TP 07 : Signals and a few helpful tools

1 Signals in ANSI and POSIX

Write a program with a parent process that
— creates a child process
— transmits a sequence of keyboard-entered bits (the user enters 0's and 1's), bit by bit, to this child process.

The child process must display the sequence in the correct order as received from the parent. To do this, use signals. The difficulty with this exercise is that the order is not guaranteed : a signal A sent before signal B by the parent process can be received in the reverse order by the child process. Take care to test several patterns of sequences (alternations of 0's and 1's, successions of 0, successions of 1). To enter the bits, you can use

```
for (c = 0; c != 0 && c != 1 && c != EOF; c = getchar());
if (c == 0) // Do something
if (c == 1) // Do something
if (c == EOF) break;
```

1. First implement with the ANSI C API (kill, pause, signal functions). What issue do you face ?
2. Improve your program by using POSIX.

2 Introduction to a few useful tools

Here are some resources to learn about tools which will hopefully be useful to you at some point. There are a lot of them, don't hesitate to store the links somewhere and come back to them when you need it !

2.1 Text editors

— emacs :
  — To learn how to use emacs, you can use the emacs tutorial avaliable within emacs itself. Use the command `emacs` and press F1, then t. Enjoy the reading afterwards. You can also check out `http://www.jesshamrick.com/2012/09/10/absolute-beginners-guide-to-emacs/`
  — The emacs manual : `https://www.gnu.org/software/emacs/manual/pdf/emacs.pdf`
  — An introduction to emacs-lisp : `https://www.gnu.org/software/emacs/manual/pdf/eintr.pdf`
  — A keyboard shortcut spreadsheet : `https://www.gnu.org/software/emacs/refcards/pdf/refcard.pdf`
  — For those who want to customize their emacs further, you can check out frameworks like spacemacs or doomemacs which gather a lot of sane defaults in one place.
— Vim :
  — A tutorial is also shipped with vim, type `vimtutor` in the terminal. `http://www.openvim.com/` proposes an interactive tutorial.
  — Vim's documentation is really helpful, use `:help foo` to your heart's content.

  — http://vim.rtorr.com or http://vimsheet.com for keybindings cheatsheets
  — A cheatsheet for vimscript https://devhints.io/vimscript

2.2 Word replacements

— LaTeX :
  — Quick tutorial from scratch : https://www.overleaf.com/learn/latex/Learn_LaTeX_in_30_minutes
  — More in-depth tutorial : https://zestedesavoir.com/tutoriels/826/introduction-a-latex/
  — Packages documentation : https://ctan.org
  — Online editor : https://www.overleaf.com
— Typst (a bit new and experimental but really cool) :
  — Website (and online editor) : https://typst.app/
  — Tutorial : https://typst.app/docs/tutorial/

2.3 RegExp

— Tutorials : https://regexlearn.com/learn/regex101 https://regexone.com/
— Cheatsheet : https://remram44.github.io/regex-cheatsheet/regex.html

2.4 git (version control)

— An interactive tutorial : https://learngitbranching.js.org
— Cheatsheets for git commands : https://ndpsoftware.com/git-cheatsheet.html https://training.github.com/downloads/fr/github-git-cheat-sheet.pdf
— The git book : https://git-scm.com/book/fr/v2/
— Upstreams you (easily) have access to : https://github.com, https://gitlab.com or https://gitlab.crans.org