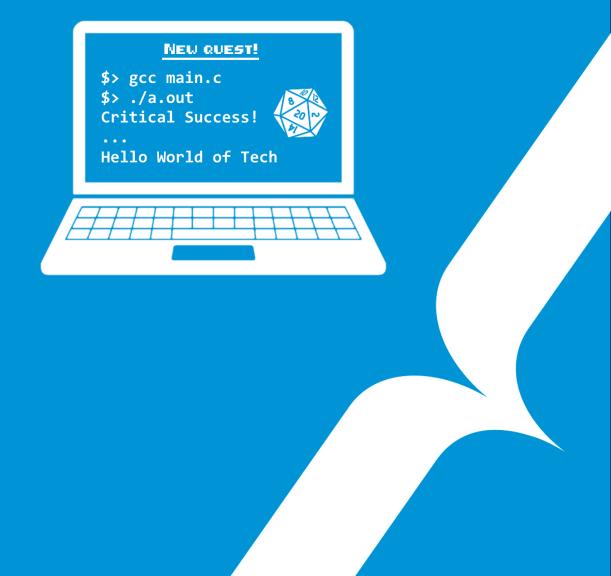
{EPITECH}

DAY 12 FILE DESCRIPTORS



DAY 12

Preliminaries



Language: C

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.
- Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).



If you use your libmy for the following tasks, it must be built (if required) from the task's Makefile.

The prefered way to do this is to call the lib's Makefile from the project's Makefile.



Allowed system function(s): open, read, write, close



Task 01 - Cat

Delivery: cat/*

Write a program called cat, which executes the same tasks as your system's cat command. You do not have to handle options.

There is an unlimited number of files given as parameter.

cat without parameters must be supported.

You must deliver a Makefile with the following rules: all, clean, fclean and re, and it must not relink. The binary's name must be cat.

You may use the errno variable (refer to man errno), but perror and malloc are prohibited.

This task can only be performed by declaring a fixed-size array. It will have a limited size of approximately 30 ko. In order to test the limitation, use the command limit in your shell:





Limit is an internal feature of a specific shell. Find the good one :D.

man cat



Task 02 - Testing cat

Delivery: tests/* cat/Makefile

You must now write unit tests for all the code composing your "cat" program. We expect that most of your functions will be tested (so don't only test the final results).

Your tests will be built and executed with a rule tests_run that you have to add to your previous Makefile. But your tests files need to be in a directory called "tests" **at the root** of your repository. See how_to_write_unit_tests.pdf for more informations.



If you need to open a file (to obtain a file descriptor) for multiple tests, we encourage you to use fixtures to avoid code duplication.



Task 03 - grep

Delivery: grep/*

Write a program called grep, which executes the same tasks as your system's grep command. You do not have to handle options.

There is an unlimited number of files given as parameter.

You must deliver a Makefile with the following rules: all, clean, fclean and re, and it must not relink. The binary's name must be grep.

You may use the errno variable (refer to man errno), but the perror function is prohibited.



For this task, and this task only, malloc and free are allowed.



You don't have to handle regex. We are only asking for a simple matching system.

man grep



The read of size 1 is forbidden.



Task 04 - Testing grep

Delivery: tests/* grep/Makefile

You must now write unit tests for all the code composing your grep program. We expect that most of your functions will be tested (so don't only test the final results).

Your tests will be built and executed with a rule tests_run that you have to add to your previous Makefile. But your tests files need to be in a directory called "tests" **at the root** of your repository. See how_to_write_unit_tests.pdf for more informations.



If you need to open a file (to obtain a file descriptor) for multiple tests, we encourage you to use fixtures to avoid code duplication.



#