Operating Systems
Programming Project

1. Scope and goals

The goal of the programming project is to implement an **alternative Linux shell in C**. The shell is the part of the operating system that interacts with the user either via command line or a graphical UI. Our shell will be called **imcsh** (short for IMC shell) and will accept commands from the user using the standard input. The shell will execute the commands and return the expected result to the user.

Important: It is **not allowed** to call the standard Linux shell to implement the commands!

2. IMCSH Description

After executing the shell, a command prompt of the form "`user@host>`" will be shown on a new line in the terminal.

The imcsh shell will accept the following syntax:

- `exec <program_to_execute>:` Accepts a program to execute with its parameters and executes it by creating a child process. The program to execute will be a string eventually with blank spaces between the single parameters of the program to call. After termination of the child process, the shell will output the PID of the just terminated process in a new line.
  - Example: `exec ps -uax` will execute the command "ps -uax"
- Modifier &: If this modifier is used at the end of an exec command, imcsh will execute the command in the background and continue immediately. As soon as the process ends, it will output the PID of the just terminated process in a new line.
- `globalusage:` This internal command will show a string with details on the version of imcsh being executed.
  - Example: `globalusage`
    - Output: `IMCSH Version 1.1 created by <author(s)>`
- Modifier >: If this modifier is used at the end of a command, imcsh will redirect the output as text to the file given after the symbol. If the file does not exist, the shell will automatically create it. If the file exists, the shell will try to append the new data to the end of the file. In this case, the only visible output in the console will be the PID of the process after termination.
  - Example: `exec ls -l > directory_output.txt`
  - Example: `globalusage > usage.txt`
- `quit:` Quits the shell. If there are any running processes, the following question will be prompted to the user: "`The following processes are running, are you sure you want to quit? [Y/n]`". A list of all running processes will follow. If the user enters Y, the shell will quit and all running processes will be cancelled.

3. Delivery

This part of the project will be completed in groups of two people.

Delivery date: December 23st via MS-Teams.

Submission contents:

- Implementation in C including makefile
- Technical documentation, including
  - Introduction
  - Description of the source code

All files will be enclosed in a single zip file.