Ambiente de trabalho

Este guia serve para ajudar a criar o ambiente de trabalho para a realização dos exercícios práticos e do projeto. Para tal são necessárias as seguintes ferramentas:

- Editor de texto. Qualquer editor de texto pode ser usado. Por exemplo: Emacs, vim, VSCode, Notepad++, Gedit, etc.
- **gcc** – Compilador de C e "*Linker*" - O compilador é um programa que transforma o texto dum programa escrito em linguagem C (código fonte) num programa equivalente, em código máquina que pode ser executado pelo processador. O "Linker" é o programa que agrega o código compilado com bibliotecas do sistema operativo para produzir o programa executável autónomo.
- **make** – É uma ferramenta que controla a produção de executáveis a partir do código fonte. Esta ferramenta é particularmente útil para programas com múltiplos ficheiros de código fonte que devem ser compilados individualmente e posteriormente "*linkados*".

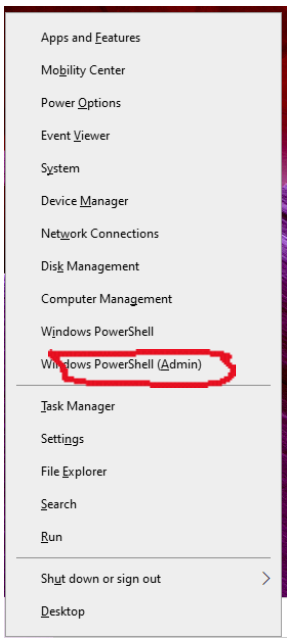
Ambas as ferramentas são abertas e disponibilizadas para Linux (ver ponto 4). Para executar em computadores com Windows é possível utilizar o WSL (Windows Subsystem for Linux). Em alternativa também se pode utilizar uma máquina virtual com uma distribuição de Linux instalada ou o cygwin, ou ainda o Docker.

Windows com WSL

Este guia é para computadores com o **Windows 10 versão 2004 ou posterior (Build 19041 ou posterior)** ou Windows 11 e necessita de uma ligação à rede. (Em computadores com versões anteriores do Windows 10, podem seguir as instruções em https://docs.microsoft.com/en-us/windows/wsl/install-manual)

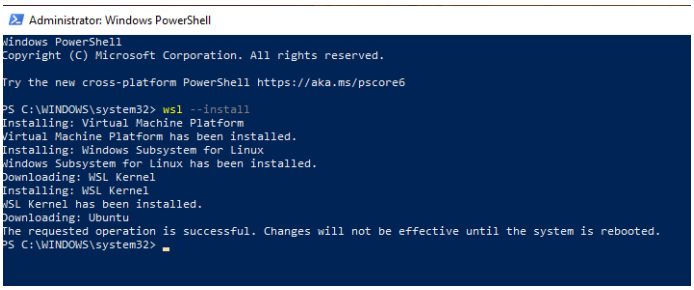
1) Abrir uma linha de comando ou powershell como administrador.

No menu WinX (pressionando as teclas Windows+x) selecionar Windows PowerShell (Admin)



2) Instalar o WSL

Execute o comando "`wsl --install`".



3) Reiniciar o computador

Após reiniciar a instalação continua com o Linux Ubuntu que pode demorar alguns minutos.

Introduzir o UNIX username e definir a password. (Não tem de ser o username nem a password do utilizador do Windows 10)

Na linha de comandos execute "`sudo apt update`" para atualizar o Ubuntu.

4) Instalar o GCC e o Make

Na linha de comandos do WSL execute o comando:

```
$ sudo apt install build-essential gdb make wget ca-certificates
```

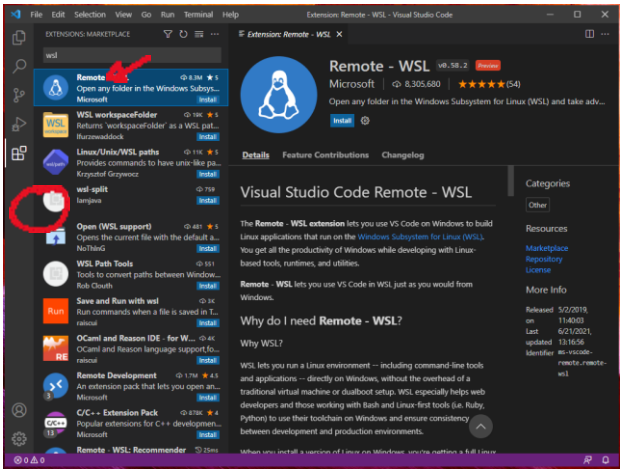
Confirme que o gcc está instalado executando o comando:

```
$ gcc --version
```

5) Instalar e configurar VSCode no Windows

Descarregar o VSCode de https://code.visualstudio.com, aceitar a licença e instalar com a opções "**Add to PATH**" selecionada.

Abrir o VSCode e instalar a extensão "Remote – WSL"



6) Fechar o VSCode e o Ubuntu

Docker

Para usar o Docker, deve-se ir buscar o Docker Desktop. Ver aqui para mais informações:

https://docs.docker.com/desktop/install/windows-install/

Quando o Docker Desktop estiver instalado, basta clonar a imagem "**gcc**" (faz parte do catálogo standard do Docker) e está pronto a correr! (já inclui o **make**). A partir da linha de comando faz-se assim:

```
$ docker pull gcc
```

e depois, para compilar e/ou fazer outras coisas na diretoria corrente, usam-se os comandos:

`$ docker run -v ${PWD}:/code -w /code gcc ...` (para compilar programas)

em que "..." é por exemplo "gcc -o xpto xpto.c" para compilar o ficheiro "xpto.c" e produzir o executável "xpto". Também pode ser um comando tipo "make".

e:

`$ docker run -v ${PWD}:/code -w /code -ti gcc ...` (para executar programas)

em que "..." é por exemplo o comando que ativa o programa que foi compilado anteriormente, i.e. "./xpto". O flag "-ti" assegura que o container (a imagem Docker ativa) vai correr com um *terminal*, i.e. será interativa.

Hello World com GCC e VSCode

Com o ambiente de trabalho preparado. Podemos fazer um pequeno programa para verificar que o compilador e editor de texto estão devidamente configurados.

a) Abrir o Ubuntu no start menu do Windows 10 (para o caso do WSL)
b) Criar uma diretoria para guardar o trabalho e executar o VScode:

```
$ mkdir p1
$ code p1
```

c) No VSCode abra uma janela de terminal (Terminal->New Terminal).
d) Crie uma diretoria chamada “ap1” (para "aula prática 1")
    a. No terminal: "mkdir ap1" e "cd ap1" (para entrar nessa diretoria)
    b. No menu de contexto no painel Explorer (botão direito do rato nesse painel):
e) Dentro da pasta ap1, criar um novo ficheiro chamado “hello.c” com o seguinte conteúdo:

```
#include <stdio.h>
#include <stdlib.h>

int main (int argc, char *argv[]) {
    printf ("Hello world!\n");
```

```
}
```

f) Compilar, linkar e executar o programa hello.
(certifique-se que a linha de comando se encontra na diretoria "ap1")
Compile o código fonte, vai produzir código objeto parcial.
`$ gcc -c hello.c`
Linkar o programa e criar o executável hello
`$ gcc -o hello hello.o`
Os dois comandos anteriores podem ser substituídos por um único:
`$ gcc -o hello hello.c`
Executar o programa
`$ ./hello`

g) Criar uma Makefile para sistematizar a compilação.
Criar um ficheiro chamado **Makefile** com o seguinte conteúdo (as linhas precedidas com # são comentários e não necessitam de ser copiadas). Cuidado que as linhas que parecem começar com espaços em branco, na realidade começam com o caráter "TAB", i.e. "Control-I", que se manifesta por vários espaços em branco.

```
# makefile para o Hello

all: hello

hello: hello.o
	$(CC) -o $@ $^

clean:
	rm -f *.o hello

%.o: %.c
	$(CC) -c $^
```

No terminal executar o comando `make clean` para eliminar os resultados da compilação anterior.

Compilar novamente com o comando `make`.

Executar o programa compilado com o comando:

`$ ./hello`.

Se executar o comando `make` novamente, nada será compilado pois o código fonte não tem alterações.

Exemplo de sessão:

```
19:23:26$ ls -l
total 8.0K
-rw-r--r-- 1 ds staff 157 Sep 18 19:13 Makefile
-rw-r--r-- 1 ds staff 109 Sep 18 19:09 hello.c
19:23:29$ make
gcc -c hello.c
gcc -o hello hello.o
19:23:36$ ls -l
total 48K
-rw-r--r-- 1 ds staff 157 Sep 18 19:13 Makefile
-rwxr-xr-x 1 ds staff 33K Sep 18 19:23 hello
-rw-r--r-- 1 ds staff 109 Sep 18 19:09 hello.c
-rw-r--r-- 1 ds staff 800 Sep 18 19:23 hello.o
19:23:39$ ./hello
Hello world!
19:23:42$ make clean
rm -f *.o hello
19:23:44$ ls -l
total 8.0K
-rw-r--r-- 1 ds staff 157 Sep 18 19:13 Makefile
-rw-r--r-- 1 ds staff 109 Sep 18 19:09 hello.c
19:23:46$
```