

2020/2021: Sistemas Operativos (*Operating Systems*) - 2º MIEIC

8.feb.2021

Exercise Sheet 1

Basics of Operating Systems and Unix programming

1. «*The make program is an interpreter of special programs (makefiles), that can only be used in Linux and just for compiling C source code.*»
 - a. Is this true?
 - b. The compilation lines of the *makefile* used in the "Interesting examples of compilations" have the option `-Wall`. Write a simple program that shows the importance of using that compiler option.
 - c. Prepare a *makefile* for easing the compilation of the programs suggested in this Exercise Sheet.
2. Present both an advantage and a disadvantage of *dynamic linking* programs over *static linking*.
3. Study the man page of the service `atexit`.
 - a. Could this service be useful when a running program is suddenly terminated (e.g. by `CTRL-C`)?
 - b. Try it with a slightly adapted code presented in the man's `EXAMPLE` section.
4. *[Study the implementation of the `read()` system call, trying to see if it is compatible with the scheme presented in the classes' sheets. For that, write a mini-program with just that system call, compile it perhaps with static linking and inspect the disassembled code of the executable with `objdump` (man `objdump!...`).]
5. What are "command line arguments" and "environment variables"?
 - a. Write a program that outputs to the screen its command line arguments and the name and value of all the shell's environment variables.
 - b. Change the program so that its output is just the value of the environment variables whose names are passed as arguments (e.g. `prog PATH HOME` would show the values of `PATH` and `HOME`). Confirm that the values output are correct using the shell commands `echo` and `printenv`.
 - c. From the command line create a new environment variable, `DIR1`, with value `123-testing` and use the program you wrote in b. to confirm the existence and value of `DIR1`.
6. A system call may fail; by words, exemplify that with `open`. (If necessary: man 2 `open` !)
 - a. Write a program that tries to open a file named in the command line argument and report the "open" failures without using `perror`. With the program prove the example of failure you gave above.
 - b. Modify your program to ensure that failure messages are output to the *standard error* (`stderr`) and not to the *standard output* (`stdout`). Confirm your new program behavior with *shell* redirection. (Clue: `prog > ofile 2> efile`)
 - c. Repeat a. using `perror`. Test `stdout` and `stderr` for the output of this program.
7. Explain the two main functions of an operating system.
8. Based on the output of the command `ps aux` say if Linux is
 - i. multi-programming

- ii. multi-user
 - iii. time sharing
 - iv. (cannot say just by the output)
9. Some system operations can be performed while in *user mode*, others only in *kernel mode*.
- a. using the `date` command classify, in the above respect, the read and write operations on the internal clock. (Clue: `date -s "20220913"`)
 - b. name one (more) operation that can only be performed in *kernel mode*.
10. Write a program that outputs "Hello world!" 100 000 times and, before ending, presents the duration time of the running and of the processor usage (both in user and system modes). (Clue: `"time ls -R /tmp"`.)
11. Think of a program that copies the content of a file to another file, both named as program arguments (e.g. `prog fname1 fname2`). Write such a program using:
- a. mainly (direct) system calls, such as `open`;
 - b. mainly C library calls, such as `fopen`.
 - c. Change one of those programs so that when the 2nd argument is missing (`fname2`, in the example above) the content of `fname1` is shown on the screen (i.e. `stdout`).
12. Think of a program that writes to the *standard output* the name and size of all the regular files in a directory whose name is given as argument (e.g. `prog dirname`).
- a. Write such a program, also assuring one output line per file.
 - b. From the output of the program and using `wc` show the number of regular files in the directory.
13. ... (additional exercises on Moodle for people with free time)
- * just for the most curious... :-)