Signals
Did you ever?

Did you ever press Ctrl-c to terminate a program?

What happens when pressing Ctrl-c?
A so called TERM signal is sent to the process, which by default terminates the process (the program).

$ date ; sleep 100
Mon May 14 22:41:58 CEST 2018
^C
[hesa@bartok ~]$ date
Mon May 14 22:42:00 CEST 2018
SIGTERM by other means

$ date; sleep 100; date
Mon May 14 22:44:03 CEST 2018
Terminated
Mon May 14 22:44:04 CEST 2018

$ pkill -TERM sleep
Signals - you have already used them

Signals provide a simple way to communicate between processes (IPC - inter-process communication).

Exists on UNIX (or UNIX like) systems and POSIX* compliant systems.

Signals are asynchronous - can interrupt the process anywhere in the normal execution.

*) POSIX - Portable Operating System Interface is a standard (rather a family of standards)
What signals exist?

There’s tons of them. Here are some of the common ones:

**SIGBUS** - sent when a process causes a so called BUS error (e.g. address faulty)

**SIGHUP** - Nowadays used to instruct a program to reload its configuration

**SIGINT or SIGTERM** - Sent to instruct the program that the user wishes to terminate the program

**SIGKILL** - Sent to terminate a process NOW. Can not be ignored.

**SIGSEGV** - Sent when a program references invalid memory (e.g. NULL).

**SIGUSR1 and SIGUSR2** - user defined
What signals exist?

$ kill -l

1) SIGHUP  2) SIGINT  3) SIGQUIT  4) SIGILL  5) SIGTRAP
6) SIGABRT 7) SIGBUS  8) SIGFPE  9) SIGKILL  10) SIGUSR1
11) SIGSEGV 12) SIGUSR2 13) SIGPIPE 14) SIGALRM 15) SIGTERM
16) SIGSTKFLT 17) SIGCHLD 18) SIGCONT 19) SIGSTOP 20) SITSTSP
21) SIGTTIN 22) SIGTTOU 23) SIGURG 24) SIGXCPU 25) SIGXFSZ
26) SIGVTALRM 27) SIGPROF 28) SIGWINCH 29) SIGIO 30) SIGPWR
31) SIGSYS 34) SIGRTMIN 35) SIGRTMIN+136) SIGRTMIN+237) SIGRTMIN+3

.....................
Signals - can interrupt a program?

```c
int main(void)
{
    while (1)
    {
        do_something();
    }
}
```

Normal flow of execution is to continue with the loop, invoking do_something().

If we press Ctrl-c the program will terminate.
Signals - can interrupt a program?

```c
signal_handler()
{
    ....
}

int main()
{
    // register handler ... 
    while (1)
    {
        do_something();
    }
}
```

If we (somehow) register a signal handler for the TERM signal
If we press Ctrl-c the program will be interrupted and execution of the code in the signal handler will done.
How to register such a signal handler?

This is dealt with in the specific signal chapters in our books.