Birzeit University - Faculty of Engineering and Technology
Electrical & Computer Engineering Department - ENCS4330
Real-Time Applications & Embedded Systems - 1st semester - 2022/2023

Project #1
Signals under Unix/Linux
Due: December 4, 2023

Instructor: Dr. Hanna Bullata

The big race

We would like to create a multi-processing application based on the signals facility under Linux. The idea is to have a parent process fork 10 children processes. The first five forked children are players that belong to *team 1 - the green team* while the last 5 are players that belong to *team 2 - the red team.*

Each team will be eager to start the race from location `A_1` and return to the same location before the other team. The race scenario can be explained as follows:

- The game playground consists of 5 locations named `A_1` to `A_5`. One player of each team will be positioned at these 5 locations. For team 1, the first forked child will be located at location `A_1`, the second child at location `A_2`, the third forked child at location `A_3`, the fourth forked child at location `A_4` and the fifth forked child at location `A_5`. For team 2, the sixth forked child will be located at location `A_1`, the seventh forked child at location `A_2`, the eighth forked child at location `A_3`, the nineth forked child at location `A_4` and the tenth forked child at location `A_5`.
- Each round of the race starts when the parent gives the order to start. The 2 players at location `A_1` will start running and try to reach location `A_2` as quickly as they can while all the other players do not move. The speed for each player is different and should be random. When the 2 players reach location `A_2`, they stop running and the players located at location `A_2` start running towards location `A_3`. Again, the speed for the new runners will be different than the previous ones. The same scenario continues until the last players run from location `A_5` back to `A_1`. The team that fulfills the round trip first is called the round winner.
- The team that wins should communicate that to the parent process which keeps the count of winning rounds for each team.
- Once a round is finished, all players for the 2 teams reset their location to the different steps they belong to and a new round starts when the parent gives the order.
- The application ends when any of the teams has won a total of a user-defined number of rounds (default is 5 rounds if no argument is provided by the user). On termination, the parent kills all its child processes, removes any allocated resource and exits.

What you should do

- Write the code for the parent and the children. The parent's code should be written in the file `parent.c` while the children's code should be written in the file `child.c`.
- Check that your program is bug-free. Use the `gdb` debugger in case you are having problems during writing the code (and most probably you will :-). In such a case, compile your code using the `-g` option of the `gcc`.

- Use graphics elements from opengl library in order to best illustrate the application. Nothing fancy, just simple and elegant elements are enough.
- Test your program.
- Send the zipped folder that contains your source code and your executable before the deadline. If the deadline is reached and you are still having problems with your code, just send it as is!