

Programação de Computadores (2013-2014) Ambiente de Trabalho

1 Introdução

Na implementação de projectos de alguma dimensão é usual utilizar o que normalmente se designa por Ambiente Integrado de Desenvolvimento (IDE – *Integrated Development Environment*), como por exemplo Borland C++, Microsoft Visual C++, ou Bloodshed Software – Dev-C++ (de domínio público para Windows e Linux).

Na unidade curricular de Estruturas de Dados e Algoritmos, em que o programas desenvolvidos já terão alguma complexidade, utilizarão o Microsoft Visual C++.

Na presente unidade curricular de Programação de Computadores, e com o objectivo de clarificar as fases:

- Escrita do programa fonte, usando um qualquer editor de texto (de preferência sensível ao contexto); estes ficheiros podem ter a extensão “cpp” ou “cc” (ou seja nomes do tipo *.cpp, ou *.cc).
- Compilação e criação do programa objecto. Estes ficheiros têm geralmente a extensão “obj” ou “o” (ou seja nomes do tipo *.obj ou *.o).
- *linker*, para ligar ficheiros (ficheiros objecto que fazem parte de um projecto, ou com uma biblioteca) e obter o ficheiro executável, que no linux geralmente não tem extensão (no sistema Windows os ficheiros executáveis têm geralmente a extensão ‘**exe**’, ou seja têm um nome do tipo “*.exe”).

nas duas primeiras aulas a compilação será executada na linha de comando, ou seja num terminal.

Em programas de alguma dimensão é também útil recorrer a um *debugger*, para localizar e remover eventuais erros de execução. Contudo, no âmbito desta unidade curricular, os alunos farão o *debug* dos programas sem recorrer a esse tipo de ferramenta – embora o docente possa usar o gdb (*The GNU¹ Debugger*) para ilustrar o funcionamento de um programa.

¹GNU é acrónimo recursivo: GNU is Not Unix

2 Contacto

Cada aluno uma vez aberta a sua conta no **domínio DEEC**, possui uma conta no servidor **alunos** do DEEC: é essa conta que permite a cada aluno fazer *login* no domínio DEEC.

Sempre que um docente de Programação de Computadores quiser contactar um aluno por correio electrónico utilizará (preferencialmente) o endereço: **aN@alunos.deec.uc.pt**, onde N representa o número desse estudante na Universidade de Coimbra.

Assim devem ter o hábito de consultar o vosso endereço de correio electrónico no DEEC. Se temem esquecer-se de o fazer então redireccionem-no para a vossa conta de correio electrónico habitual. Para conseguir esse redireccionamento basta criar na directoria raiz da vossa conta no servidor **alunos.deec.uc.pt** um ficheiro de nome **.forward** (o ponto como primeiro carácter do nome do ficheiro é importante) e colocar como único conteúdo o vosso endereço de correio electrónico habitual.


3 Ambiente de trabalho

3.1 Primeira sessão no PC

As aulas de programação vão decorrer em ambiente Linux (Cent OS). Deverá utilizar sempre o mesmo computador para aí encontrar o trabalho realizado em aulas anteriores. Para fazer *login* utilizará o nome de utilizador **aN** e a senha correspondente de acesso ao domínio DEEC.

Ao fazer *login* pela primeira vez, com o nome de utilizador e a senha fornecidos, ficará com um ambiente como se indica na figura 1 (ver página 3) – neste texto o nome do utilizador é “teresa”. Note que por omissão tem dois “mundos” (ver botão no canto inferior direito).

Para navegar nas suas pastas pode clicar com o rato na pasta que é a sua directoria base (*home directory*), como no exemplo da figura 2 (onde está a usar o programa *nautilus*).

Se clicar com o rato no símbolo  **Applications** do canto superior esquerdo do ecrã na figura 1 surge um menu com uma lista de programas. Para abrir um terminal pode seguir o exemplo ilustrado na figura 5(a) ficará com um terminal (consola) – ver figura 5(b) (na página 7).

Na consola ao fazer `pwd <enter>`² obtém “/home/teresa” ou seja a *home directory* ou seja a pasta do utilizador “teresa” (no seu caso em vez de “teresa” surgirá o seu nome de utilizador). Se em seguida fizer `ls -la <enter>`³ obtém todos os ficheiros na pasta corrente (no caso a *home directory* do utilizador “teresa”), por ordem alfabética (ignorando o “.”). Como pode

²<enter> significa carregar na tecla *return* ou *enter*.

³Cada comando na consola só é enviado para o sistema depois de premir o <enter>. Dito isto, daqui em diante será omitida a necessidade de terminar cada comando com um <enter>.

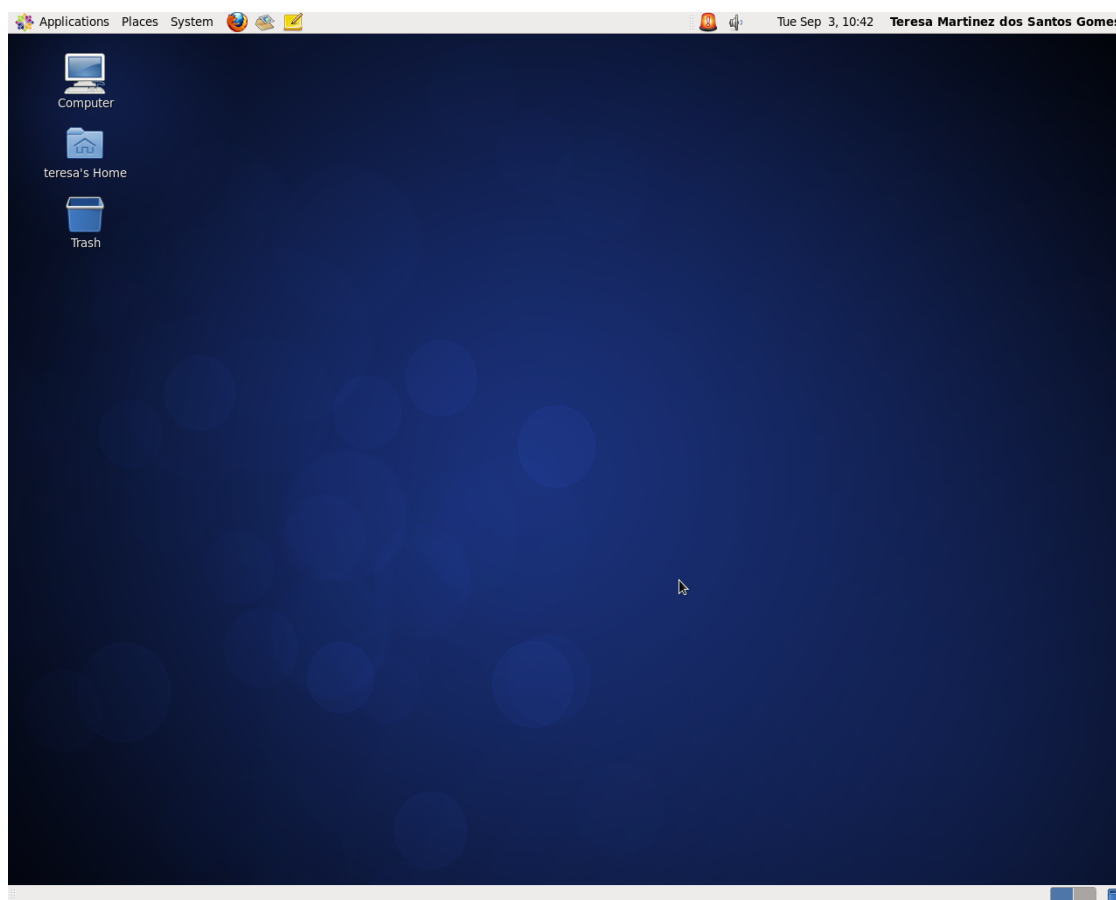


Figura 1: Ambiente de trabalho após o *login*

verifica na figura 3.1 (na página 6) algumas pastas já foram criadas pelo sistema (a lista de ficheiros foi o resultado do comando `ls -la`). Os ficheiros cujo nome começa por “.” são especiais em geral não são visíveis, e por isso não aparecem na figura 2 na página 4).

Poderá utilizar o navegador na *web* `mozilla`, para o qual tem um atalho na barra superior do seu ecrã. Contudo recomenda-se que traga em papel os enunciados da folhas práticas de forma a deixar o ecrã livre apenas para a escrita e teste dos programas. Trazendo os enunciados em papel, *em caso de problemas de acesso ao InforEstudante*, terá sempre o material necessário ao bom desenrolar da aula.

Os seus programas serão escritos usando o editor de texto `geany` o qual utiliza o *GTK2 toolkit* para fornecer as funções básicas de um sistema integrado de desenvolvimento. Contudo nas primeiras duas aulas o `geany` deve ser usado apenas como editor de texto e a compilação (usando o `g++` – ver secção 3.3), criação do ficheiro executável e execução deste, deve ser feita na linha de comando – ver Figura 5.

O editor de texto `geany` (<http://www.geany.org/>) aceita os atalhos habituais do ambiente

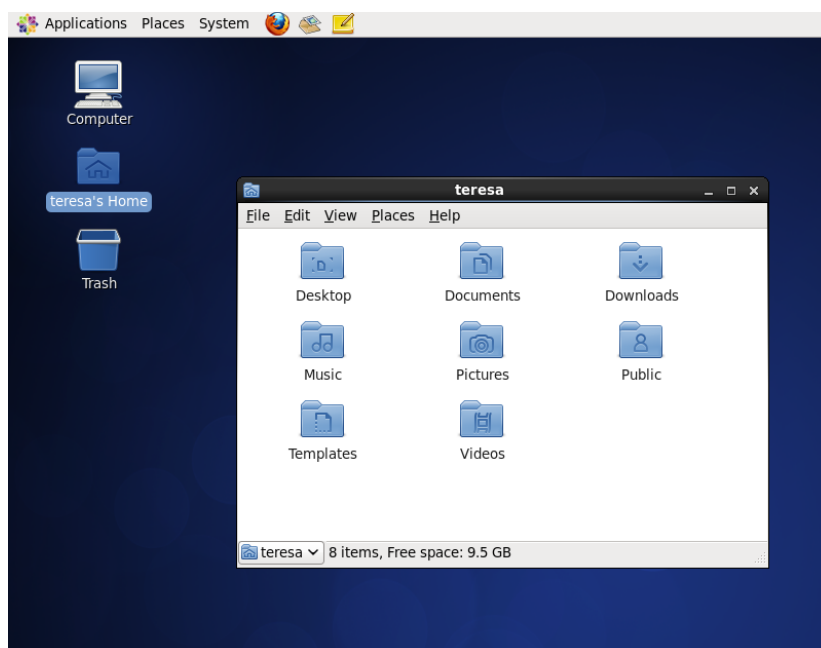


Figura 2: Navegando usando o *nautilus*.

Windows (“Ctrl+C” para copiar, “Ctrl+V” para colar, etc).

O **geany** também pode ser invocado a partir da consola: `geany &`. O símbolo “&” no final de um comando solicita execução do mesmo em *background* libertando a linha de comando para outra instrução.

Na figura 5 encontra o ambiente de trabalho com um terminal, a pasta `aula1`, a pasta `Exemplo` e o **geany** com o ficheiro “`bom_dia.cpp`” na pasta `Exemplo` aberto. Para criar uma pasta ou um ficheiro basta clicar com o rato direito sobre uma pasta – ver Figura 6.

IMPORTANTE: No final de cada aula deve copiar os exercícios realizados na aula para uma PEN ou para a sua conta no servidor **alunos**. Se houver algum problema com o PC este pode ter de ser formatado, e nessa caso perderá toda a informação nele contida. Ou em alternativa envie-os por correio electrónico para si.

Para fazer uma cópia para o servidor **alunos**, na consola deve fazer:

```
scp -r aulaW aN@alunos.deec.uc.pt:pdC/
```

onde onde **N** representa o número do estudante na Universidade de Coimbra, **W** representa o número corrente da aula e onde *se assume que a pasta `pdC` foi criada previamente* no servidor **alunos**.

Para criar a pasta `pdC` no servidor **alunos** faça *login* remoto: `ssh aN@alunos.deec.uc.pt`. O servido **alunos** solicita-lhe a sua senha (a de acesso ao domínio DEEC). Uma vez feito o *login* pode criar a pasta `pdC`, fazendo `mkdir pdC` e em seguida desligue digitando: `exit`.

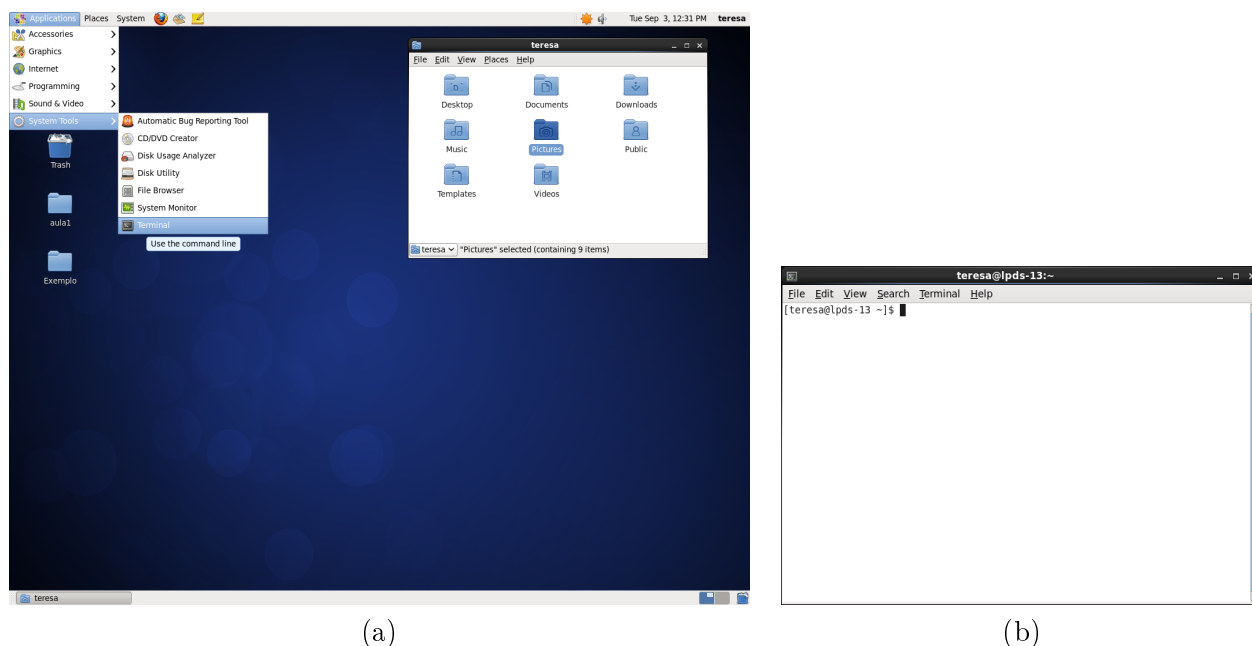


Figura 3: Abrindo um terminal (consola)

3.2 Organização

No servidor do Laboratório os ficheiros (código fonte, objecto e executáveis) **devem ser criados na respectiva directoria de trabalho** em `aulaz`, em que `z` é a ordem da aula laboratorial em curso – em cada aula laboratorial criará a respectiva directoria: `aulaz`, com $z = 1, 2, 3, \dots, 12$.

Cada ficheiro criado deve ter um nome do tipo `f α e β .cpp`, em que α é o número da folha de problemas e β o número do exercício.

3.3 O compilador

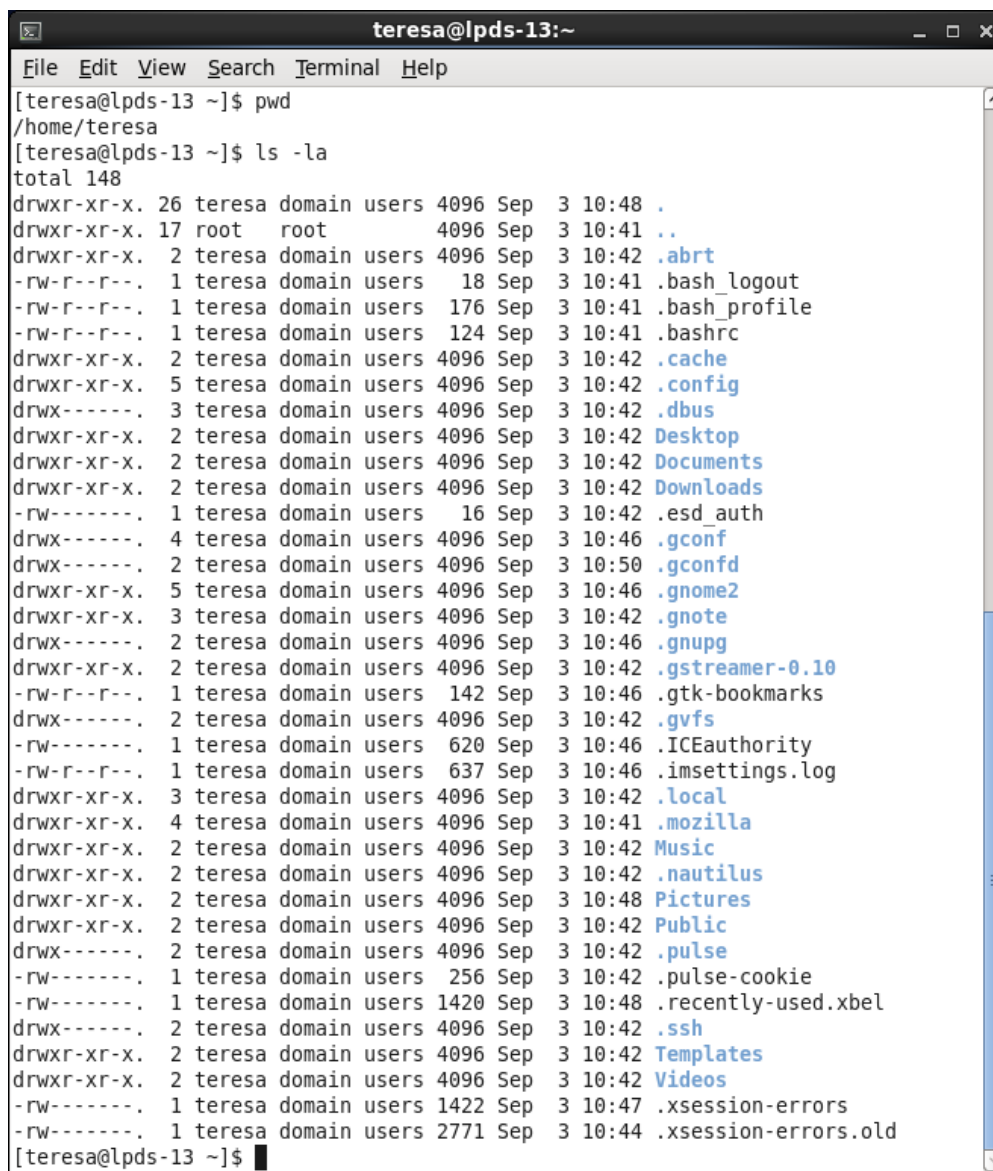
O compilador escolhido foi o `g++` da GNU (<http://gcc.gnu.org/>) pois é de domínio público e será também utilizado no contexto de Sistemas de Microprocessadores no 2º semestre do 1º ano do MEEC.

O `g++` funciona em linha de comando. Assim para compilar um ficheiro e criar um ficheiro executável, deverá escrever na linha de comando do terminal

```
g++ -c -Wall nomeficheiro.cpp
```

O interruptor `-c` indica que deseja a *compilação* do ficheiro `nomeficheiro.cpp` o que deve resultar na criação de `nomeficheiro.o`, se não houver erros de compilação. O ficheiro em código objecto (`nomeficheiro.o`) contém rotinas numa forma localizável.

O interruptor `-Wall` activa um conjunto de avisos (pode não ser utilizado, mas alguns



```
teresa@lpds-13:~  
File Edit View Search Terminal Help  
[teresa@lpds-13 ~]$ pwd  
/home/teresa  
[teresa@lpds-13 ~]$ ls -la  
total 148  
drwxr-xr-x. 26 teresa domain users 4096 Sep  3 10:48 .  
drwxr-xr-x. 17 root root 4096 Sep  3 10:41 ..  
drwxr-xr-x. 2 teresa domain users 4096 Sep  3 10:42 .abrt  
-rw-r--r--. 1 teresa domain users  18 Sep  3 10:41 .bash_logout  
-rw-r--r--. 1 teresa domain users 176 Sep  3 10:41 .bash_profile  
-rw-r--r--. 1 teresa domain users 124 Sep  3 10:41 .bashrc  
drwxr-xr-x. 2 teresa domain users 4096 Sep  3 10:42 .cache  
drwxr-xr-x. 5 teresa domain users 4096 Sep  3 10:42 .config  
drwx-----. 3 teresa domain users 4096 Sep  3 10:42 .dbus  
drwxr-xr-x. 2 teresa domain users 4096 Sep  3 10:42 Desktop  
drwxr-xr-x. 2 teresa domain users 4096 Sep  3 10:42 Documents  
drwxr-xr-x. 2 teresa domain users 4096 Sep  3 10:42 Downloads  
-rw-----. 1 teresa domain users  16 Sep  3 10:42 .esd_auth  
drwx-----. 4 teresa domain users 4096 Sep  3 10:46 .gconf  
drwx-----. 2 teresa domain users 4096 Sep  3 10:50 .gconfd  
drwxr-xr-x. 5 teresa domain users 4096 Sep  3 10:46 .gnome2  
drwxr-xr-x. 3 teresa domain users 4096 Sep  3 10:42 .gnote  
drwx-----. 2 teresa domain users 4096 Sep  3 10:46 .gnupg  
drwxr-xr-x. 2 teresa domain users 4096 Sep  3 10:42 .gstreamer-0.10  
-rw-r--r--. 1 teresa domain users 142 Sep  3 10:46 .gtk-bookmarks  
drwx-----. 2 teresa domain users 4096 Sep  3 10:42 .gvfs  
-rw-----. 1 teresa domain users  620 Sep  3 10:46 .ICEauthority  
-rw-r--r--. 1 teresa domain users  637 Sep  3 10:46 .imsettings.log  
drwxr-xr-x. 3 teresa domain users 4096 Sep  3 10:42 .local  
drwxr-xr-x. 4 teresa domain users 4096 Sep  3 10:41 .mozilla  
drwxr-xr-x. 2 teresa domain users 4096 Sep  3 10:42 Music  
drwxr-xr-x. 2 teresa domain users 4096 Sep  3 10:42 .nautilus  
drwxr-xr-x. 2 teresa domain users 4096 Sep  3 10:48 Pictures  
drwxr-xr-x. 2 teresa domain users 4096 Sep  3 10:42 Public  
drwx-----. 2 teresa domain users 4096 Sep  3 10:42 .pulse  
-rw-----. 1 teresa domain users  256 Sep  3 10:42 .pulse-cookie  
-rw-----. 1 teresa domain users 1420 Sep  3 10:48 .recently-used.xbel  
drwx-----. 2 teresa domain users 4096 Sep  3 10:42 .ssh  
drwxr-xr-x. 2 teresa domain users 4096 Sep  3 10:42 Templates  
drwxr-xr-x. 2 teresa domain users 4096 Sep  3 10:42 Videos  
-rw-----. 1 teresa domain users 1422 Sep  3 10:47 .xsession-errors  
-rw-----. 1 teresa domain users 2771 Sep  3 10:44 .xsession-errors.old  
[teresa@lpds-13 ~]$
```

Figura 4: A *home directory* num terminal, usando “ls -la”. Na figura 2 apenas estão visíveis os ficheiros cujo nome não começa por “.”, ou seja as pastas Desktop, Documents, Downloads, Music, Pictures, Public, Templates and Videos.

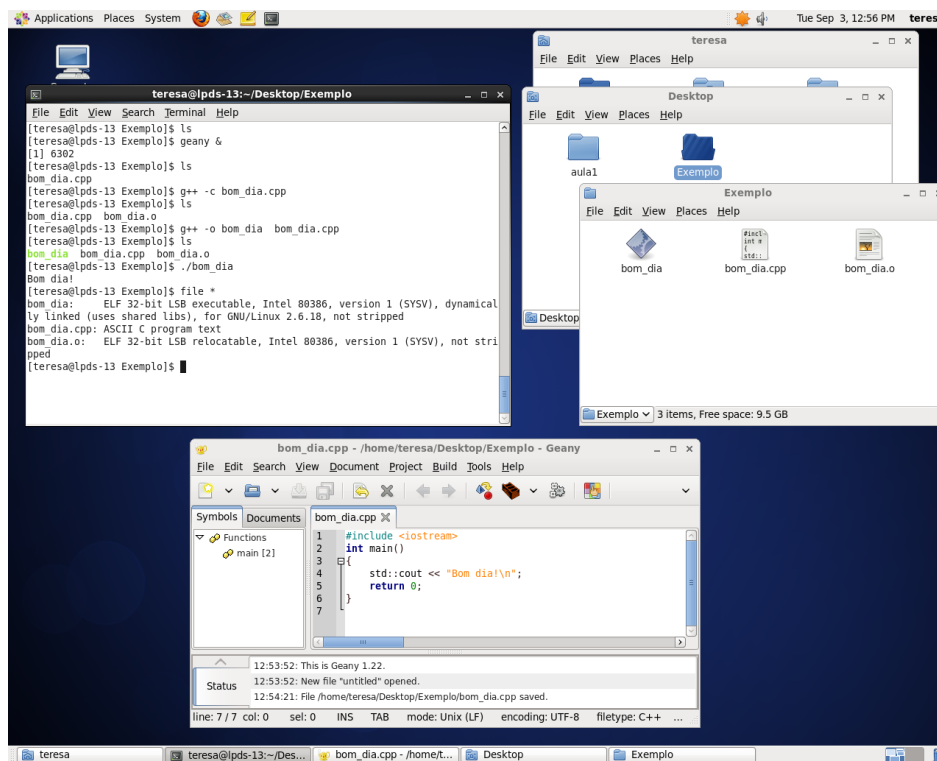


Figura 5: Invocando o geany e compilando na linha de comando.

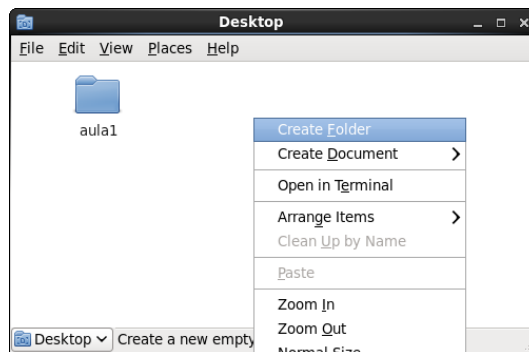


Figura 6: Criação de pasta ou ficheiro

desses avisos são úteis).

Seguidamente, deve ligar o ficheiro objecto com as bibliotecas C++ utilizadas (por exemplo para ler e escrever no ecrã) para criar um ficheiro executável:

```
g++ -Wall nomeficheiro.o -o nomeficheiro
```

ou

```
g++ -Wall -o nomeficheiro nomeficheiro.o
```

O interruptor `-o` indica que desejamos criar um ficheiro executável, cujo nome se encontra imediatamente após o `-o`.

O programa responsável por criar o executável e *ligar* as várias partes de um programa chama-se *linker*. Anteriormente era um programa à parte, mas agora faz parte do gcc.

Após o interruptor `-o` deve vir sempre o nome do ficheiro executável a criar⁴.

Se quisermos **saltar o passo** de criação do ficheiro “objecto” (*object file*) podemos fazer:

```
g++ -Wall -o nomeficheiro nomeficheiro.cpp
```

ou

```
g++ -Wall nomeficheiro.cpp -o nomeficheiro
```

Em projectos pequenos, em que todo o programa está num único ficheiro, esta última opção é mais simples. Contudo, quando surgem muitos erros, há vezes é mais identificar a sua origem se separarmos a compilação da ligação.

Cuidado com a colocação do interruptor `-o`! Se o nome do ficheiro seguinte a `-o` for nomeficheiro.cpp, este é destruído passando a conter o que deveria ficar em ficheiro. Se tiver o nomeficheiro.cpp aberto no editor de texto pode ainda recuperar o ficheiro de texto fazendo Guardar/Save (se não fizer *reload*).

Podemos igualmente compilar vários programas fonte numa só instrução, mas isso será explicado mais adiante na unidade curricular.

3.4 Aceder aos exercícios resolvidos na aula

O PC que utiliza não tem garantia de permanecer imutável entre aulas! Assim o aluno deve sempre no final da aula salvar o trabalho de cada aula numa PEN ou para o servidor **alunos**.

A partir de casa, poderá obter cópias dos ficheiros no servidor **alunos** utilizando o comando **scp**, caso tenha um PC com sistema operativo Linux, ou utilizando o WinSCP a partir do Windows.

O servidor **alunos** não disponibiliza o **g++** nem o editor **geany**.

3.5 Resumindo

Em primeiro lugar, na máquina local, faz *login* como utilizador **aN**, em que N é o seu número de aluno. Cria a directoria da aula corrente na pasta Desktop, assim fica tudo mais visível, como se pode ver na Figuras 5(a) e 7(a). Pode abrir um terminal a partir dessa pasta (ver Figura 7) ou em alternativa abre um terminal e utiliza o comando *change dir* `cd` para se colocar nessa pasta: `cd Desktop/aula1` para ficar na directoria certa quando quiser compilar o programa criado.

Para compilar um programa fonte e criar o corresponde executável tem duas opções:

⁴Se fizermos `g++ nomeficheiro.cpp` é criado por omissão o executável **a.out**, se não houver erros.

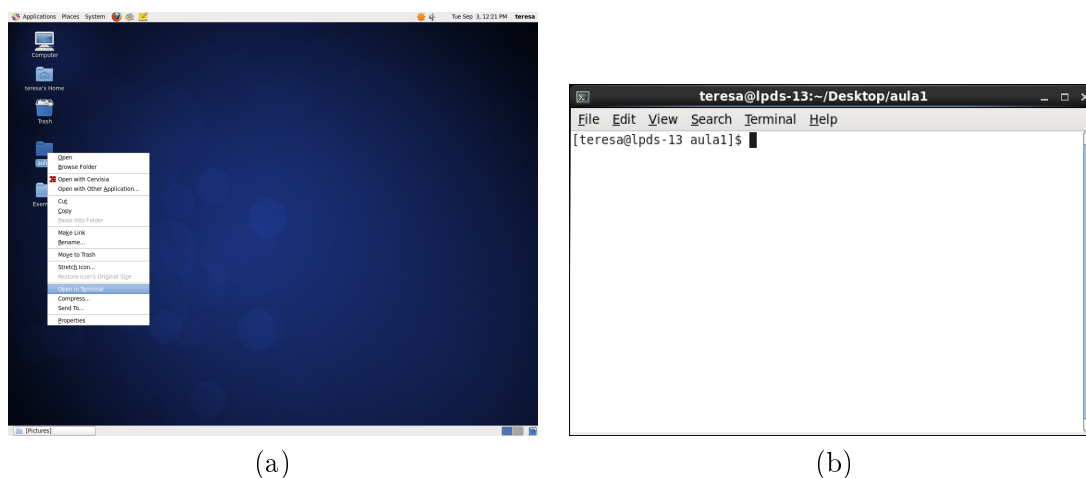


Figura 7: Abrindo um terminal na pasta Desktop/aula1.

- **1ª opção:** Compilar o ficheiro de nome `meuprograma.cpp` fazendo

```
g++ -c -Wall meuprograma.cpp
```

Este comando produz o ficheiro `meuprograma.o`, que é um ficheiro em **código objecto**. Para criar um num programa executável é preciso utilizar um programa (*linker*). Para, chamando o *linker*, ligar o programa com as bibliotecas utilizadas e assim criar o ficheiro executável escreva:

```
g++ -o meuprograma meuprograma.o
```

A linha anterior produzirá o ficheiro `meuprograma`.

Nota: O compilador escreverá sobre o ficheiro cujo nome surge após o interruptor `-o`, mesmo quando esse ficheiro já existe e sem qualquer pré-aviso.

O ficheiro `meuprograma` pode executado a partir da linha de comando.

- **2ª opção:** Compilar e ligar tudo num mesmo passo:

```
g++ -Wall meuprograma.cpp -o meuprograma
```

ou

```
g++ -Wall -o meuprograma meuprograma.cpp
```

Para executar o ficheiro `meuprograma` existente na directoria `~/Desktop/aula1`, basta fazer na linha de comando (assumindo que a mesma está sobre a referida directoria):

```
./meuprograma <return>
```

`./` representa a directoria corrente e `~` representa a *home directory* do utilizador (neste caso deverá ser `/home/Desktop/aN`, com N igual ao número de aluno), ou seja a directoria em que está a linha de comando no terminal.

Em Linux o sistema procura comandos (ficheiros executáveis) em pastas específicas, as quais não incluem as pastas dos utilizadores. Assim para chamar um comando que existe apenas numa pasta de um utilizador é necessário dar o seu nome completo

`~/Desktop/aula1/meuprograma`

ou o seu nome relativo `./aula1/meuprograma` (assumindo se se encontra na pasta `Desktop`) ou `./meuprograma` (assumindo que se encontra na pasta onde está o ficheiro `meuprograma`).

No final da aula não se esqueça de fazer uma cópia do trabalho realizado. Termine a sessão usando o botão no canto superior direito do ecrã (selecione **Quit**), no ecrã seguinte confirme que deseja fazer **logout** e em seguida faça deligue o PC fazendo **shutdown**, para o que deve premir o botão no canto inferior direito do seu ecrã.

4 Criando no seu PC uma réplica do ambiente de trabalho da sala de aula

Caso deseje ter no seu PC um ambiente de trabalho semelhante ao da sala de aula no seu PC, tem as seguintes possibilidades:

- Utilização de uma máquina virtual:
 - Se o seu PC tem sistema operativo Windows, e este tem pelo menos 1 GB de memória pode instalar uma máquina Linux virtual (<http://www.virtualbox.org/>).
 - Vá buscar uma imagem do do sistema operativo, Linux (Centos OS) pronta a utilizar na máquina virtual, no endereço: <http://www2.deec.uc.pt/downloads/centos.zip>, sem senha. Descompacte esse ficheiro. A imagem criada pode ocupar até 10 GB de disco, mas fá-lo de forma dinâmica: ou seja só ocupa o que precisa sendo 10 GB o máximo permitido.
 - Clique no ícone da VirtualBox e para criar uma nova máquina prima o botão (Novo/New) mais à esquerda da barra na superior da VirtualBox e preencha as opções:
 - * nome (MeuCentos)
 - * tipo de sistema (Linux)
distribuição (Other linux)
 - * memória (512MB ou 1GB se a sua máquina tiver pelo menos 2 GB)
 - * Usar disco rígido existente (deve procurar o ficheiro que estava na pasta centos.zip, surge como um cubo vermelho), e já está!

Na imagem que foi buscar já existe um utilizador `user` cuja senha é `q1w2e3r4` que pode executar comandos de administração (como *sudoer*).

Depois é só fazer “Iniciar” e arrancar com o sistema. **Não se esqueça de fazer *shutdown no Linux*, antes de desligar o PC no Windows!**

- Se o seu PC tem sistema operativo Windows, e não tem recursos suficientes para usar uma VirtualBox, pode instalar o Cygwin (<http://cygwin.com/>) o qual é um programa que se instala no Windows, e fornece um ambiente tipo-Linux no Windows. Dessa forma tem acesso ao terminal do Linux no Windows, e pode ter acesso ao g++. Não é preciso "dual boot" ou instalar o Linux, pois o Cygwin funciona ao mesmo tempo que o Windows. Não se esqueça de seleccionar o g++, pois este não faz parte do pacote de programas pré-seleccionados por omissão.
- Possuir um PC com o sistema operativo Windows, pode refazer a partição (manobra sempre arriscada), instalar o Linux e ficar com um sistema "dual boot".
- Possuir um PC com o sistema operativo Linux. Nesse caso basta instalar o **geany** e o **g++**, se ainda não estiverem presentes no seu sistema.

5 Introduzindo alguns comandos Linux

A estrutura de directoria em que a informação está guardada num disco, tem uma organização em árvore.

Alguns dos comandos que pode executar em modo linha de comando são (onde está <texto> significa é obrigatório escrever o parâmetro entre <>):

cd <nomeDir> Muda da directoria corrente para <nomeDir>.

mkdir <nomeDir> Cria a directoria <nomeDir>.

cd <nomeDir> Muda para a directoria <nomeDir> (assumindo que se está da directoria imediatamente anterior).

Se fizer `cd` sem argumentos, volta para a sua *home directory* (aquela em que fica por omissão após **login**).

É possível mudar para qualquer directoria indicando o caminho absoluto (desde a raiz do sistema de ficheiros) ou de forma relativa à posição corrente.

Suponha que a *home directory* é `/home/aN` e que acabou de fazer login. Se fizer `cd Desktop` e em seguida `mkdir aula1` e `mkdir aula2`, criou duas pastas cujo nome completo é `/home/aN/Desktop/aula1` e `/home/aN/Desktop/aula2`, respectivamente.

Para ir para `aula2` faça `cd aula2`. Estando aí, se quiser ir para `aula1` não pode fazer `cd aula1`, porque `aula1` não é uma sub-pasta da pasta `aula2`. Mas pode fa-

zer `/home/aN/Desktop/aula1` (endereço absoluto de `aula1`) ou `../aula1` (endereço relativo de `aula1`, estando em `aula2`).

A pasta corrente é `.` e a pasta anterior (que tem a pasta corrente como sub-pasta) é `..`. Assim se estiver numa pasta e fizer `cd ..` vai para a pasta anterior.

clear Limpa (apaga) o ecrã.

cp `<fichOrigem> <fichDestino>` Copia o ficheiro de nome `<fichOrigem>` para o ficheiro `<fichDestino>`. Se `<fichDestino>` não existe é criado caso contrário será reescrito (se isso for permitido).

mv `<fichOrigem> <fichDestino>` Move o ficheiro de nome `<fichOrigem>` para o ficheiro `<fichDestino>`. Se `<fichDestino>` não existe é criado caso contrário será reescrito (se isso for permitido).

O ficheiro `<fichOrigem>` deixa de existir.

rm `<nomefich>` Elimina o ficheiro de nome `<nomefich>`, sem pedir confirmação!

cat `<nomefich>` Mostra (no ecrã) o conteúdo de um ficheiro de texto.

man `<comando>` Disponibiliza informação sobre `<comando>`.

Mostrar o conteúdo da pasta corrente: `ls` ou em alternativa `dir`
ou ainda `ls -la` para mostrar ficheiros escondidos e informação alargada sobre cada ficheiro:

```
-a,    do not ignore entries starting with .
-l     use a long listing format
```

Recorde que os ficheiros cujo nome começa por `.` são especiais e por isso não são apresentados (por omissão) no *nautilus*, nem no comando *ls*.

Nota final: A avaliação da postura na sala de aula levará em consideração a organização do seu espaço de disco.