Mandatory part

The command './pipex file1 cmd1 cmd2 file2' must behave like this command: '< file1 cmd1 | cmd2 > file2'

Error and arguments management

- The program takes 4 arguments, no more, no less (except for bonus part) and only in the required order.
- Error management is correct: (un)existing files, files rights, (un)existing command binary, and so forth.

If these points are successfully passed, check 'Yes' and continue the evaluation process. Otherwise, the evaluation is over. Use 'Incomplete work' or any other appropriate flag.

Yes | No

The program

The program does what the subject requires and doesn't display more information/steps than the shell command it should replicate.

Run your own tests and compare the program results against the original shell output. Take a look at the subject examples if you need to.

If no error happens, check 'Yes' and continue. Otherwise, the evaluation process ends now.

Yes | No

Bonus part

Evaluate the bonus part if, and only if, the mandatory part has been entirely and perfectly done, and the error management handles unexpected or bad usage. In case all the mandatory points were not passed during the defense, bonus points must be totally ignored.

Multiple pipes

The program manages the usage of several pipes one after another.
As for the mandatory part, test with shell commands then compare with program output.

If it works with 2 pipes only in the same command but fails with 5 pipes, the bonus is not passed.

Yes | No

<< and >> with here_doc parameter

The program replicates the use of << and >>.

Test multiple times something like:
'CMD << STOP_VALUE | CMD1 >> file1'

Yes | No

Preliminary tests

If cheating is suspected, the evaluation stops here. Use the "Cheat" flag to report it. Take this decision calmly, wisely, and please, use this button with caution.

Prerequisites

- Defense can only happen if the evaluated student or group is present.
This way everybody learns by sharing knowledge.
- If no work has been submitted (or wrong files, wrong directory, or wrong filenames), the grade is 0, and the evaluation process ends.
- No empty repository (= nothing in Git repository).
- No Norm error.
- Cheating (= -42).
- No compilation error. Also, the Makefile must not re-link.

If all of these requirements are passed, check 'Yes' and continue the evaluation process. Otherwise, use the appropriate flag at the end of the scale!

Yes | No

General instructions

General instructions

- If a crash or unexpected error occurs (segmentation fault, bus error, nonsense display, and so forth), use the flag 'Crash'!
- The Makefile compiles the executable with the required options.
- The executable is named 'pipex'.
- No forbidden function is used.

Yes | No