis de directórios

O Sistema de Ficheiros é uma estrutura em árvore invertida.



1.1.1 DIRECTÓRIOS STANDARD

/ - directório base (root)

bin - comandos essenciais do Unix

dev - dispositivos (devices) ligados ao computador

etc - comandos e ficheiros usados pelo administrador do sistema

lib - bibliotecas (conjunto de ficheiros relacionados contidos num único ficheiro) usadas por compiladores, processadores de texto e outros comandos Unix

lost+found - usado pelo `fsck`, programa especial usado pelo administrador do sistema para verificar o sistema; se o `fsck` encontra algum ficheiro que pareça não estar ligado a nenhum directório, o programa liga-o ao *lost+found*, para que posteriormente o administrador do sistema decida o que fazer com ele

tmp - usado por diversos comandos Unix para criação de ficheiros temporários; pode ser usado por qualquer utilizador; este directório é limpo regularmente

usr - é a parte do sistema de ficheiros pertencente aos utilizadores; é a partir deste directório que se estendem os directórios de todos os utilizadores do sistema

1.1.2 HOME DIRECTORY

- Directório ao qual temos acesso logo após o login
- Em muitos sistemas Unix, nomeadamente no Linux, pode ser abreviado com o sinal "~"

1.1.3 DIRECTÓRIO DE TRABALHO (CURRENT WORKING DIRECTORY)

- Directório onde "estamos" em cada momento
- Quando fazemos login, o nosso *home directory* é simultaneamente o nosso *current working directory*

Podemos determinar o nosso directório de trabalho executando o comando

```
pwd
```

que nos apresenta o caminho completo desde a raiz (/) até ao nosso directório de trabalho. Por exemplo:

```
$ pwd
/users/2/lino
$
```

1.1.4 PATHNAMES

- Referenciam um ficheiro ou directório no sistema de ficheiros
- Indicam o caminho a seguir através do sistema de ficheiro para encontrar o ficheiro ou directório

Tipos de Pathnames:

Full pathname - caminho completo: nome do ficheiro ou directório em relação à raiz do sistema; começa sempre com "/" e apresenta cada um dos directórios separados por "/"; por cada ficheiro ou directório só existe um e um só full pathname

Relative pathname - caminho relativo: nome do ficheiro ou directório relativamente ao directório corrente de trabalho; nunca começa com "/"

Simple pathname - caminho simples: nome do ficheiro ou directório que está directamente abaixo, ou no interior, do directório corrente de trabalho

1.1.5 DIRECTÓRIOS ESPECIAIS

Presentes em cada um dos directórios:

"." - (ponto) abreviatura do directório corrente
".." - (ponto-ponto) abreviatura do directório imediatamente acima do corrente (directório pai)

1.1.6 MUDAR DE DIRECTÓRIO

O comando para mudar de directório é o *cd* (change directory) cuja sintaxe é a seguinte:

```
cd [ directório ]
```

O directório deve existir e temos de ter permissão para lá estar. Se não usarmos nenhum directório, isto é, se executarmos *cd* sem nenhum parâmetro, mudamos para a nossa *home directory*.

1.1.7 CRIAR NOVOS DIRECTÓRIOS

É possível organizarmos os nossos ficheiros no nosso *home directory* da mesma maneira que o Unix o faz na *root directory*. Para tal ser possível é necessário termos a possibilidade de criarmos directórios.

Tal é conseguido com o comando *mkdir (make directory)* cuja sintaxe é a seguinte:

```
mkdir directório ...
```

Se tentarmos criar um directório que já existe ou num sítio no qual não temos permissão para o fazer, *mkdir* apresentará uma mensagem de erro.

Os nomes a utilizar na criação de directórios seguem as mesmas regras que os nomes dos ficheiros: existe distinção entre maiúsculas e minúsculas. Algumas pessoas seguem a convenção de criar os directórios em maiúsculas o que permite uma mais fácil distinção dos ficheiros.

1.1.8 ELIMINAR DIRECTÓRIOS

Quando já não precisamos do directório podemos eliminá-lo com o comando *rmdir* cuja sintaxe é a seguinte:

```
rmdir directório ...
```

Para que um directório possa ser eliminado, duas condições devem ser respeitadas:

- o directório tem de estar vazio: é necessário apagar os ficheiros contidos dentro do directório a eliminar antes de proceder à operação
- o directório corrente não pode ser o directório que estamos a tentar eliminar; o que faz sentido: em que parte do sistema de ficheiros seríamos “colocados” se eliminássemos o nosso directório de trabalho corrente ?

O *rmdir* não pede confirmação antes de proceder à eliminação. Como muitos outros comandos de Unix, o *rmdir* assume que sabemos o que estamos a fazer e deixa-nos fazer quase tudo o que queremos sem nos interromper com mensagens triviais!

1.2 FICHEIROS

1.2.1 LISTAR NOME DE FICHEIROS

Para vermos os ficheiros contidos nos directórios usamos o comando *ls*. Quando usado sem parâmetros, permite ver os ficheiros e os directórios do directório corrente. Eis alguns exemplos:

`ls dir` - lista os fiheiros contido no directório e não o directório em si; se quisermos ver os ficheiros de um directório não precisamos de fazer cd

`ls –d dir` - permite ver o nome do directório e não o seu conteúdo; trata o directório como um ficheiro

`ls –F` - lista os nomes, identificando os directórios com um “/” e os ficheiros executáveis com “*” (para obtermos este resultado deveremos estar a usar bash)

ls –R - modo recursivo: lista os directórios e respectivos conteúdos.
Exemplo: ls –R / | more

ls –i - lista o número do *i-node* associado a cada ficheiro; o *i-node* é um identificador único usado pelo Unix para "seguir a pista" de cada ficheiro do *file system*

1.2.2 DETERMINAR O QUE ESTÁ DENTRO DE UM FICHEIRO

O comando *file* é usado para determinar que tipo de dados estão contidos num ficheiro. A sintaxe é a seguinte:

```
file [ -f nome_ficheiro ] ficheiro ...
```

O *file* examina o conteúdo dos ficheiros indicados como parâmetros do comando, juntamente com outros dados armazenados pelo sistema de ficheiros em cada ficheiro, no sentido de determinar o tipo de informação guardado em cada ficheiro.

Se a opção ***–f*** é usada, o *file* examina os ficheiros cujos nomes aparecem em linhas separadas no ficheiro *nome_ficheiro*.

1.2.3 VER O CONTEÚDO DE UM FICHEIRO

O comando *cat* (abreviatura de *concatenate and print*) recolhe um mais nomes de ficheiros da linha de comando, lê o seu conteúdo e apresenta-o no ecrã, um a seguir ao outro. A sintaxe do comando é a seguinte:

```
cat [-s] [-v [-t] [-e]] ficheiro ...
```

O comando *cat* "despeja" tudo para o ecrã, sem qualquer pausa ou separação entre os conteúdos dos diversos ficheiros.

A opção **–s** suprime quaisquer mensagens de erro ou aviso que possam ser geradas pelo comando.

O comando *cat* é usado para apresentar o conteúdo de ficheiros de texto (ASCII). Se, no entanto, houver caracteres de controlo invisíveis, eles podem ser vistos usando a opção **–v**, que faz com que esses caracteres sejam precedidos do acento circunflexo (^). Este modo "visual" pode ser usado com mais duas outras opções. A opção **–t** permite ver os caracteres tab com Control-I (^I). A opção **–e** podemos ver a posição de fim de linha (end of line) com o cifrão ($).

Embora seja útil em muitas situações, não é o comando que é normalmente usado para ver o conteúdo de um ficheiro.

1.2.4 VER O CONTEÚDO DE UM FICHEIRO PÁGINA A PÁGINA

Um comando que é mais útil para ver o conteúdo de um ficheiro é o *pg*. Permite verificar o conteúdo do ficheiro, um ecrã de cada vez. A forma mais simples do comando é

```
pg ficheiro ...
```

Deste modo o *pg* permite ver o ficheiro 23 linhas (1 ecrã) de cada vez. Depois de apresentar as linhas, o *pg* fica à espera de um comando por parte do utilizador indicando o que fazer a seguir. Tal é assinalado pela apresentação do prompt ":". Podemos abandonar a visualização do ficheiro pressionando a letra q seguido

de <Enter>. Para continuar para a próxima página, pressionamos a tecla <Enter>. Quando é atingido o fim do ficheiro, o *pg* apresenta "(EOF):".

Outro comando útil é o *more*. Este comando apresenta basicamente as mesmas características do comando *pg*. A principal diferença entre eles reside no formato do prompt "--More--" em vez de ":" e no facto do *more* não esperar pelo <Enter> para introduzirmos um comando, simples carregamos na tecla correspondente ao comando. Por exemplo, h para help, q para quit, etc.

A forma mais simples do comando é

```
more ficheiro ...
```

Para avançar para a página seguinte pressionamos a barra de espaços e <Enter> para ver linha a linha.

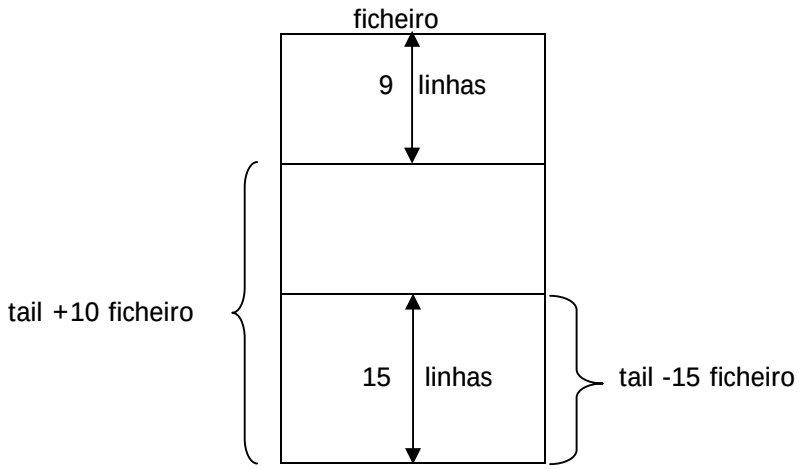
Como para qualquer comando apresentado neste texto, podemos encontrar toda a informação consultando o manual através do comando *man comando*.0

1.2.5 VER A PARTE FINAL DUM FICHEIRO

Outro comando útil para ver o conteúdo de um ficheiro é o *tail*. Por defeito, o *tail* apresenta as últimas 10 linhas do ficheiro. A sintaxe do comando é:

```
tail [+|-[número] [lbc]] ficheiro
```

Por defeito, o *tail* executa o equivalente a *tail -10l*. O número fornecido com parâmetro indica o número de linha que queremos ver. Se usarmos o sinal "-", as linhas são contadas do fim para o princípio, para determinar o ponto a partir do qual começa a apresentar as linhas. Se usarmos o sinal "+", as linhas são contadas do início para o fim.



A seguir ao número indicamos l para linhas, b para blocos (1024 bytes) e c para caracteres (ou bytes). Se omitirmos a letra, o Unix assume que pretendemos linhas.

1.2.6 VER A PARTE INICIAL DUM FICHEIRO

O comando *head* apresenta características semelhantes ao *tail*. A sintaxe é a seguinte:

```
head [-número] ficheiro ...
```

Por defeito, o comando *head* apresenta as 10 primeiras linhas. O número passado como parâmetro permite-nos indicar o número de linhas que queremos ver. Por exemplo, para vermos as primeiras 6 linhas do ficheiro:

```
$ head -6 ficheiro
```

1.2.7 COPIAR UM FICHEIRO

O Unix permite-nos efectuar cópias de ficheiros com o comando *cp* usando a sintaxe:

Cópia dum ficheiro para outro:	**`cp fich_orig fich_dest`**
Cópia de ficheiro para directório:	**`cp fich_orig dir_dest`**
Cópia de vários ficheiros para directório:	**`cp fich_orig1 fich_orig2 ... dir_dest`**

Quando se copia um ficheiro para outro, o primeiro ficheiro é lido e copiado para o segundo ficheiro. Se o segundo ficheiro não existir, é criado. Se existir, é substituído. Como na maior parte dos comandos Unix, *cp* não pede confirmação antes de efectuar a cópia.

Quando se copia um ou mais ficheiros para um directório, cada ficheiro é lido individualmente e copiado para um ficheiro com o mesmo nome no directório indicado. O directório tem de existir. Se o ficheiro não existir nesse directório, é criado. Se existir é substituído.

1.2.8 MOVER OU RENOMEAR FICHEIRO

O Unix permite mover um ficheiro ou alterar o seu nome através do comando *mv*.

Mover ou alterar nome dum ficheiro para outro:	**`mv [-f] fich_orig fich_dest`**
Mover ficheiro para directório:	**`mv [-f] fich_orig dir_dest`**
Mover vários ficheiros para directório:	**`mv [-f] fich_orig1 fich_orig2 ... dir_dest`**

Quando se "move" um ficheiro para outro, o primeiro ficheiro é lido e copiado para o segundo. O primeiro ficheiro é removido. Se o segundo ficheiro não existir, é criado. Se existir, é substituído. Como na maior parte dos comandos Unix, *mv* não pede confirmação.

Quando se "move" um ou mais ficheiros para um directório, cada ficheiro é lido individualmente e copiado para um ficheiro com o mesmo nome no directório indicado. Os ficheiros originais são removidos. O directório tem de existir. Se o ficheiro não existir nesse directório, é criado. Se existir é substituído.

Se se tentar mover um ficheiro para outro ficheiro ou directório que nos pertence mas para o qual não temos permissão para alterar, o *mv* pede confirmação para completar a operação. A opção **`-f`** elimina estas mensagens de aviso e confirmação bem como quaisquer outras mensagens de erro ou aviso que possam ocorrer durante uma operação de "move".

1.2.9 CRIAR OU ALTERAR NOME ALTERNATIVO PARA FICHEIRO

O Unix permite criar nomes alternativos (*aliases*) para ficheiros através do comando *ln* . A terminologia Unix para um nome alternativo é *link*.

"Linkar" ou criar nome alternativo para ficheiro:	**`ln [-f] fich_orig fich_dest0`**
"Linkar" ficheiro para directório:	**`ln [-f] fich_orig dir_dest`**
"Linkar" vários ficheiros para directório:	**`ln [-f] fich_orig1 fich_orig2 ... dir_dest`**

Quando se cria um link para um ficheiro já existente, uma nova entrada no directório é criada que se refere aos meus dados do ficheiro original. Se o segundo nome de ficheiro não existir, é criado. Se existir, é substituído.

Quando se cria um link de um ou mais ficheiros para um directório, cada ficheiro é lido individualmente e "linkado" para um ficheiro com o mesmo nome no directório indicado. O directório tem de existir. Se o ficheiro não existir nesse directório, é criado. Se existir é substituído.

Se se tentar "linkar" um ficheiro para outro ficheiro ou directório que nos pertence mas para o qual não temos permissão para alterar, o *ln* pede confirmação para completar a operação. A opção **`-f`** elimina estas mensagens de aviso e confirmação bem como quaisquer outras mensagens de erro ou aviso que possam ocorrer durante uma operação de "move".

Existem algumas limitações na utilização do comando *ln*. O Unix não permite links entre *file systems*. Em determinadas versões de Unix, esta limitação pode ser ultrapassada pela utilização de links simbólicos.

1.2.10 COMPARAÇÃO ENTRE cp, mv E ln

Da explicação anterior deu para perceber que os três comandos são parecidos. Têm a mesma sintaxe.

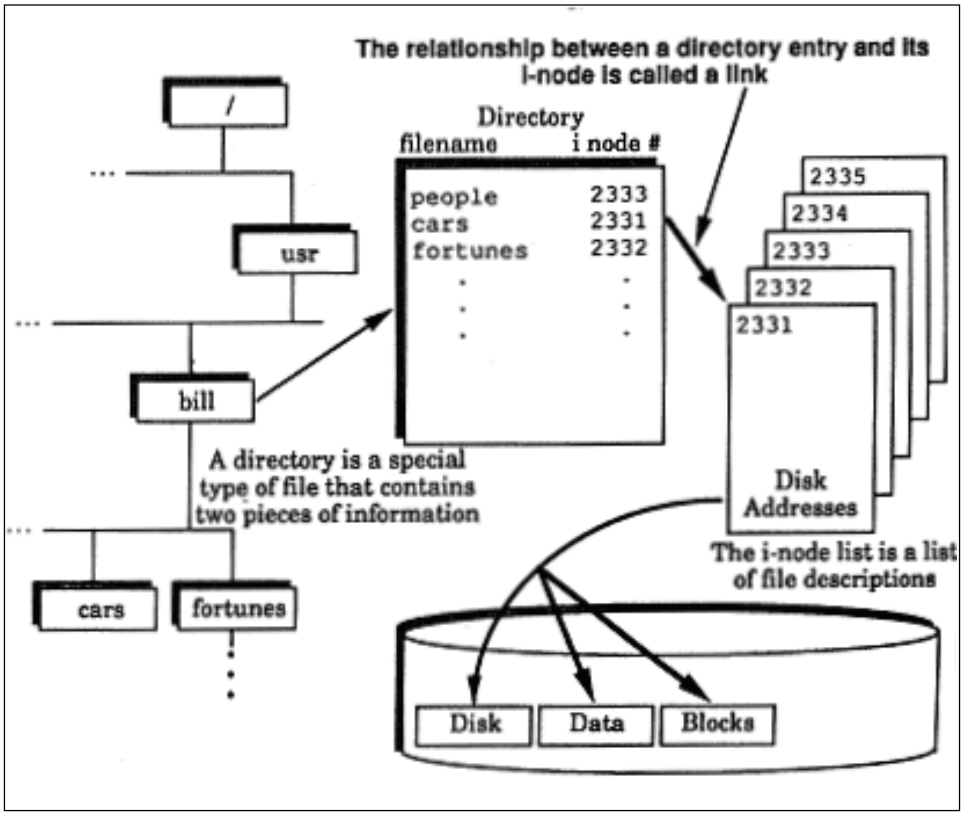
Para perceber as diferenças entre eles, é preciso saber um pouco mais acerca do modo como o Unix armazena os ficheiros no sistema de ficheiros.

Um directório é uma espécie de ficheiro que contém duas pequenas informações acerca dos ficheiros que estão "dentro". Essas informações são o nome do ficheiro, claro, e o seu número de *i-node*.

O número de i-node é um índice duma lista de i-nodes (números internos). Existe apenas um, e um só, i-node para um ficheiro no file system. Não é o nome do ficheiro num directório que define o ficheiro. É a existência de um i-node que define os atributos dum fiheiro e o local do disco onde está guardado. O i-node contém o endereço do blocos de disco que contêm os dados do ficheiro. Ver figura da página seguinte.

As diferenças entre os três comandos explicam-se da seguinte maneira:

- cp — Cria uma nova entrada no directório, destina-lhe um novo i-node e depois copia os dados dum ficheiros para o outro
- mv — Na sua forma mais simples, cria numa entrada no directório para o novo ficheiro usando o mesmo i-node do ficheiro antigo. O nome antigo é retirado do directório.
- ln — Cria uma entrada no directório para o novo ficheiro, usando o mesmo i-node do ficheiro antigo. O ficheiro antigo mantém-se de tal forma que existem duas entrada com nomes diferentes no directório que respondem ao mesmo i-node e aos meus dados.



1.2.11 ELIMINAR FICHEIROS

O comando *rm* permite remover, ou eliminar, um ficheiro. A sintaxe do comando é a seguinte:

```
rm [ -fir ] ficheiro ...
```

Cada ficheiro indicado no comando é removido do sistema de ficheiros. Mais uma vez, o Unix assume que sabemos o que estamos a fazer pelo que não efectua qualquer confirmação antes de proceder à execução do comando.

Algumas opções podem ser usadas para modificar a execução do comando *rm*:

-f Esta opção, de maneira semelhante ao que acontece nos comandos *mv* e *ln*, força a execuação do comando, isto é, neste caso, a eliminação do ficheiro. Nenhumas perguntas são feitas, nem são apresentados quaisquer avisos ou mensagens de erro.

-i Esta opção é a opção de interactividade. Informa o *rm* para interagir com o utilizador durante a operação de eliminação. Antes do ficheiro ser efectivamente eliminado, o *rm* apresenta o nome do ficheiro seguido de um "?". Se se pretende eliminar, pressiona-se o "y" seguindo de "Enter". Qualquer outra resposta, "salta" a eliminação do ficheiro.

-r Esta opção permite ultrapassar a limitação do *rmdir* na eliminação de directórios não vazios. Com esta opção, o *rm* elimina qualquer directório indicado bem como os ficheiros neles contidos. O *rm* desce recursivamente ao longo da estrutura de directórios começando no directório indicado e elimina todos os ficheiros e directórios árvore acima, incluindo o directório inicial.

Devido ao facto do Unix ser um sistema multiutilizador com utilizadores a criar e a eliminar ficheiros constantemente, é praticamente impossível recuperar um ficheiro eliminado. Logo que um ficheiro é

eliminado, o sistema de ficheiros verifica que existe algum espaço livre que pode usar. No próximo pedido de espaço que alguém faça, o sistema de ficheiros provavelmente usará o espaço acabado de ser libertado.

A única possibilidade de recuperação de um ficheiro eliminado é obter a cópia a partir de uma cópia de segurança efectuada recentemente.

Cuidado, pois, com o comando *rm*. Com apenas 5 teclas podemos colocar o nosso sistema de ficheiros Unix em apuros. O comando `rm *` remove tudo no directório corrente, e não nos será pedida qualquer confirmação.

Pior do que isto, se estivermos *logged in* como administradores do sistema, nunca deveremos experimentar o comando `rm –r /` . Se removermos tudo a partir da raiz, perderemos o nosso sistema de ficheiros por inteiro.

Como se pode imaginar, o comando *rm* não remove efectivamente os dados. Remove uma entrada (um *link*) na tabela de ficheiros representativa do sistema de ficheiros (conhecida com FAT *File Allocation Table*). Quando o sistema de ficheiros verifica que não há mais *links* para o *i-node* em particular, liberta esse *i-node* e os blocos de disco a ele alocados para poder ser usado por outro ficheiro.

1.2.12 UTILIZANDO *PATHNAMES*

Qualquer comando que opere com ficheiros no sistema de ficheiros Unix pode especificar esse ficheiro através do seu *full pathname* (caminho completo), *relative pathname* (caminho relativo) ou *simple pathname* (caminho simples).

O tipo de representação do ficheiro fica a nosso cargo. Usamos aquele que mais nos convier em função da posição relativa do directório corrente e do directório onde se encontra o ficheiro em causa.

Não é necessário efectuar um *cd* para o directório do ficheiro só para usar o caminho simples. Podemos especificar o caminho completo ou o caminho relativo ao directório corrente.

1.3 QUESTÕES

1. Como é que o sistema de ficheiros (*file system*) de um sistema UNIX se encontra estruturado ?
2. Para que serve um directório ?
3. Diga o nome de três directórios existentes num *file system* UNIX, e refira-se ao seu conteúdo.
4. O que é um directório *home* ? E o directório corrente ?
5. Qual é o comando que pode utilizar para ver os nomes dos ficheiros em UNIX.
6. Descreva o comportamento de três das suas opções.
7. Explique para que serve o comando *pwd* ?
8. Qual é a diferença entre um *pathname* completo e um *pathname* relativo ?
9. Como se muda de directório num sistema UNIX ?
10. Qual é o comando UNIX usado para criar directórios ?
11. Para que serve o comando *rmdir* ?

12. O que é que o comando *file* faz ?

13. Descreva as diferenças entre os comandos *cat*, *more*, *tail* e *head*.

14. Qual é o comando para copiar um ficheiro ? E para mudar o seu nome ?

15. O que é um *link* ? Como se cria um *link* ?

16. Como se apaga um ficheiro ?

17. Crie três directórios chamados DOCUMENTS, PROGRAMS, e MAIL.

18. Dentro do directório PROGRAMS, crie mais dois directórios chamados DATABASE e C_LANGUAGE.

19. Faça com que o directório C_LANGUAGE se torne o seu directório corrente, e determine o seu pathname completo.

20. Faça com que o seu directório *home* se torne o seu directório corrente, e determine o seu *pathname* completo.

21. Copie o ficheiro fortunes.txt que se encontra no directório /users/1/bach/so1 para o directório DOCUMENTS.

22. Crie uma cópia do ficheiros cars.txt no directório PROGRAMS.

23. Mova o ficheiro cars.txt do directório PROGRAMS para o directório DOCUMENTS.

24. Mude o nome do ficheiro cars.txt no directório DOCUMENTS para old_cars.

25. Crie no directório DATABASE, um link para o ficheiro old_cars chamado cars.

26. Diga qual é o *pathname* completo do ficheiro old_cars. Diga também qual é o seu *pathname* relativamente ao seu directório *home*, ao pai do seu directório *home*, ao directório DOCUMENTS e ao directório PROGRAMS.

27. Liste todos os ficheiros e directórios que estão dentro do seu directório *home*.

28. Remova o directório MAIL.

29. Remova o ficheiro old_cars.

30. Remova o directório PROGRAMS.

31. Determine que tipo de dados contém o ficheiro /etc/group.

32. Mostre no écran as ultimas 10 linhas do ficheiro /etc/passwd.