

Aula prática #1 – Guião de Familiarização com o Ambiente de Trabalho

Objetivo: Familiarização com o ambiente de desenvolvimento de programas (Ubuntu, Terminal, Visual Studio Code, C, clang).

Nota: O estudante deve previamente obter no CICA a identificação (user ID/ login) e a palavra-chave (password).

Problema 1

1.1 – Aceda à sua conta Linux (Ubuntu) fornecendo a sua identificação (login) e a palavra-chave (password).

1.2 – Após o login, aparecerá o ambiente de trabalho do Ubuntu. Clique no menu principal (botão do canto superior esquerdo) e escreva ou procure pela aplicação “Visual Studio Code”. Execute-a. Repita o procedimento para a aplicação “Terminal”.

Nota: Daqui em diante representamos por “\$” os comandos a ser executados no Terminal.

1.3 – Crie um diretório para os programas que fizer nesta aula, chamado “pro-tp1”:

```
1 $ mkdir pro-tp1
```

Nota: “mkdir” significa make directory.

1.4 – Utilize o seguinte comando para verificar o conteúdo do diretório corrente, certificando-se que o diretório “pro-tp1” foi criado:

```
1 $ ls
```

Nota: “ls” significa list. Existem variantes para este comando como por exemplo “ls -l” que usa um formato de lista longa com mais informação.

A palavra “pro-tp1” deve aparecer listada no ecrã.

1.5 – Mude o diretório do Terminal para o diretório “pro-tp1”:

```
1 $ cd pro-tp1
```

Nota: “cd” significa change directory.

Escreva de novo “ls” e verifique que o diretório “pro-tp1” está vazio (ou seja, que não aparece nada no ecrã aquando da execução do comando).

Problema 2

2.1 – Na aplicação Visual Studio Code, escreva o seguinte programa em linguagem C:

```
1 #include <stdio.h>
2
3 int main() {
4     printf("Hello World!\n");
5
6     return 0;
7 }
```

Nota: Em alternativa ao Terminal, poderá usar o terminal integrado do Visual Studio Code (View ⇒ Terminal).

2.2 – Guarde (File ⇒ Save As) o programa que editou, indicando o diretório do novo ficheiro (Places: “Home/pro-tp1/”) e o respetivo nome (Name: “teste.c”).**2.3** – Liste o conteúdo do diretório “pro-tp1” no Terminal, verificando que nele já consta o ficheiro que acaba de ser criado.**2.4** – Compile o programa que acabou de escrever em teste.c:

```
1 $ clang teste.c
```

Nota: “clang” é o nome do compilador e “teste.c” o nome do ficheiro a compilar.

```
1 $ ./a.out
```

2.5 – Execute o programa e verifique o resultado da execução:

Nota: “a.out” é o nome executável criado no passo anterior.

No ecrã deve aparecer a frase “Hello World!”. O ficheiro a.out é o nome por omissão do ficheiro executável criado pelo clang.

Nota: A compilação do programa também pode ser obtida com o comando “gcc”, uma vez que ambos são compiladores para a linguagem C. Nesta Unidade Curricular será usado “clang” por apresentar os erros de programação de forma mais clara.

2.6 – Compile novamente o programa, desta vez especificando o nome do ficheiro executável:

```
1 $ clang teste.c -o teste
```

Nota: A opção “-o” é uma abreviatura de output. O nome indicado depois dessa opção será o nome do output, ou seja, o nome do novo ficheiro executável.

2.7 – Execute desta vez o ficheiro teste e verifique que se obtém o mesmo resultado que no passo 2.5.

2.8 – Remova o primeiro programa que criou:

```
1 $ rm a.out
```

Nota: “rm” significa remove.

Liste a diretoria em uso e verifique que “a.out” foi removido.

Nota: Usar “rmdir” para remover diretórios vazios ou “rm -r” para remover diretórios não vazios.

2.9 – Para evitar erros, no decorrer da escrita de um programa (como o “teste.c”) ou na execução de comandos no Terminal (como “cd pro-tp1”) há muitas vezes a necessidade de verificar a sua sintaxe. Desta maneira, podemos obter informação sobre um comando através de Man Pages (Manuais de Informação) do próprio computador na aplicação Terminal. Obtenha informação sobre o comando “rm”, escrevendo:

```
1 $ man rm
```

Nota: “man” é um programa que permite ver manuais de programas de unix/funções C/etc..

Navegue pelas instruções da página do manual do comando rm e confirme a funcionalidade do comando “rm -r”, mencionado no passo 2.8 e saia do manual.

Nota: Para sair do manual e regressar ao terminal, clique em “q”, que significa “quit”.

Problema 3

3.1 – Crie um novo programa com o seguinte código e guarde-o como “exp.c”:

```
1 #include <stdio.h>
2
3 int main() {
4     int den, r
5
6     den = 0;
7     r = 10 % den;
8
9     printf("%d\n", r);
10
11     return 0;
12 }
```

3.2 – Compile este novo programa.

```
1 $ clang exp.c -o exp
```

Repare que um dos erros de compilação é:

```
1 main.c:9:10: warning: missing terminating '"' character [-Winvalid-pp-token]
2 printf("%d\n", r);
```

De facto, existe um erro sintático na linha 9. Os erros apresentados pelo “clang” apresentam o formato “nome_ficheiro:linha:caracter”.

```
1 printf("%d\n", r);
```

3.3 – No editor altere a linha 9 para:

Guarde de novo o ficheiro com Save.

3.4 – Compile de novo (no terminal, basta clicar na seta para cima e carregar em “Enter”). O resultado desta vez deverá ser:

```
1      main.c:4:13: error: expected ';' at end of declaration
2      int den, r
```

De facto, na linha 4 falta um “;” a seguir à declaração da variável “r”.

3.5 – No editor altere a linha 4 para:

```
1 int den, r;
```

3.6 – Compile novamente e verifique que não existem erros. Execute o programa novamente. No entanto, o programa não termina corretamente, analise-o e tente perceber porquê.

3.7 – Crie um novo programa com o seguinte código e guarde-o como “input.c”:

```
1 #include <stdio.h>
2
3 int main() {
4     int number;
5
6     printf("Enter an integer\n");
7     scanf("%d", &number);
8
9     printf("Integer entered by you is %d\n", number);
10
11     return 0;
12 }
```

3.8 – Compile o programa. De seguida execute e siga as instruções que aparecem no terminal.

Nota: Com este programa pretende-se mostrar que a execução de programas por vezes esperam a participação do utilizador. Até que o utilizador introduza o conteúdo desejado o programa fica em espera. Note também que não pode executar outros comandos sem terminar esta ação.

3.9 – Copie o ficheiro teste.c para t.c:

```
1 $ cp teste.c t.c
```

3.10 – Liste o conteúdo do diretório pro-tp1.

3.11 – Verifique o conteúdo do novo ficheiro.

```
1 $ cat t.c
```

3.12 – Copie um ficheiro do diretório atual (“pro-tp1”) para um novo diretório dentro deste, com um nome à sua escolha. Por exemplo:

```
1 $ ls
2 $ mkdir pasta-de-teste
3 $ cp teste.c pasta-de-teste/
4 $ ls -R
```



3.13 – Copie novamente o ficheiro para o mesmo diretório, mas com um outro nome. Por exemplo:

```
1 $ cp teste.c pasta-de-teste/teste_copiado.c
2 $ ls -R
```



Problema 4

Uma das funcionalidades do Visual Studio Code é o Live Share, que permite a edição colaborativa de código.

4.1 – Verifique que a versão mais recente do Live Share está instalada:

- Abrir as Extensions  na Activity Bar (do lado esquerdo da janela)
- Pesquisar a extensão “Live Share”
- Aceder às configurações , escolher a opção “Install Another Version...” e escolher a versão mais recente (caso não seja essa a instalada)

4.2 – Inicie uma sessão colaborativa:

- Abrir Live Share  na Activity Bar (do lado esquerdo da janela)
- Um dos elementos do grupo inicia a sessão, abrindo o Live Share  na Status Bar (barra na parte de baixo da janela)
- Fazer login (Microsoft Account), caso ainda não tenha sido feito - usar conta da Universidade do Porto - up2021xxxxx@edu.fe.up.pt¹
- Depois de iniciada a sessão, o endereço da sessão é copiado para o clipboard; enviar esse endereço aos restantes elementos por email ou através de mensagem no Teams
- Os restantes elementos juntam-se à sessão (Join) através do link partilhado

4.3 – Explore as funcionalidades do Live Share no Visual Studio Code implementando juntamente com o seu colega de grupo um pequeno programa onde são lidos dois números inteiros e é apresentada a soma.

Problema 5

Escreva o algoritmo da resolução de uma equação de segundo grau. Com os conhecimentos adquiridos nas primeiras aulas, implemente o correspondente programa na linguagem C. Por exemplo, para a equação ax^2+bx+c , onde $a = 1$, $b = -5$ e $c = 6$, o resultado deverá ser 2 e 3.

¹caso tenha uma conta @fe.up.pt, será necessário ativar a conta Microsoft no perfil do Sigarra <https://www.up.pt/it/pt/servicos/contas-e-passwords/microsoft-office-365-na-uporto/microsoft-office-365-c69ea052>