Aeronautics Institute of Technology — ITA
Computer Science Division — IEC

CSC-33 Operating Systems
Prof. Lourenco A Pereira

The mini-shell project
March 28, 2023

This work is about the implementation of a miniature shell. When started, it shows a prompt command. Example:

```
Welcome to the miniature-shell.
cmd> _
```

The user must provide a full path executable file name corresponding to the command the shell will spawn. Your mini-shell implementation must use: `fork(2)` to create a new process; `dup2(2)` to change the new process' standard input (`stdin`) and output (`stdout`); `execv(3)` to load the binary image into the memory and execute it. The parent process (in this case, mini-shell) must wait for the child process (command triggered by the user) to finish with the `wait(2)` function.

The mini-shell has two redirect commands:

- `>`: the process' stdout will be a file. E.g., `prog > out.txt`, where all the process' output go to `out.txt` file;
- `<`: stdin read from a file. E.g., `prog < in.txt`, where all input data will be read from the `in.txt` file.

You must pay special attention to high-level functions. `system(3)` is not allowed because it hides many OS system call interactions, which we do not want. Instead, `fork(2)` and `execv(2)` are in the manual section two. This means using system-call functions as much as possible. Sticking to that guidance (preference for section two functions and system calls) will keep you on the right path.

Attention:

1. Use DC (*disciplina consciente*) and do the mini-shell project by yourself. No sharing solutions allowed. You can brainstorm the problem and strategies to tackle it, whereas our guideline forbids sharing code and other artifacts.
2. Mandatory use of C language. Provide a `makefile` to compile your mini-shell.
3. Deadline: April 28, 2023. (submitted in classroom)

Some valid inputs to mini-shell:

```
./prog arg1 arg2 argn
/bin/ls
/bin/cat test.txt
/bin/cat test.txt > out.txt
./prog < in.txt
./prog1 < in.txt > out.txt
```