ATAD | Software, Alternative Methods

These options should be used only if you have exausted the methods described in the Software.pdf document, namely for Windows and, particularly, MacOS.

They are presented in preferred order:

1. Docker container;
2. Virtual Machine, or;
3. CodeBlocks and MinGW (only for school workstations).

1 | Docker container

This method will give you a containerized development solution and, if you have success, will give you a complete workflow using VS Code and the complete development toolchain (gcc, gdb, valgrind and doxygen).

1. Follow the **Docker Desktop** installation for your operating system:
    - https://docs.docker.com/desktop/install/windows-install/
        - **Note:** In Windows you'll still need to enable WSL2 beforehand. See step 1 at Software.pdf (section Windows/WSL or Linux)
    - https://docs.docker.com/desktop/install/mac-install/
2. Open a terminal and pull the `brunomnsilva/docker-atad` image:

```
$> docker pull brunomnsilva/docker-atad:latest
```

3. Go to step 11 at Software.pdf (section Windows/WSL or Linux | Test the environment)

High memory usage on Windows?

If you are experience heavy RAM usage on Windows, this is due to the default behavior of memory allocation of the WSL2/docker backend (e.g., may utilize 80% of available RAM memory, which is way too much for our needs)

If that is the case, create a `.wslconfig` file in you user directory, e.g., `C:\Users\Bruno Silva` with the following contents:

```
[wsl2]
memory=2GB
```

Adjust the maximum memory according to your resources.

2 | Virtual Machine

Using a virtual machine will give you a virtualized Linux operating system (guest) on your main operating system (host).

This solution may be heavier on resources than the previous method.

1. Follow the instructions at https://ubuntu.com/tutorials/how-to-run-ubuntu-desktop-on-a-virtual-machine-using-virtualbox#1-overview
2. Follow the instructions from Software.pdf, starting at step 6 (Windows/WSL or Linux)

> The `Shared Folders` functionality is higly advised, allowing you to keep all your projects in your main operating system (host) filesystem.

3 | CodeBlocks and MinGW

This is the last option, since you'll not be able to work the necessary development methodology, IDE and testing. **You'll be left working without**:

- *Git*;
- *Valgrind*;
- *Makefiles*, and;
- All the integration with VS Code.

However, **it will be the option reserved for the school workstations to perform the assignments**, if you don't have a personal laptop.

> This is because WSL requires individual image installation for each student account and the school computers simply don't have the necessary disk space to accommodate multiple installations.

Follow the following steps:

1. Install the CodeBlocks IDE + Compiler from https://www.codeblocks.org/downloads/binaries/
    - Make sure you are using the ***setup*** version, e.g., `codeblocks-20.03mingw-setup.exe`.
    - Perform a "Full" instalation, when given the option in the installer.

Working with git repositories

When asked to use a *template* **git repository** for a given assignment, you should perform the following steps:

Create the `ATAD` folder in your personal documents folder.

1. In the repository's GitHub page, use the `Code > Download ZIP` option, and save it to your disk; extract the archive;
    - Make sure you have a single folder with all the source code; if not, move the source code to the parent directory.
    - You should be left only with something like:

```
- ATAD
    - Lab1_Template-master
        - main.c
        - input.c
        - input.h
        - ...
    - Lab2_Template-master
        - ...
```

2. Open *CodeBlocks* and create a new project: `File > New... > Project`;
3. Follow the project configuration steps in the following order carefully:

1. Select `Console application` in the wizard;
2. Select `C` language in the next step;
3. **First, select the project location**: it should be folder `ATAD` (see above);
4. Second, **name you project with the same name as the folder you got from step 1**, e.g., `Lab1_Template-master`;
5. When asked to confirm the "OVERWRITE" of the `main.c` file, **SAY NO**!

> This will ensure that the `*.cbp` file (codeblocks project file) is within the existing source code folder.

4. We now must add the existing files to the code blocks project (it doesn't add them automatically):
    - Select `Project > Add files recursively` and select the project folder, e.g., `Lab1_Template-master`.
5. You should be set. Open the `main.c` file and run the project with `Build > Build and run (F9)`.
    - Proceed with the assignment.
    - To add new modules, i.e., `*.h` and `*.c` files, use `File > New... > File...`.

Author and support

Bruno Silva (bruno.silva@estsetubal.ips.pt)

You should ask your PL teacher for any help regarding these contents and procedures.