

Fundamentos de Programação

Guião da Aula 1

António Neves

João Rodrigues



2017–2018

Aula Prática 1

Resumo:

- Introdução ao sistema operativo UNIX.
- O ambiente de trabalho para as aulas práticas.
- Conceitos base da linguagem Python.
- Edição e execução de programas em Python.

As aulas práticas de Fundamentos de Programação decorrem em salas equipadas com computadores pessoais (PCs) correndo o sistema operativo Linux. O Linux (ou mais corretamente, GNU/Linux) é uma variante *livre e gratuita* do conhecido sistema operativo UNIX.

Na Universidade de Aveiro, um grupo de utilizadores de Linux denominado GLUA¹ disponibiliza diversas distribuições populares de Linux e organiza sessões de esclarecimento e de ajuda para quem estiver interessado em instalar e utilizar este sistema.

Na página da disciplina no elearning vão também encontrar um ficheiro que contém uma imagem de um sistema Linux e que pode ser usado como máquina virtual nos vossos computadores. Em caso de dúvida, falem com os professores da vossa turma prática sobre a sua utilização.

1.1 O Arranque, *Login* e *Logout*

Os computadores das salas de aula têm atualmente dois sistemas operativos instalados: o Windows e o Linux.

Assim, ao ligar o computador será confrontado com um menu para escolher o sistema que deseja iniciar. Terá alguns segundos para escolher a opção certa (o Linux, neste caso), usando as teclas de direção ↑ ou ↓ e a tecla **Enter** ↵. Se o computador já se encontrar ligado e a correr Windows, deverá seleccionar a opção para reiniciar e poder voltar ao menu de arranque.

Logo que o sistema esteja em funcionamento, aparece um ecrã de boas-vindas onde terá de se identificar, introduzindo o *nome-de-utilizador* (*username*) do tipo **a12345** (sem **@ua.pt**), e a *palavra-passe* (*password*) correspondente. Estes dados são os mesmos que

¹<http://glua.ua.pt>

utiliza para aceder ao ambiente Windows. Se introduziu os dados corretos, surge um ambiente gráfico que lhe permite interagir com o sistema e completar os exercícios da aula. Chama-se *entrar no sistema* (em Inglês *log in* usualmente escrito *login*) a este processo de autenticação para ter acesso ao sistema.

Quando terminar de usar o sistema, deve sempre *sair do sistema* (*log out* ou *logout*) de forma a que mais ninguém tenha acesso à sua área de trabalho. Se quiser desligar ou reiniciar o computador deve escolher a acção desejada no ecrã de boas-vindas que entretanto reaparece.

Exercício 1.1

Entre no sistema, introduzindo o seu nome-de-utilizador e palavra-chave na janela de *login*. Explore os menus e ícones do ambiente gráfico. Descubra a opção de *Log Out* (geralmente *System/Quit/Log Out*) e seleccione-a para sair do sistema. Repita o processo de *login* para regressar ao sistema.

1.2 A Linha de Comandos UNIX

Quando o sistema UNIX foi concebido, os computadores eram controlados essencialmente através de *consolas* ou *terminais* de texto: dispositivos dotados de um teclado e de um ecrã onde se podia visualizar somente texto. A interação com o sistema fazia-se tipicamente através da introdução de comandos escritos no teclado e da observação da resposta produzida no ecrã pelos programas executados. Actualmente existem ambientes gráficos que correm sobre o UNIX e permitem visualizar informação de texto e gráfica, e interagir por manipulação virtual de objectos gráficos recorrendo a um rato e ao teclado. É o caso do Sistema de Janelas X, ou simplesmente X, que está instalado nos PCs das salas de aula.

Apesar das novas formas de interação proporcionadas pelos ambientes gráficos, continua a ser possível e em certos casos preferível usar a interface de *linha de comandos* para muitas operações. No X, isto pode fazer-se usando um *emulador de terminal*, um programa que abre uma janela onde se podem introduzir comandos linha-a-linha e observar as respostas geradas tal como num terminal de texto à moda antiga.

Exercício 1.2

Abra uma janela de terminal (a partir do menu principal)² e quando surgir o *prompt*³ execute o comando `date`.

Observe que a resposta foi impressa imediatamente a seguir à linha do comando, de forma concisa, sem distrações nem grandes explicações. Este comportamento é usual em muitos comandos UNIX e é típico de um certo estilo defendido pelos criadores deste sistema. Simples, mas eficaz.

Exercício 1.3

²Possivelmente: *Applications/Accessories/Terminal*.

³<http://pt.wikipedia.org/wiki/Prompt>

Execute o comando `cal` e observe o resultado. Descubra em que dia da semana nasceu, passando o mês e o ano como *argumentos* ao comando `cal`, por exemplo: `cal jan 1981`.

Os comandos em UNIX têm sempre a forma:

```
comando argumento1 argumento2 ...
```

onde **comando** é o nome do programa a executar e os argumentos são cadeias de caracteres, que podem ser incluídas ou não, de acordo com a sintaxe esperada por esse programa.

Na linha de comandos é possível recapitular um comando dado anteriormente usando as teclas de direção ↑ e ↓. É possível depois editá-lo para produzir um novo comando com argumentos diferentes, por exemplo. Outra funcionalidade muito útil é a possibilidade de o sistema completar automaticamente comandos ou argumentos parcialmente escritos usando a tecla `Tab`.

1.2.1 Navegação no Sistema de Ficheiros

Tal como noutros sistemas operativos, no UNIX a informação é armazenada numa estrutura hierárquica formada por diretórios, subdiretórios e ficheiros. O diretório-raiz desta árvore é representado simplesmente por uma barra “/”. Cada utilizador possui um diretório próprio nesta árvore, a partir do qual pode (e deve) criar e gerir toda a sua sub-árvore de diretórios e ficheiros: é o chamado *diretório do utilizador* ou *home directory*. Após a operação de *login* o sistema coloca-se nesse diretório. Portanto neste momento deve ser esse o *diretório actual* (*current directory*).

Para saber qual é o diretório actual execute o comando `pwd`. Deve surgir um nome como

```
/homermt/a12345
```

que indica que está no diretório `a12345` que é um subdiretório de `homermt` que é um subdiretório directo da raiz `/`.

Para listar o conteúdo do diretório actual execute o comando `ls`. Deve ver uma lista dos ficheiros (e subdiretórios) contidos no seu diretório neste momento, por exemplo:

```
arca Desktop Examples
```

Neste caso, observam-se dois subdiretórios e um *soft link* que é um tipo de ficheiro especial que serve de atalho para outro ficheiro ou diretório. Dependendo da configuração do sistema, os nomes nesta listagem poderão aparecer com cores diferentes e/ou com uns caracteres especiais (`/`, `@`, `*`) no final, que servem para indicar o tipo de ficheiro mas de facto não fazem parte do seu nome.

(Num ambiente gráfico a mesma informação está disponível numa representação mais visual. Experimente, por exemplo, escolher *Places/Home Folder* para ver o conteúdo do seu diretório pessoal.)

Ficheiros cujos nomes começam por “.” não são listados por defeito, são ficheiros *escondidos*, usados geralmente para guardar informações de configuração de diversos programas.

Para listar todos os ficheiros de um diretório, incluindo os escondidos, deve executar a variante **ls -a**.

Por vezes é necessário listar alguns atributos dos ficheiros para além do nome. Pode fazê-lo executando as variantes **ls -l** ou **ls -la**.

```
total 88
drwx----- 13 a12345 users 4096 2007-01-26 14:03 .
drwxr-xr-x   3 root    root  4096 2007-01-25 10:52 ..
drwx-----  1 a12345 users    0 2007-01-26 08:00 arca
drwxr-xr-x   2 a12345 users 4096 2007-01-25 10:52 Desktop
lrwxrwxrwx   1 a12345 users   26 2007-01-25 10:52 Examples -> ...
```

Os principais atributos mostrados nestas listagens longas são:

Tipo de ficheiro identificado pelo primeiro carácter à esquerda, sendo **d** para diretório, **-** para ficheiro normal, **l** para *soft link*, etc.

Permissões representadas por 3 conjuntos de 3 caracteres. Indicam as permissões de leitura **r**, escrita **w** e execução/pesquisa **x** relativamente ao dono do ficheiro, aos outros elementos do mesmo grupo e aos restantes utilizadores da máquina.

Propriedade indica a que utilizador e a que grupo pertence o ficheiro.

Tamanho em número de bytes.

Data e hora da última modificação.

Nome do ficheiro.

Normalmente existe um *alias*⁴ **ll** equivalente ao comando **ls -l**.

Além do **ls** e variantes, existem outros comandos importantes para a observação e manipulação de diretórios, por exemplo:

cd — o diretório actual passa a ser o diretório do utilizador.

cd dir — o diretório actual passa a ser o diretório **dir**.

mkdir dir — cria um novo diretório chamado **dir**.

rmdir dir — remove o diretório **dir**, desde que esteja vazio.

O argumento **dir** pode ser dado de uma forma absoluta ou relativa. Na forma absoluta, **dir** identifica o caminho (*path*) para o diretório pretendido a partir da raiz de todo o sistema de ficheiros; tem a forma **/subdir1/.../subdirN**. Na forma relativa, **dir** indica o caminho para o diretório pretendido a partir do diretório actual; tem a forma **subdir1/.../subdirN**.

⁴Um *alias* é um nome alternativo usado em representação de um determinado comando. São criados usando o comando interno **alias**.

Há dois nomes especiais para diretórios: “.” e “..” que representam respectivamente o diretório actual e o diretório pai, ou seja, o diretório ao qual o actual pertence.

Exercício 1.4

Execute os comandos seguintes e interprete os resultados:

```
ls -l /
cd /
pwd
ls -l
cd usr
ls
cd local/src
pwd
ls
cd ../../bin
ls
cd
pwd
```

Exercício 1.5

Experimente utilizar o programa gráfico gestor de ficheiros⁵ para navegar pelos mesmos diretórios que no exercício anterior: /, /usr, /usr/local/src, etc.

Exercício 1.6

Mude o diretório actual para o seu subdiretório **arca**. Liste o seu conteúdo. Reconhece algum dos ficheiros?

Importante: O subdiretório **arca** não é um diretório local do PC onde está a trabalhar; é na verdade a sua área privada de armazenamento no Arquivo Central de Dados (ARCA⁶), um servidor de ficheiros da Universidade de Aveiro. Esta área também é acessível a partir do ambiente Windows e através da Web, e é natural que já aí tenha colocado ficheiros noutras ocasiões. É neste diretório que deve gravar os ficheiros e diretórios que criar no decurso das aulas práticas. Os computadores das salas de aulas foram programados para apagarem o diretório de utilizador (e.g. /homermt/a1245/) sempre que são reiniciados. Só o conteúdo do subdiretório **arca** é salvaguardado. É portanto aí que deve colocar todo o seu trabalho.

Exercício 1.7

Crie, no diretório **arca**, um subdiretório chamado **fp1** e, dentro desse, um diretório chamado **aula01**.

⁵Acessível no menu *Places*.

⁶<https://arca.ua.pt>

1.2.2 Manipulação de ficheiros

O Linux (UNIX) dispõe de diversos comandos de manipulação de ficheiros. Eis alguns:

`cat fic` — imprime no dispositivo de saída *standard* (por defeito o ecrã) o conteúdo do ficheiro `fic`.

`rm fic` — remove (apaga) o ficheiro `fic`.

`mv fic1 fic2` — muda o nome do ficheiro `fic1` para `fic2`.

`mv fic dir` — move o ficheiro `fic` para dentro do diretório `dir`.

`cp fic1 fic2` — cria uma cópia do ficheiro `fic1` chamada `fic2`.

`cp fic dir` — cria uma cópia do ficheiro `fic` dentro do diretório `dir`.

`head fic` — mostra as primeiras linhas do ficheiro de texto `fic`.

`tail fic` — mostra as últimas linhas do ficheiro de texto `fic`.

`more fic` — imprime no dispositivo de saída *standard* (por defeito o ecrã), página a página, o conteúdo do ficheiro `fic`.

`grep padrão fic` — selecciona as linhas do ficheiro texto `fic` que satisfazem o critério de selecção `padrão`.

`wc fic` — conta o número de linhas, palavras e caracteres do ficheiro `fic`.

`sort fic` — ordena as linhas do ficheiro `fic`.

`find dir -name fic` — procura um ficheiro com o nome `fic` a partir do diretório `dir`.

Além destes pode ainda considerar outros tais como: `less`, `cut`, `paste`, `tr`, etc. Todos estes comandos podem ser invocados usando argumentos opcionais que configuram o seu modo de funcionamento.

Exercício 1.8

Efectue o download do ficheiro `primeiro.py`, que se encontra no Moodle, para o diretório `aula01` que criou no exercício anterior. Imprima o seu conteúdo no ecrã. Experimente outros comandos da lista acima.

1.2.3 Ajuda *On-line*

O Linux dispõe de vários mecanismos de ajuda imediata para a maioria dos seus comandos. Dois dos mais importantes são acedidos através dos comandos `man` e `info`, sendo o primeiro comum em todos os sistemas UNIX e o segundo mais específico do projecto GNU. Muitos comandos aceitam também uma opção `--help` que apresenta um resumo da sua forma de utilização.

Por exemplo, para conhecer as muitas opções de execução do comando `ls` pode executar `man ls`, ou `info ls`, ou `ls --help`.

Nota: Para navegar ao longo das páginas apresentadas pelo `man` ou pelo `info` pode usar as teclas de direcção `↑`, `↓` ou as teclas `PageUp`, `PageDown`. Para abandonar as páginas de ajuda e regressar à linha de comando deve premir a tecla `q`. Estes programas têm outras possibilidades de navegação e pesquisa que poderá ficar a conhecer fazendo por exemplo, `man man` ou `info info`.⁷

1.3 Ambiente de Programação em Python

1.3.1 Edição

Comece por editar o programa `primeiro.py`. Para esse efeito dispõe de vários editores de texto. Aconselhamos, no entanto, a usar o `geany` ou o `gvim` (*VI editor*), visto possuírem a função de realce da sintaxe da linguagem Python. Na janela de terminal pode usar o editor `vim`, embora este editor tenha uma aprendizagem mais difícil.⁸

Pode também utilizar ambientes de programação integrados como o `PyCharm`.

1.3.2 Execução

O ficheiro que acabou de editar é usualmente designado por programa fonte. O passo seguinte consiste em executar o programa desenvolvido. Isto é feito usando o comando `python3 primeiro.py` no terminal⁹ o que, se não houver erros, é apresentado o resultado da execução do programa.

1.4 Problemas para resolver

Explore os vários exemplos de programas em Python que estão no elearning. Experimente alterar alguns valores dos mesmos e avalie o resultado obtido.

⁷Pelo contrário, `busca busca`, `mata mata`, não têm qualquer significado conhecido em UNIX, mas pode sempre tentar!

⁸Também é possível lançar a partir da linha de comandos qualquer outro editor ou programa. Por exemplo, experimente o comando `geany &`.

⁹Caso existam erros de compilação, eles serão apresentados, pela ordem com que foram detectados, na própria janela do terminal.