Programming Techniques II
Exercise 10

Task 10.1 Working Together

Pick one of your colleagues whose PennaLV implementation is on a similar level as yours.
You will be working together on each other's projects in this exercise, so remember to grant this person read access rights on GitLab.

Task 10.2 License

Consult the lecture materials and the links in the license wiki-entry on the repository to choose a licence for your project. This choice must allow you and your colleague to view and reuse each other's code, and should allow the TAs to review and modify your project as well.
One common way to apply a license to your project is the following:

- Create a file in the main directory called COPYRIGHT
- Copy the full license or a link to the full license into this file
- State the name of the license in every source file, together with the remark where the full text can be found (i.e mention the file COPYRIGHT)
- If you use doxygen, you can use its copyright tag:

```
/**
 * ...
 * \copyright some_descriptive_licence_name_v2.1 (see COPYRIGHT)
 * ...
 */
```

Task 10.3 Code Review (informal, manual)

A great way to learn to write better code is to look at *lots* of it.
Review your colleague's code:

- Generate its documentation and read it to get a general understanding of the code base
- Go through the source files and notify the author of discrepancies between documentation and implementation
- Report inconsistencies in style
- Note differences to your implementation in the application logic and techinques used for the PennaLV simulation and animal libraries. It is recommended to sit down together for a few minutes and discuss the possible advantages and drawbacks of your respective approaches.

Task 10.4 Animal Exchange

Copy your colleague's animal library into your project, then include and run it through your simulation code in your main function.
Do not forget to use the provided adapter (i.e. `xxx_to_sim<sheep>` instead of `sheep`).
Discuss the observed behaviour.

Task 10.5 Property Challenge (optional)

Our implementation and yours (in case you didn't already fix it) have the following inconsistency in the setup:

```
sheep::set_gene_size(32);
sheep::prop.repr_age = 8;
sheep::prop.threshold = 3;
sheep::prop.mut_rate = 2;
```

We set the gene_size differently than the other variables. It would be nicer to just write:

```
sheep::prop.gene_size = 32; // change
sheep::prop.repr_age = 8;
sheep::prop.threshold = 3;
sheep::prop.mut_rate = 2;
```

The second issue arises from the fact that we use inheritance for the simulation. Now we need to pass in the animals in opposite order (since we cannot change the recursion order for the constructor).

```
sim::simulation<bear, sheep> pennaLV(...);
```

Correct this (somehow but again elegantly), s.t. we can write this again:

```
sim::simulation<sheep, bear> pennaLV(...); // change
```

Make this happen (somehow, but elegantly) in your or the lecture implementation, without changing anything else in the `main.cpp`.
The submission deadline for this challenge is Wednesday 25.11.15 at 05:00 in the morning. Notify us via the mailing list (pt2_hs15_ta@lists.phys.ethz.ch) with a link to your uploaded solution on your PT2 repository on GitLab.