Exercise sheet 8 – Process communication 1

Goals:

- Understand signals
- Network socket programming (client/server)

Exercise 8.1: Signal handling

(a) Update the `OS_exercises` repository with `git pull`.

(b) Change into the `OS_exercises/sheet_08_process_comm1/signal` directory.

(c) Inspect the `signal_example.c` program.

(d) Run the `signal_example` program.

(e) Send a `SIGHUP` to the running `signal_example`. What do you expect? What happens?

(f) Send a `SIGINT` to the running `signal_example`. What do you expect? What happens?

(g) Send a `SIGQUIT` to the running `signal_example`. What do you expect? What happens?

(h) Send a `SIGTERM` to the running `signal_example`.

(i) Send a `SIGKILL` to the running `signal_example`. Is `signal_example` still running? Is it possible to register to this signal inside the `signal_example.c`?.

(j) Run the `signal_example` program with the parameters `--abort`. What happens here?

(k) Run the `signal_example` program with the parameters `--alarm 10`. What happens here?

Exercise 8.2: Chat client/server: network sockets

(a) Change into the `sheet_08_process_comm1/nw_chatserver` directory.

(b) Inspect the `nw_chat_server.c`.

(c) Inspect the `nw_chat_client.c`.

(d) Complete `nw_chat_client.c`.

(e) Compile your program into `nw_chat_client`. Use the prepared `Makefile` with the target `nw_chat_client` for this!

(f) Start the provided `nw_chat_server` or use the `nw_chat_server` provided by the lecturer.

(g) Start your chat client with `nw_chat_client <ip>` and chat. You may use a separate shell for that. You can exit your client by typing `\quit` and press enter.