Birzeit University - Faculty of Engineering and Technology
Electrical & Computer Engineering Department - ENCS4330
Real-Time Applications & Embedded Systems - 2^{nd} semester - 2023/2024

Project #1
Signals & Pipes under Linux
Due: April 5, 2024

Instructor: Dr. Hanna Bullata

Beach ball game simulation

We would like to create a multi-processing application that simulates the behavior of 2 teams of players playing beach ball at Gaza sea shore using signals and pipes facilities. The simulated game can be explained as follows:

1. We'll call the two teams **team_A** and **team_B**.
2. Each team is composed of 5 players in addition to a team lead. The players are numbered 1 to 5 in each team. Of course, initially each player and each team lead has a high level of energy but still all players and team leads do not have the same high level of energy.
3. A parent application will throw initially 2 balls, 1 to the team lead of **team_A** and another to the team lead of **team_B**.
4. Once the team lead has the ball, it'll throw it to player 1 in its team after a random short pause. Player 1 then throws the ball to player 2 after another random short pause. The sequence continues until player 5 throws the ball back to its team lead after a random short pause. Keep in mind the random short pauses are function of the energy level each player and team lead has. Consider as well that a thrown ball might get dropped accidentally so re-collecting it takes time.
5. When the team lead gets the ball back (e.g. the team lead of **team_A**), it'll throw it to the other team lead (e.g. the team lead of **team_B**). In the above example, **team_A** has no ball but **team_B** has 2 balls).
6. When a team has no ball, the parent application will throw a new ball to its team lead. So now we have 3 balls in the game. The sequence of throwing the ball continues as described above.
7. The game continues for a user-defined amount of time (e.g. 5 minutes). When the time is over, we'll count how many balls each team has. The team that has more balls loses the round. If the 2 teams have the same number of balls each, the game scores stay unchanged.
8. To start a new round, go to step **(3)** above.
9. The simulation ends if any of the following is true:
   - The user-defined amount of time allocated to the game is over.
   - One of the teams has lost a user-defined number of rounds.

What you should do

- In order to implement the above-described application, you need to use the signals and pipes facilities. Be wise in the choices you make and be ready to convince me that you made the best choices :-)

- Write the code for the above-described application using a multi-processing approach.
- In order to avoid hard-coding values in your programs, think of creating a text file that contains all the game settings (e.g. values and ranges that should be user-defined) and give the file name as an argument to the main program. That will spare you from having to change your code permanently and re-compile.
- 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.
- 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`.
- 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!