LSV / ENS de Cachan
Architecture et Système

2014/2015
Feuille de TP n° 10

TP10

The course homepage is here:

`http://www.lsv.ens-cachan.fr/~schwoon/enseignement/systemes/ws1415/`.

You will find the slides and demonstration programs from the course and some other files for the exercise there.

Details of shell commands and C functions can be obtained by using the `man` command.

1 Copying files

The goal of this exercise is to write a simple program that copies one file into another (like `cp`). A skeleton program can be found on the webpage of the course.

You will need functions like `open`, `creat`, `read`, `write`, `close` (all in section 2 of the man pages).

To find the size of the file, there are three possibilities: (i) check what values `read` returns; (ii) use `lseek`, (iii) use `stat`.

1. Write a basic version of the program. Make it failsafe by checking whether the operations succeed and printing an error (there is a function `syserr` in the skeleton for this purpose).
2. How many bytes does your program read and write at a time? How does this affect the execution time? Use a large file and the `time` command in the shell to test.
3. For those who advance more quickly: Make the destination file preserve the access rights and the modification time of the source file (see `stat`, `chmod`, `utime`).

2 Pipes (simple exercise)

Let us make two processes communicate with a pipe. As a first simple demonstration, write a program that uses a pipe (`pipe`) between parent and child.

1. Write a program that forks into a parent and a child. The parent then reads input on the console, sends it through the pipe, and the child prints it.
2. Assure that the child terminates correctly. More precisely, when the user presses Ctrl+D on the console (signalling end-of-file), the parent should close its end of the pipe, the child should notice this and terminate, too.
3. Advanced exercise: Make the program print `zzzz...` when the user hasn't typed anything for five seconds. (Several solutions are possible.)

3 Database

We will now use pipes to interact with shell commands, in particular to give input to and read output from other programs.

On the webpage, you will find a zip with the file `dep.sq`. It is a database of the French departements. Running `sqlite dep.sq` in the shell will give you an input prompt. Typing `"SELECT name,cheflieu FROM dep WHERE nr='07';"` will tell you that the name of the departement is Ardèche and its capital Privas.

The zip file also contains a skeleton program, which creates a child that execs `sqlite`. Extend it so that parent and child communicate through the pipe; the parent lets the user input a department number, passes the appropriate query to the child, and then outputs the result according to the pattern

```
name = Ardèche
chef-lieu = Privas
```

Notes: You will need two pipes to communicate both ways. Useful man pages: pipe, dup2, execvp