

### **B2 - Unix System Programming**

B-PSU-200

# Bootstrap

navy





## Bootstrap

**binary name**: process\_info, kill\_it, who\_sig\_me, signal\_me repository name: PSU\_navy\_bootstrap\_\$ACADEMICYEAR

language: C

compilation: via Makefile, including re, clean and fclean rules



• 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 (O if there is no error).

This Bootstrap is a introduction to the way signals function in a Unix system.

The Unix kernel "informs" processes through signals in order to transmit potential problems (SIGSEGV for a segmentation error for instance).

Each signal is assigned a default behavior.

Read the related man pages in order to understand each signal's procces' behavior.

Numerous system calls exist to handle signals.

We will purposely not cover a large part of them, so that you can have fun researching them yourself.

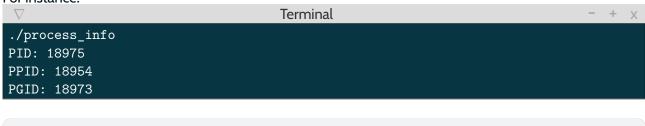


### **EXERCISE 1**

Write a program named process\_info

that displays the following process information: process ID, parent process ID and process group ID.

For instance:





man getpid

#### **EXERCISE 2**

Write a program named kill\_it that sends the SIGQUIT signal to the process whose PID is passed as parameter.

For instance:





kill man page. In order to test it, use the ps program to obtain the PID of a program (firefox, xeyes etc.)





#### **EXERCISE 3**

Write a program named who\_sig\_me

that, for each received signal, displays its name and the PID of the emitter process.

The program takes the list of signals to be rerouted as parameter. If a signal's rerouting fails, an error message is displayed.

#### For instance:

```
Terminal - + x

~/B-PSU-200> ./who_sig_me 12 9 > stdout.log &

[2] 1590

~/B-PSU-200> kill -USR2 1590

~/B-PSU-200> kill -KILL 1590

~/B-PSU-200> cat -e stdout.log

Unable to handle Killed signal

Signal User defined sigal 2 received from 1591
```



sigaction and strsignal man pages. Scan the system's .h in order to understand the number that is associated with each signal.





### **EXERCISE 4**

Write a program named **signal\_me** that counts the number of times it receives the SIGUSR1 and SIGUSR2 signals.

This program must display a summary when it receives the SIGQUIT signal, before exiting.