ATAD | Makefiles

A makefile is a file (by default named "makefile") containing a set of directives used by a make build automation tool to generate a target/goal.

Whether your program consists of one or multiple source files, the process is relatively straigthforward. Inside of your working directory you must create a file (without extension) named `makefile`. If you wish to accomplish this using the CLI, use the `$> touch makefile` command.

You should obtain a file structure like this:

```
- <WorkingDirectory>
    - main.c
    - time.c
    - time.h
    - makefile
```

Then you must edit the makefile. You can provide any number of directives to use later; a directive consists of a named sequence of commands. Example of a makefile contents:

```
default:
    gcc -Wall -o prog main.c time.c
debug:
    gcc -Wall -o prog -g  main.c time.c
clean:
    rm -f prog
```

You can see three directives, i.e., `default`, `debug` and `clean`. You can choose any names, as long as they are meaningful. The `default` directive is the first, and therefore the "default". You can invoke these directives by using the `make` command:

```
$> make
```

This will execute the default directive, namely the `gcc -Wall -o prog main.c time.c` command.

If you instead run:

```
$> make debug
```

This will execute the `debug` directive, namely the `gcc -Wall -o prog -g main.c time.c` command.

Finally, if you execute:

```
$> make clean
```

This will delete the executable file `prog`, if it exists.

Author and support

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

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