# Making birds fly

**Homework | Agent-based modelling, Konstanz, 2024**

Henri Kauhanen

16 April 2024

> **Note**
>
> In addition to a working Julia installation, you will need the Plots package in order to complete this homework. Run the following commands to install and load it:
>
> ```
> using Pkg
> Pkg.add("Plots")
> using Plots
> ```

In this week's lecture, you learned about custom types, functions and array comprehensions. In this homework, you get to put these concepts to practice.

Specifically, we will implement a version of the artificial birds that we saw in first week's lecture's flocking example. For now, however, we omit the collision avoidance, tracking and navigation rules and only concentrate on figuring out how to make the birds "fly".

Your task is to:

1. Write a custom type called `Bird` which has the following fields, each of them of type `Float64`:
    - `x`, the bird's location coordinate in the horizontal dimension
    - `y`, the bird's location coordinate in the vertical dimension
    - `dir_x`, the bird's direction (where its beak is pointing) in the horizontal dimension
    - `dir_y`, the birds' direction in the vertical dimension
2. Use an array comprehension to make a population of `Bird` objects such that each bird gets a random location and random direction in both dimensions. Store this array in a variable named `population`.

3. Download this file and save it with the filename `plot_birds.jl`. Then run the following in Julia:

```
include("plot_birds.jl")
```

(You may need to specify the whole path of the file in the above command if the file is not located in the same directory as your Julia session.)

> **Tip**
>
> If the above fails, you can also simply copy-paste the contents of `plot_birds.jl` into your own file.

4. Run the following command:

```
plot(population)
```

This will draw the bird's positions and directions in a graphical plot.

5. Write a function called `fly!` which takes one argument – a `Bird` object – and makes this bird fly. In this context, to fly means to move `x` in the direction of `dir_x` by a little amount – let's call that little amount `delta` – and to likewise move `y` in the direction of `dir_y` by the same amount. (You can make `delta` an argument of `fly!` if you want.)

> **Tip**
>
> Use pen and paper to visually figure out how to update the `x` and `y` values of the `Bird` object. Use your imagination. Also note that there is no single "correct" answer as to how this function ought to be implemented. We will look at a few different possibilities in next week's lecture.

6. Test your function by calling it on a few birds in your population of birds. Then re-plot the population. Do the birds' positions change as you'd expect?