



**FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA**

Departamento de Engenharia Electrotécnica

PROGRAMAÇÃO DE MICROPROCESSADORES

2012 / 2013

Mestrado Integrado em Engenharia Electrotécnica
e Computadores

1º Ano

1º Semestre

Trabalho nº 1

Introdução ao Linux – Kit de Sobrevivência

Versão: 1.0

Paulo da Fonseca Pinto

Luis Bernardo

Índice

| | | |
|-----|--|----|
| 1 | Objectivos..... | 3 |
| 2 | Introdução..... | 3 |
| 3 | Antes do início: Alguns conceitos úteis | 3 |
| 4 | O início de tudo | 6 |
| 4.1 | Carregar o kernel e instalar <i>device drivers</i> | 6 |
| 4.2 | Arrancar com o kernel e um conjunto de serviços | 6 |
| 4.3 | Arrancar com terminais | 7 |
| 4.4 | O início de uma sessão de utilizador | 7 |
| 5 | Uma visita guiada pelo sistema de ficheiros em Cygwin..... | 8 |
| 6 | Comandos e Conceitos Básicos..... | 9 |
| 6.1 | Páginas de Manual..... | 11 |
| 6.2 | Comandos sobre Directorias..... | 12 |
| 6.3 | Comandos sobre Ficheiros..... | 13 |
| 6.4 | Outros comandos | 18 |
| 7 | O meu primeiro programa em C | 18 |
| 7.1 | Uso da <i>Memory Stick</i> | 18 |
| 7.2 | O programa para saudar o Mundo | 18 |

Cada vez que este documento tiver uma modificação, a identificação da versão muda. O critério de mudança é o seguinte:

Versão x.y

Em que

- | | |
|---|--|
| x | o primeiro dígito é incrementado quando existirem mudanças de substância, i.e. melhoramentos técnicos, correcções, actualizações, etc. |
| y | o segundo dígito é incrementado quando apenas mudanças editoriais forem incorporadas no documento. |

1 Objectivos

O objectivo deste trabalho é explicar o Linux, e o uso da aplicação Cygwin, em poucas palavras.

A estrutura do documento é a seguinte: Tem uma explicação inicial do Linux para que uma pessoa com conhecimentos mínimos de uso de sistemas de exploração possa compreender o que está a acontecer no Linux. Depois, contém uma sequência de explicações de comandos de uso prático em que se usa o Linux sobre um sistema de exploração da Microsoft (XP, Vista, etc.). Esta segunda parte foi escrita para ser realizada na primeira aula de Laboratório da disciplina de Programação de Microprocessadores.

Este documento vai ter evoluções ao longo do tempo consoante a disponibilidade de tempo dos seus autores. Não será necessariamente no início dos semestres. Assim, **existe um identificador da versão na página inicial.**

2 Introdução

O sistema operativo Linux começa a ter uma grande popularidade fruto de diversas razões. Entre elas está o facto de não existirem direitos de propriedade, ser muito menos atacável por vírus e ter o seu código aberto. Só estas três razões já são suficientes para um futuro engenheiro considerar francamente o seu uso.

A utilização directa do Linux por alunos do 1º ano pode ser um pouco traumatizante, pois um computador menos comum pode exigir um esforço muito considerável na instalação de *device drivers*. Por exemplo, problemas destes podem ser comuns: “a minha placa gráfica ainda não tem o *device driver* no software que descarreguei. Como devo proceder?” Assim, existem duas alternativas:

1. O uso de máquinas virtuais, em que o sistema Linux fica completamente instalado de um modo muito fácil pois todos os *device drivers* são do sistema anfitrião. No entanto, as máquinas virtuais exigem computadores relativamente potentes; ou
2. Aplicações como a Cygwin que criam um ambiente muito parecido com o Linux numa janela de Windows. Em situações de desespero, o aluno pode sempre fazer o que quer (apagar um ficheiro, copiar um ficheiro, etc.) no ambiente Windows. O Cygwin tem ainda a vantagem de ser quase um sistema completo de Linux em que existem pacotes que podem ir sendo instalados (ou retirados) ao longo do tempo. Assim, à medida que o aluno vai percebendo mais de Linux vai acrescentado o seu Cygwin. Chegará o momento em que usará um sistema Linux completo (máquina virtual ou residente).

3 Antes do início: Alguns conceitos úteis

O Unix (de onde o Linux originou) trouxe uma característica muito interessante: **SIMPLICIDADE.**

Havemos de ver isso ao longo desta explicação depois de passado o pânico de se estar a utilizar um sistema novo e desconhecido.

Os sistemas Windows têm uma forma conhecida de identificar os dispositivos de armazenamento e as partições neles usadas:

- A: floppy disk
- B: floppy disk
- C: hard disk
- D: hard disk
- E: Memory Stick

Etc.etc.etc.

Por outro lado, outros dispositivos têm uma forma diferente de serem tratados. Por exemplo, o rato, a ligação de rede, etc.

No Linux é simples: tudo é tratado do mesmo modo – são apenas dispositivos, ou *devices*. E os programas para os gerir e comunicar com eles, os chamados *device drivers*, estão todos numa pasta chamada **/dev**. Por exemplo, o ficheiro **/dev/mouse** contém o programa para controlar o rato. Para os dispositivos de armazenamento existe um nome para o dispositivo e depois cada dispositivo pode ter várias partições. No Windows as coisas são um bocadinho mais confusas pois não se sabe à partida se duas partições estão no mesmo dispositivo ou não. Por exemplo as partições C: e D: podem estar no mesmo disco físico, ou não). No Linux podemos ter os seguintes nomes para os dispositivos. Ao longo do tempo, e dependendo das várias distribuições de Linux, estes nomes podem variar ligeiramente.

| Device | Nome | Windows |
|--|--|---------|
| Primeiro floppy | /dev/fd0 | A: |
| Segundo floppy | /dev/fd1 | B: |
| Primeiro disco (disco todo) | /dev/hda | |
| Primeiro disco, partição 1 | /dev/hda1 | C: |
| Primeiro disco, partição 2 | /dev/hda2 | D: |
| Primeiro disco, partição 3 | /dev/hda3 | E: |
| | | |
| Segundo disco (disco todo) | /dev/hdb | |
| Segundo disco, partição 1 | /dev/hdb1 | |
| Segundo disco, partição 2 | /dev/hdb2 | |
| Segundo disco, partição 3 | /dev/hdb3 | |
| | | |
| Primeiro disco SCSI (disco todo) | /dev/sda | |
| Primeiro disco, SCSI partição 1 | /dev/sda1 | |
| Primeiro disco, SCSI partição 2 | /dev/sda2 | |
| Primeiro disco, SCSI partição 3 | /dev/sda3 | |
| | | |
| Segundo disco SCSI (disco todo) | /dev/sdb | |
| Segundo disco, SCSI partição 1 | /dev/sdb1 | |
| Segundo disco, SCSI partição 2 | /dev/sdb2 | |
| Segundo disco, SCSI partição 3 | /dev/sdb3 | |
| | | |
| Primeiro disco SCSI CD-ROM | /dev/scd0 | |
| Segundo disco SCSI CD-ROM | /dev/scd1 | |
| | | |
| Primeiro dispositivo SCSI genérico (do tipo scanners, impressoras, etc.) | /dev/sga | |
| Segundo dispositivo SCSI genérico | /dev/sgb | |
| | Podem ser números /dev/sg0 /dev/sg1 | |

As partições nos discos podem-se criar com um comando chamado **fdisk**, ou com outras aplicações com interface gráfica, como o **gparted**. As partições vão servir para colocar ficheiros. Existe um outro uso que é utilizar uma partição como memória RAM virtual (e aumentar o desempenho da máquina). Chama-se a isso usar essa partição como **swap**. O comando **mkswap** permite atribuir este uso a uma partição que é identificada no comando.

Antes de se poder usar uma partição para guardar ficheiros tem de se criar um sistema de ficheiros. Esta operação é semelhante à de formatar uma partição no Windows. Existem muitos tipos de sistemas de ficheiros para o Linux que diferem no comprimento máximo de um nome para um ficheiro, comprimento máximo dos ficheiros, etc., etc. O tipo mais usado é o Third **Extended Filesystem**, ou *ext3fs*. O comando para criar este sistema de ficheiros é *mke2fs*.

Pode parecer uma lenga-lenga, mas antes de se poder usar um sistema de ficheiros, ele tem de ser “montado” no sistema. Os nomes das várias pastas no sistema Linux começam com uma barra / e depois tem sempre sequências de nomes com barras. Por exemplo

```
/home/Paulo/docs/disciplinas/pm/laboratório/enunciados/Trabalho_1.doc
```

Esta árvore imensa começa em /. Ao contrário do Windows, existe uma raiz única. As partições podem ter os seus sistemas de ficheiros, mas onde é que se vão “colar” nesta árvore imensa do Linux? É para isso que serve o *mount*.

Esta operação associa um certo sistema de ficheiros que foi feito numa certa partição a um sítio específico da árvore. Por exemplo, a raiz do sistema de ficheiros que pode estar na primeira partição do primeiro disco e usar um certo sistema de ficheiros é montada em /; um outro sistema de ficheiros que contém o nome de raiz dele de */usr* e está numa certa partição deve ser colocado em */usr* na árvore. Assim todos os ficheiros que estão lá vão ser acedidos a partir do nome */usr/...* como se quer

Existe um ficheiro que pode ser editado pelo utilizador e que contém a descrição dos sistemas de ficheiros (pode-se ter mais do que um ao mesmo tempo) e onde estão montados. Aliás, este ficheiro serve mesmo para dar as ordens de montagem quando o sistema arranca. Chama-se */etc/fstab*. Lá estão também as partições usadas para swap. Um exemplo é o seguinte:

```
#/etc/fstab
#device          directory      type          options
#
/dev/hda1         /              ext3          defaults
/dev/hdb2         /home          ext3          defaults
/dev/hdb1         none           swap          sw
/proc             /proc          proc          defaults
```

As linhas começadas por “#” são de comentário. Pode-se ver aqui que a raiz do sistema de ficheiros está na partição 1 do primeiro disco, é do tipo *ext3fs* e as opções são as por defeito. Como vamos ver a seguir a montagem da raiz do sistema é feita automaticamente no arranque do sistema. Os outros não. O sistema de ficheiros */home* é também do tipo *ext3fs* e as opções de montagem são as por defeito, e os seus ficheiros estão na segunda partição do disco 2. A primeira partição do disco 2 tem o espaço de swap, e finalmente, */proc* é um sistema de ficheiros virtual cujo significado não se descreve neste enunciado.

No caso do Cygwin, o sistema de ficheiros raiz “/” está montado em *c:|cygwin*. Para o Linux é como se o “mundo todo” estivesse aí.

O kernel (núcleo) é o âmago do sistema operativo Linux. Controla a interface entre os nossos programas e os dispositivos de hardware, faz a gestão que permite que muitos programas corram ao mesmo tempo no computador, e muitas outras tarefas.

O kernel não é uma entidade independente. É como se fosse uma quantidade de código que os nossos programas mandam correr quando chamam “**rotinas de sistema**” (*system calls*) como por exemplo rotinas para ler do teclado ou escrever no ecrã.

O kernel do Linux é considerado monolítico no sentido em que inclui todos os *device drivers*. No entanto, existe também a possibilidade hoje em dia de se terem *device drivers* que são carregados e descarregados em memória através de comandos do utilizador.

4 O início de tudo

4.1 CARREGAR O KERNEL E INSTALAR *DEVICE DRIVERS*

Quando se liga o computador uma imagem do kernel é carregada em memória. Normalmente é carregada uma pequena porção que não está comprimida com o *gzip* que por sua vez descomprime o resto do kernel.

Existe um conjunto de parâmetros que estão guardados no kernel. Exemplos são o nome do dispositivo que vai ser usado para montar a raiz do sistema de ficheiros, o modo de texto usado para a consola (se vai ter acentos, ou não, por exemplo). Este parâmetros podem ser modificados usando o comando *rdev*, ou directamente na linha de comando que arranca o kernel.

À medida que o kernel se vai carregando em memória são escritas mensagens na consola (e são também guardadas num ficheiro de log em */var/log/messages*).

Estas mensagens vão dando indicação da inicialização dos vários *device drivers*. Um exemplo de um início de um kernel baseado nas mensagens escritas (fruto das acções efectuadas) é o seguinte:

Primeiro o kernel indica que *font* (tipo de letras) escolheu para a consola e que tipo de consola detectou. Depois reúne informação sobre o bus PCI e verifica se existem placas PCI no sistema. Seguidamente calcula a velocidade do processador. Este cálculo é pouco preciso e tem o nome de *BogoMIPS*. É necessário calculá-lo para controlar os tempos de espera pelos *device drivers*. Escreve depois a informação sobre o tamanho da memória e quanta dessa memória fica reservada para ele. Inicializa seguidamente a parte de rede e verifica o tipo de CPU da máquina. Escreve depois o número da versão do kernel, quem a compilou, em que máquina e com que compilador. A seguir começa a inicializar a porta série (*/dev/tty00*, ou *COM1*). Mostra o seu endereço, linha de interrupção, etc. Inicializa os dispositivos SCSI e detecta o uso de swap, dizendo o tamanho que vai utilizar. A seguir inicializa os portos paralelo, a placa de rede, o floppy disk, etc., etc.

4.2 ARRANCAR COM O KERNEL E UM CONJUNTO DE SERVIÇOS

Quando finaliza a inicialização de todos os *device drivers* o kernel corre o programa *init*. Dependendo de qual a distribuição do sistema Linux este programa pode estar em */etc*, */bin*, ou */sbin*. No caso do Cygwin, como não se está perante um verdadeiro sistema Linux, não existe o programa *init*. O *init* é um programa genérico que arranca com programas e recomeça com certos programas quando eles acabam. Por exemplo, cada consola tem um processo *getty* que a controla que é começado pelo *init*.

Tudo o que o programa *init* faz é controlado a partir do ficheiro */etc/inittab*. Os vários serviços do Linux que são iniciados estão organizados em níveis, designados vulgarmente por *runlevels*. Com isso consegue-se vários modos de arranque do sistema. Por exemplo, no Linux Fedora existem os seguintes níveis:

1 - Single user mode (só com consola)

- # 2 - Multiuser, sem NFS (O mesmo que o 3 se não se usar rede)
- # 3 - Full multiuser mode (só com consola, mas com vários utilizadores)
- # 4 – unused
- # 5 - X11 (ambiente gráfico, vários utilizadores)

Os vários serviços são iniciados não a partir do ficheiro */etc/inittab* mas de ficheiros que têm no nome as letras *rc* (*resource configuration*). Antigamente era o próprio */etc/inittab* que corria o ficheiro de arranque do kernel (chamado de */etc/rc.d/rc.sysinit*) e o ficheiro *rc* respectivo do *runlevel* que se queria. Hoje em dia o ficheiro */etc/inittab* especifica apenas qual é o valor do *runlevel*. Os mesmos ficheiros que eram chamados são corridos automaticamente.

Os serviços que realmente podem ser iniciados num *runlevel* podem variar ligeiramente ao gosto do administrador do sistema. Para poder configurar isso (quais os serviços que realmente quer iniciar) usa-se o comando *chkconfig*. Os serviços que se estão a referir são, por exemplo, email, servidores de Web, rede, etc., etc. Os nomes podem variar ligeiramente consoante a distribuição de Linux. Para a descrição seguinte usou-se a distribuição Fedora.

Assim, após correr o arranque do kernel através dos comandos especificados no ficheiro */etc/rc.d/rc.sysinit*, o sistema vai executar o ficheiro de arranque associado ao *runlevel* definido */etc/rc.d/rc0.d*; */etc/rc.d/rc1.d*; */etc/rc.d/rc2.d*; */etc/rc.d/rc3.d*; */etc/rc.d/rc4.d*; etc. Associado a cada serviço há um ficheiro nestas directorias com um nome começado por K ou S (para *Kill* ou *Start*), seguido do número de serviço e do nome do serviço (e.g. K15httpd), com contém as instruções de paragem ou arranque do serviço. No início, são executados os S, claro.

Claro que se pode arrancar (ou parar) um serviço especificamente, sem estar preso a um certo *runlevel* nem ao momento inicial de arranque do sistema. Por exemplo, um sistema Linux arrancou num certo *runlevel*, que não incluía a servidor apache (Web) mas o utilizador queria arrancar com o servidor. Assim, podia fazer depois do arranque (o nome do servidor apache é *httpd*)

```
/etc/init.d/httpd start
```

A seguir a correr o ficheiro *rcN.d* do *runlevel N*, é corrido outro ficheiro importante no Linux – o */etc/rc.d/rc.local*. São lá colocados outros comandos que devem ser corridos sempre no arranque do sistema mas que não sabemos exactamente onde colocá-los tendo em conta as hipóteses anteriores.

O arranque do sistema está quase no final!

4.3 ARRANCAR COM TERMINAIS

As últimas acções são o arranque de programas que vão controlar as consolas. Têm o nome */sbin/agetty*. Existem muitas variantes de *agetty*. Basicamente, são eles que vão reagir quando o utilizador carrega no botão do rato dentro da consola, ou escrever algum carácter, e lançam o */bin/login* que vai tratar do login do utilizador. A lista de utilizadores existente no sistema está guardada no ficheiro */etc/passwd*.

4.4 O INÍCIO DE UMA SESSÃO DE UTILIZADOR

Tal como quando o sistema arranca, quando um utilizador inicia sessão são também corridos comandos de inicialização. Basicamente, são comandos de configuração para que o ambiente seja o que o utilizador mais goste.

O nome dos ficheiros que contêm estes comandos começam com um ponto “.”. É assim pois não aparecem quando se dá o comando de listar todos os ficheiros da directoria. É um pouco como os ficheiros ocultos no Windows...

O método de configuração é o seguinte: primeiro são corridos ficheiros de configuração gerais (que estão colocados em */etc*, por exemplo) e depois são corridos os nossos. À medida que for usando um sistema Linux vai ver que podem existir ficheiros de configuração de editores de texto, de sistemas de janelas, de gestor de sistemas de janelas, etc., etc.

Nós vamos apenas ver o de configuração do gestor de comandos de linha da consola. Existem dois gestores de comandos de linha mais populares: o *bash*, e o *tcsh*. O Cygwin usa o primeiro e os ficheiros de configuração são os *.bashrc* e *.bash_profile*.

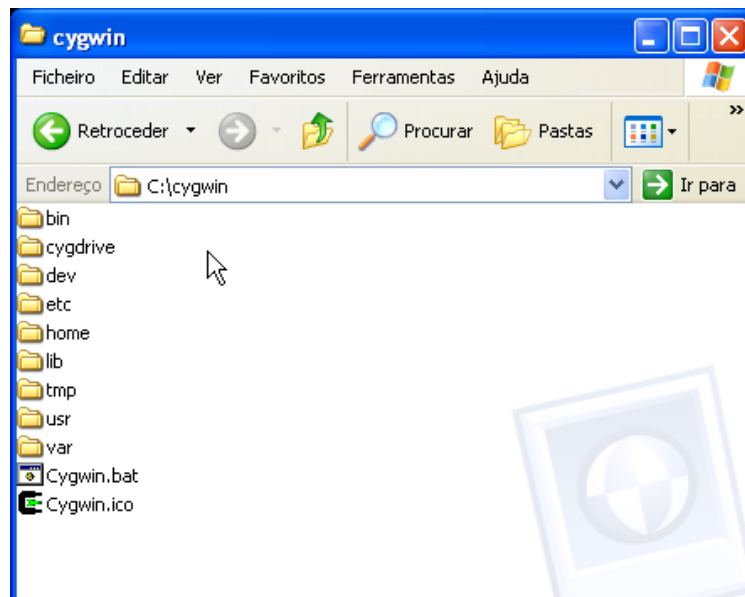
O comando para correr este tipo de ficheiros é o *source*.

Com o tempo poderá achar útil acrescentar linhas para configurar o ambiente de trabalho. No caso de Programação de Microprocessadores, como os computadores do laboratório vão ser usados por muitos alunos, não haverá muitas hipóteses de alterar estes ficheiros. No entanto, à boa moda do Linux, pode “trazer” um ficheiro deste tipo na sua Memory Stick e correr o ficheiro no início da sessão.

Abra estes dois ficheiros com o editor Crimson e tente perceber o que lá está escrito

5 Uma visita guiada pelo sistema de ficheiros em Cygwin

A estrutura de ficheiros dos sistemas Linux também varia ligeiramente de distribuição para distribuição. De qualquer modo muita coisa se mantém e é sobre esta estrutura que esta secção se vai debruçar. A figura abaixo mostra as pastas logo na raiz “/” do sistema de ficheiros para o Cygwin. Como não foram instalados muitos pacotes, faltam centenas de ficheiros habitualmente existentes no Linux.



Vai-se fazer uma pequena descrição de cada uma destas pastas. O propósito é apresentá-las aos alunos e também aproveitar para descrever certas características importantes do Linux.

/bin – Nesta directoria são guardados os comandos do Linux. Por exemplo, o comando para copiar ficheiros tem o nome de cp. No Linux os comandos são ficheiros com o código do

próprio comando. A maior parte deles foi escrita em C, compilado, e o código executável colocado em **/bin**. Entre nessa directoria e veja as dezenas de comandos que lá estão.

/dev – Nesta directoria são guardados os ficheiros dos dispositivos. O Linux tem uma característica muito interessante – todos os dispositivos são como ficheiros. Por exemplo, a rede é um ficheiro. Quando queremos enviar dados pela rede escrevemo-los no ficheiro. Quando queremos receber dados da rede lemos o ficheiro. O mesmo acontece com os terminal. No caso do nosso Cygwin (com os pacotes que instalámos) existem apenas 4 dispositivos: o **fd** para o disco; o **stdin** que significa standard input que é o teclado; o **stdout** que é o ecrã para escrever; e o **stderr** que é outro ecrã que serviria para escrever apenas as mensagens de erro. Normalmente o **stderr** está redireccionado para o **stdout**.

/etc – Esta directoria, como o nome sugere tem coisas que não “cabiam” noutras directorias. Tem por exemplo, uma directoria chamada de **skel** (esqueleto) que contém ficheiros tipo para serem copiados para uma área de utilizador quando ela for criada, e quase todos os ficheiros de configuração dos serviços do sistema operativo.

/home – Esta directoria contém como pastas os utilizadores que estão definidos no sistema. No caso do Cygwin ela irá conter tantos utilizadores quantos os que existirem no sistema Windows na altura de instalação do Cygwin.

/lib – O nome lib significa *library*, que é biblioteca. São aqui guardados ficheiros com código ou executáveis dos vários programas que vamos carregando no Linux. Num sistema completo existem muitos ficheiros aqui. No nosso caso, não são assim tantos.

/tmp – O nome tmp significa *temporary*, e é o local onde são criados por omissão todos ficheiros temporários no sistema.

/usr – Esta directoria tem também muito do que se descreveu para as outras. É outro sítio para colocar certos tipos de ficheiros. Por exemplo, existem as seguintes pastas: **bin**; **lib**; **local**; **sbin**; **src**; **tmp**. É por baixo de **/usr** que são colocadas as páginas de manual do Linux. Na nossa instalação de Cygwin esta directoria está quase vazia, mas é um lugar onde os sistemas actuais Linux colocam muitos ficheiros.

/var – Tal como **/etc**, também aqui se colocam ficheiros que não cabem noutros sítios.

É muito habitual no Linux começar a colocar um certo tipo de ficheiros num certo sítio e com o tempo ele vai ficando lá e toda a comunidade se vai habituando a vê-lo lá.

6 Comandos e Conceitos Básicos

Esta secção vai descrever vários comandos de linha do sistema Linux. O objectivo é ensinar os alunos a utilizar a janela de Linux e a pouco e pouco conseguir fazer todas as tarefas sem precisar das janelas do Windows. Os comandos que vão ser descritos podem-se dividir em quatro grupos. A tabela seguinte resume os comandos a abordar. Nesta versão do documento o último grupo não será coberto.

| Directorias | |
|-------------|---|
| pwd | Mostra a directoria onde se está “ <i>print working directory</i> ” |
| cd | Muda de directoria “ <i>change directory</i> ” |

| | |
|------------------|---|
| mkdir | Cria uma directoria “ <i>make directory</i> ” |
| rmdir | Apaga uma directoria “ <i>remove directory</i> ” |
| Ficheiros | |
| ls | Lista os ficheiros de uma directoria “ <i>list</i> ” |
| ls -lah | O mesmo do que o anterior em que se realça o uso de opções |
| mv | Muda um ficheiro de sítio, ou muda-lhe o nome “ <i>move</i> ” |
| cp | Copia um ficheiro “ <i>copy</i> ” |
| rm | Apaga um ficheiro “ <i>remove</i> ” |
| cat | Lista o conteúdo de um ficheiro “ <i>concatenate</i> ” |
| more | Lista o conteúdo de um ficheiro de um modo mais sofisticado |
| less | Lista o conteúdo de um ficheiro de um modo ainda mais sofisticado |
| Manual | |
| man | Mostra páginas do manual |
| Outros | |
| whoami | Mostra quem é este utilizador |
| who | Mostra todos os utilizadores que estão em sessão |
| ps | Mostra os processos que estão a correr |
| CONTROL C | Interrompe o programa activo associado ao teclado (<i>foreground</i>) |
| CONTROL Z | Pára um programa, deixando parado em <i>background</i> (sem receber eventos do teclado) |
| fg | Passa um programa parado em <i>background</i> a <i>foreground</i> |
| bg | Corre um programa parado em <i>background</i> (sem teclado) |
| kill | Pára um processo identificado pelo número de processo |
| jobs | Lista programas a correr numa consola |

Comece então por colocar o seu monitor com cinco janelas como mostra a figura em baixo. Para simplificar a explicação vão-se dar números de identificação às janelas. A janela zero é a janela do Cygwin e as outras quatro são do Windows. Coloque as janelas do Windows na directoria */home*, como está mostrado.

Na janela 1 entre na pasta do utilizador (é Paulo na figura, mas no caso do Laboratório é capaz de ser DEE). Veja que existem três ficheiros começados com ponto – são os ficheiros de configuração explicados atrás. Com o editor Crimson abra esses ficheiros e tente perceber o que eles contêm. As linhas começadas com “#” são linhas de comentários e explicam um pouco o que está a acontecer. No ficheiro *.bash_profile* existem apenas seis linhas que não estão comentadas. Consegue perceber o que elas fazem? Não se preocupe muito se nesta altura ainda não entender tudo muito bem.



6.1 PÁGINAS DE MANUAL

O sistema operativo Linux oferece ao utilizador um manual completo de programador, utilizador e administrador com todas as funções que podem ser utilizadas, desde comandos de linha a funções de biblioteca disponibilizadas pela biblioteca do C explicando os seus parâmetros e opções.

Sempre que tiver dúvidas sobre alguma coisa do Linux tente ler o manual. Nos primeiros tempos vai ser difícil perceber o tipo de texto e o modo como está escrito pois o manual assume que as pessoas sabem já bastante de Linux. A pouco e pouco vai-se habituando. Lembre-se que na sua vida profissional vai ter de aprender muito por si só.

Este manual está organizado em pastas numeradas de 1 a 9: na pasta 1 estão os comandos da linha de utilizador; na pasta 2 estão a *system calls*, na pasta 3 está a maior parte das funções da biblioteca da linguagem C, e por aí adiante (a pasta 5 tem os formatos dos ficheiros e a pasta 8 os comandos de administração de sistema, por exemplo). Pode-se aceder a este manual utilizando o comando **man**, seguido do número de pasta, e do nome do comando.

Tente ver uma página sobre um comando de linha muito simples. O comando **“pwd”** que mostra o nome da directoria onde se está.

Pode simplesmente executar o comando e veja se consegue perceber o que lá está escrito.

```
man pwd
```

Para sair carregue em **“q”** de **“quit”**.

Às vezes tem de se escrever mesmo a pasta em que queremos que o manual encontre a entrada, pois pode haver mais do que uma entrada possível (uma função de C e um comando de linha com o mesmo nome). Assim, neste caso, pode também escrever

```
man 1 pwd
```

Se quiser ver as entradas que existem para o *man* (isto é, a que palavras o *man* responde) vá numa janela de Windows a */usr/share/man/manN* em que *N* é a pasta respectiva. Não se assuste com o número de entradas!...

Já agora, as páginas do manual podem estar noutro sítio e o sistema não as consegue localizar. Para isso é preciso definir essa localização num parâmetro chamado de MANPATH. Para saber qual o seu actual valor use

```
echo $MANPATH
```

Para acrescentar uma certa directoria chamada */didi/dodo* tem de fazer¹

```
export MANPATH=/didi/dodo:$MANPATH
```

Com isto, vai ser procurado primeiro em */didi/dodo*, e depois nos outros sítios que já estavam definidos em MANPATH.

6.2 COMANDOS SOBRE DIRECTORIAS

Nos comandos sobre directorias, vamos começar mesmo com o “*pwd*” pois já sabe o que ele faz por ter lido a página do manual na secção anterior. Na janela do Cygwin faça então

```
pwd
```

Vai aparecer qualquer coisa do género

```
/home/Paulo
```

Vamos agora começar a criar qualquer coisa. Coloque as janelas 1, 2 3 e 4 na directoria base. Isto é, em todas as janelas aparecem os tais três ficheiros de configuração. Depois na janela Cygwin faça o seguinte comando para criar a directoria chamada “*primeiro*” e observe que a pasta aparece nas outras janelas do Windows.

```
mkdir primeiro
```

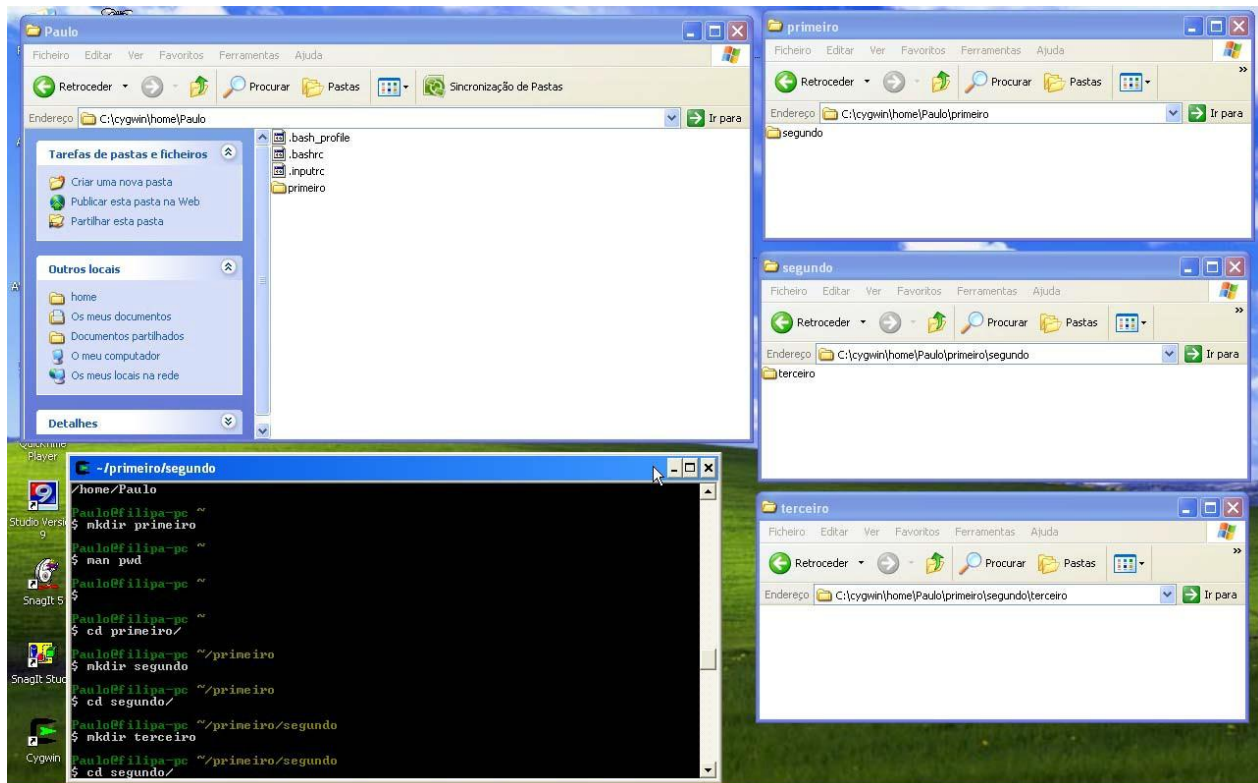
O comando *mkdir* (*make directory*) cria uma directoria. Vamos mudar para lá a nossa directoria de trabalho. Faça então o comando “*cd*” (*change directory*) do modo seguinte

```
cd primeiro
```

¹ Dependendo do programa que está a gerir os comandos de linha na consola esta instrução pode ser ligeiramente diferente. No Cygwin é como se mostra.

Repare que no **prompt** já não aparece simplesmente o til, mas agora aparece “~/primeiro” para nos indicar qual a directoria de trabalho. Já estamos dentro da pasta **primeiro**, como pode também verificar fazendo “**pwd**”. Crie agora uma nova pasta aí chamada de “segundo”. Mude depois para essa pasta e crie uma directoria chamada de “terceiro”.

Coloque agora as janelas Windows do seguinte modo: Janela 1 na directoria base (a tal com os três ficheiros de configuração); janela 2 na directoria primeiro (tem apenas uma pasta chamada “segundo”); janela 3 na directoria segundo (tem apenas uma pasta chamada “terceiro”); janela 4 na directoria terceiro (não tem nada). A figura abaixo mostra o que se pede.



Como última actividade desta secção faça o comando seguinte na janela Cygwin

```
cd
```

O que aconteceu? Faça “**pwd**” na janela para ver o que aconteceu e depois se tem dúvidas leia a nota de pé de página².

Falta testar o comando “**rmdir**” que apaga pastas/directorias. Não se faz aqui nenhum exercício pois precisamos das directorias que foram criadas.

6.3 COMANDOS SOBRE FICHEIROS

O que está numa directoria? Nas janelas de Windows vê-se bem quais são os ficheiros. Vamos ver na janela do Cygwin...

² O comando “**cd**” sem mais nada faz com que a directoria de trabalho volte a ser a directoria base. Às vezes é importante para se ir directamente para lá.

Nessa janela execute o seguinte comando

```
ls
```

A resposta é o ficheiro “*primeiro*” que é uma pasta. Ora, não se percebe de facto se o que está escrito é um ficheiro ou uma pasta. Também sabemos que existem lá três ficheiros (os tais começados por ponto”) que não aparecem...

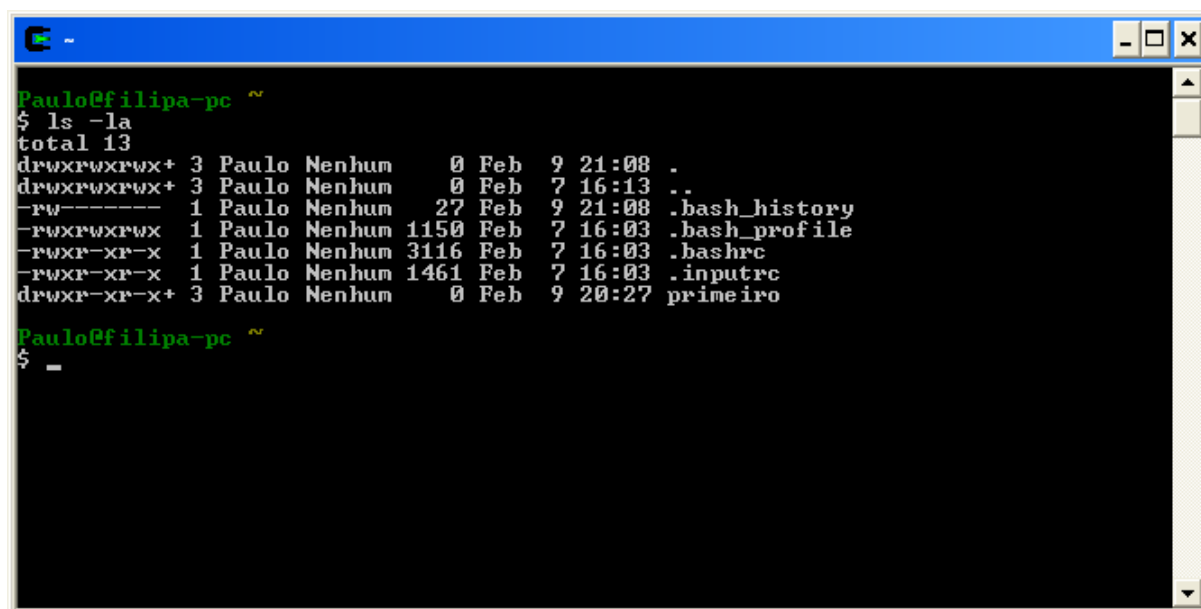
O comando *ls* tem a possibilidade de colocarmos opções e com isso saber mais coisas. Um modo de se saber as opções possíveis é usarmos o manual

```
man ls
```

Faça então o comando *ls* com as opções “*l*” e “*a*”, de *long* e de *all*. As opções escrevem-se depois de um hífen.

```
ls -la
```

Agora já aparece qualquer coisa como o seguinte



```
Paulo@filipa-pc ~  
$ ls -la  
total 13  
drwxrwxrwx+ 3 Paulo Nenhum 0 Feb 9 21:08 .  
drwxrwxrwx+ 3 Paulo Nenhum 0 Feb 7 16:13 ..  
-rw----- 1 Paulo Nenhum 27 Feb 9 21:08 .bash_history  
-rwxrwxrwx 1 Paulo Nenhum 1150 Feb 7 16:03 .bash_profile  
-rwxr-xr-x 1 Paulo Nenhum 3116 Feb 7 16:03 .bashrc  
-rwxr-xr-x 1 Paulo Nenhum 1461 Feb 7 16:03 .inputrc  
drwxr-xr-x+ 3 Paulo Nenhum 0 Feb 9 20:27 primeiro  
Paulo@filipa-pc ~  
$
```

É importante perceber o que está escrito, mas antes veja a figura abaixo para a explicação de alguns campos.

Vamos agora criar ficheiros em vários locais. Como ainda não sabemos realmente escrever ficheiros que façam sentido vamos usar os ficheiros que já existem. Note que o propósito é perceber os comandos de manipulação de ficheiros, e não propriamente o que os ficheiros contêm.

Relembrando, a janela Cygwin está na directoria base. Faça então o seguinte comando

```
cp .bashrc banana
```

Este comando copiou o ficheiro *.bashrc* para outro ficheiro com o nome de *banana*. Se fizer “*ls -la*” pode verificar isso, mas também pode verificar na janela 1. Os ficheiros são iguais. Um modo de vermos isso é vermos o que o ficheiro contém. Para isso use o seguinte comando

```
cat .bashrc
```

e depois

```
cat banana
```

Deve ter percebido que o ficheiro passou tão depressa que não deu para ver o início. Um modo de controlar mais as coisas é usar “*more*” em vez de “*cat*”. O “*more*” mostra um ecrã de cada vez e quando tocamos na *barra de espaço* passa para o ecrã seguinte. Se quisermos ver linha a linha use o “*Enter*”. Quando quiser sair carregue em “*q*” de “*quit*”. Por esta altura já deve ter reparado que o Cygwin não tem “*more*”.

Não faz mal, existe o “*less*” que faz o mesmo que o “*more*” e mais ainda pois permite andar um ecrã para trás quando se prime a tecla “*b*” de “*back*”. As páginas do manual usam o *less*. Faça então

```
less banana
```

Podemos mudar o nome do ficheiro banana. Faça o seguinte comando

```
mv banana bacano
```

e verifique na janela 1 a alteração. Agora vamos criar um ficheiro dentro da directoria *primeiro*. Faça o seguinte comando na janela Cygwin e repare na janela 2.

```
cp bacano primeiro/tomate
```

Já agora para aprender uma utilização do ponto “.” faça o seguinte comando

```
cp primeiro/tomate .
```

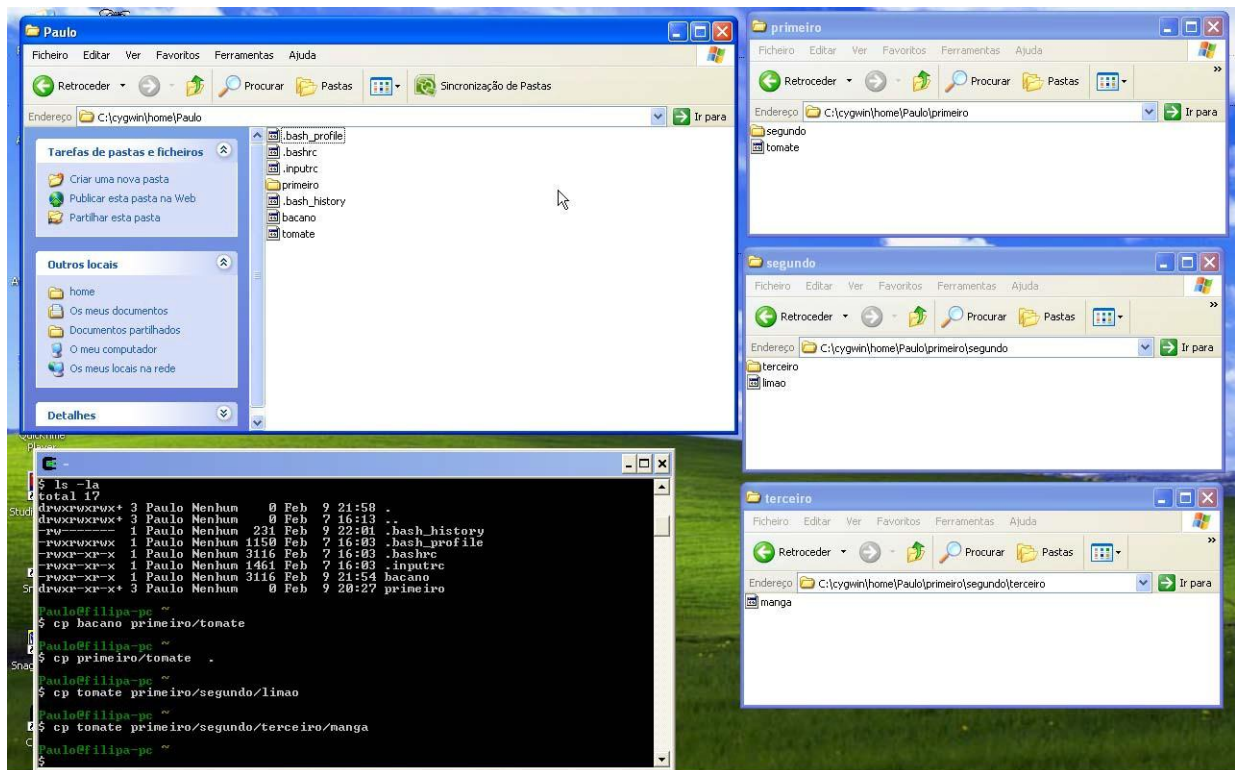
Olhe para as janelas 1 e 2 e veja o que aconteceu. Qual o significado do ponto neste comando? Se não percebeu pergunte ao docente.

Faça agora a seguinte sequência de comandos na janela Cygwin. Já agora saiba que o Unix e o Linux foram desenhados por pessoal que entendia “*o esforço mínimo*” como um lema de vida. Nos comandos seguintes faça uma coisa quando os está a escrever.

Depois de escrever a primeira letra dos nomes das directorias ou dos ficheiros carregue em *Tab* (->) em vez de escrever o resto das letras.


```
cp tomate primeiro/segundo/limão
cp tomate primeiro/segundo/terceiro/manga
```

O ecrã deve estar como mostra a figura.



Vamos então aprender a apagar coisas. Mude a directoria de trabalho na janela Cygwin para a directoria “segundo”. Já agora escreva sempre a primeira letra e **Tab**. Temos de ser preguiçosos para honrar o espírito do Linux!...

Vamos então apagar a directoria “terceiro” com o comando **rmdir**. Faça o seguinte comando

```
rmdir terceiro
```

O que aconteceu? A directoria não foi apagada pois não estava vazia. Tinha lá um ficheiro – o ficheiro **manga**. Podemos fazer duas coisas.

- 1) Forçar o comando a apagar directorias não vazias por meio da escolha de uma opção (vai ter de aprender qual é essa opção com o **man**);
- 2) Apagar todos os ficheiros da directoria antes de a apagar a ela (ficam apenas os ficheiros ponto e ponto-ponto).

Vamos fazer esta segunda hipótese pois assim vamos usar o comando “**rm**”.

A directoria onde estamos é a de cima. Assim podemos apagar o ficheiro **manga** a partir daqui (mostrado em baixo à esquerda), ou ir para a directoria de baixo, apagar o ficheiro, e voltar para cima (mostrado em baixo à direita).

```
rm terceiro/manga      cd terceiro
                        rm manga
                        cd ..
```

Agora sim, podemos apagar a directoria.

```
rmdir terceiro
```

Para ver se já domina isto tudo, apague todos os ficheiros e directorias que acabámos de criar **sempre a partir** da janela Cygwin. Fique outra vez com a directoria base apenas com os três ficheiros de configuração.

6.4 OUTROS COMANDOS

Esta secção será escrita proximamente.

7 O meu primeiro programa em C

7.1 USO DA *MEMORY STICK*

As áreas no laboratório 1.1-X são apagadas depois de cada aula terminar. Assim, os alunos têm de guardar os seus programas nas chamadas *Memory Sticks* com interface USB que em Portugal são conhecidas pelo nome bem Português de “**PEN**”.

Quando introduzimos o dispositivo na interface USB o Windows reconhece-o e no mundo Linux, ou Cygwin, o dispositivo é montado na directoria “/cygdrive/D” em que **D** é o nome da partição que o Windows lhe deu.

É tudo o que é preciso saber. Agora é fácil copiar os ficheiros para a *Memory Stick* e guardá-los para a próxima aula. Se quiser fazer as coisas de um modo gráfico, use as janelas do Windows...

A parte interessante é que se for à directoria /cygdrive nas janelas de Windows não consegue ver a *memory stick*...

7.2 O PROGRAMA PARA SAUDAR O MUNDO

Nesta secção vai-se fazer um primeiro programa em C. O objectivo é ensinar como se usa o compilador de C.

Comece por arrancar o editor Crimson e escreva o seguinte programa

```
#include <stdio.h>

main ()
{
    printf ("Olá Mundo\n");
}
```

Guarde este ficheiro na directoria base com o nome “**ola.c**”. Na janela Cygwin compile então o programa. Para isso faça o seguinte comando

```
gcc -o ola ola.c
```

A opção “-o” do compilador serve para indicar ao compilador qual o nome que queremos para o ficheiro de saída, o ficheiro executável. Verifique com o comando *ls -la* que o ficheiro que queríamos foi criado.

Podemos então correr o programa. Para isso basta chamar o programa como se ele fosse um comando de linha. Faça então

```
ola
```

Como deve ter reparado, **NÃO FUNCIONOU!** O problema está que a directoria onde se está não pertence ao conjunto de directorias onde o sistema procura comandos de linha. O conjunto dessas directorias é designado de *\$PATH* e pode ser visto com o comando

```
echo $PATH
```

Como a directoria onde se está não pertence, temos de forçar o correr este programa com o seguinte comando de linha

```
./ola
```

Conseguiu?

Já agora. É capaz de dizer para que serve o comando seguinte e o que se ganha com isso?

```
export PATH=.: $PATH
```

...e se em vez do anterior se tivesse feito este?

```
export PATH=$PATH:.
```

BOM

LINUX

E BOA

PROGRAMAÇÃO DE MICROPROCESSADORES