
# REPORT PRACTICE 1
## OPERATING SYSTEMS


| | |
|---|---|
| Denis Loren Moldovan | 100522240 |
| Antonio Nicolas Lemus Yeguas | 100522110 |
| Jorge Adrian Saghin Dudulea | 100522257 |

# INDEX

# Code Description

All the functions and implementations written during the development of this practice have been properly commented in their corresponding files, explaining the arguments and return outcomes of the previously mentioned.

## *crear.c*

The functionality of this program is entirely written inside the main function as it is a really simple program. The UMASK in this process is changed to 000 in order to comply with the argument passed to the program.
Because there may be folders with a predefined ACL, we added a safeguard at the end of the program, where we used the function chmod() to change, yet again, the permissions of the newly created file with the one passed as argument to the program.

## *combine.c*

The execution of this program is divided into a series of functions, abstracting the whole implementation seen inside the main function, while also making it easier to read.

- **create_csv**: this function creates the file containing the number of students with each mark and calculates the percentage of students with each mark.

- **fetch_alumno**: this function gathers all the data from the files and stores them in arrays to operate with them later.

- **join_alumnos**: this function performs a bubble sort algorithm to the array in order to sort them in ascending order.

- **classify_alumnos**: this function makes the counting of how many students have each type of mark, to later call **create_csv**, so the arguments are correctly parsed.

- **output_new_data**: this function writes in the output file provided all the students received and the name of the output file.

# Test cases

| Test ID | Description | Inputs | Expected outputs | Outputs |
|---|---|---|---|---|
| *TCr1* | Create a file that does not exist and give valid permissions. | "./crear test1 0666" | test1 created with permissions: rwx-rwx-rwx | test1 created with permissions: rwx-rwx-rwx |
| *TCr2* | Create a file that already exists. | "./crear test1 0666" | Raise error: "File exists" | Error: "File exists" |
| *TCr3* | Create a file that does not exist without giving permissions | "./crear test2" | Raise error: "Wrong usage of program" | Error: "Wrong usage of program" |
| *TCr4* | Create a file that does not exist but giving wrong permissions | "./crear test2 XXX" | Raise error: "Wrong format of the permissions" | Error: "Wrong format of the permissions" |
| *TCo1* | A normal execution where both student lists are valid and a successful program execution | "./combine f1.dat f2.dat test5" | test5 and estadisticas.csv created successfully | test5 and estadisticas.csv created successfully |
| *TCo2* | Merge a valid students list with an empty one | "./combine f1.dat f6.dat test5"<br>PD: f6.dat is empty | Creates test5 empty and estadisticas.csv with everything equal to 0.00% | Creates test5 empty and estadisticas.csv with everything equal to 0.00% |
| *TCo3* | The combined length of the input lists is bigger than 100 | "./combine f2.dat f3.dat test6" | Raise error: "Students limit surpassed" | Error: "Students limit surpassed" |
| *TCo4* | Give a wrong number of arguments | "./combine f2.dat test6" | Raise error: "Wrong number of arguments" | Error: "Wrong number of arguments" |
| *TCo5* | The output file already exists | "./combine f1.dat f2.dat test1" | Raise error: "Error creating new file" | Error: "Error creating new file" |
| *TCo6* | Giving a list in a wrong format | "./combine f1.dat f2.pdf test6" | Raise error: "Invalid format inside input file" | Error: "Invalid format inside input file" |

# Conclusion

In conclusion, this lab has taught us how to use the most basic system calls and has also instructed us to solve some errors encountered during the development of the code.
For instance, we started developing a merge-sort algorithm for the array of students but due to its complexity, we decided to implement the bubble sort one, which was almost a quarter long.
In addition, we also had to solve some warnings that the compiler was giving us, like the way we used constant parameters or even unused variables that we had lying inside the code.