project 1: simple strategies for turn-based games

task 1.1: acquaint yourself with python / numpy

Download the file

`tic-tac-toe.py`

from the researchgate site that accompanies this course and run the script. It provides a frame and functions for the game mechanics of *tic tac toe* (if you are not familiar with this game and its rules, just look it up on wikipedia). Try to understand this code, play with it, and get a feeling for it; this will come in handy in task 1.3.

task 1.2: simple strategies for *tic tac toe*

Note that the above script has both players move entirely at random. Do something about this and implement functions that realize more intelligent moves. Proceed as follows:

1. **implement a probabilistic strategy**

   have both players play many games (at least, say, 10000) in order to create a statistic of auspicious positions on the *tic tac toe* board;

   after the whole tournament, plot a histogram of wins and draws;

   after each game in the tournament that did not end in a draw, check which player has won and determine the fields this player occupied in order to count for each field how often it contributed to a win;

   properly normalize your count data (such that they sum to one) and store them on disk; now implement a function that realizes a game move using the probabilities you just determined;

   use this function for the moves of player **X** and have player **O** move at random; start another tournament and plot the new histogram of wins and draws

2. **implement a heuristic strategy**

   think of ways of evaluating the quality of a potential move;

   implement a strategy where the moving player evaluates all free positions on the board and selects the most auspicious one;

   use this function for the moves of player **X** and have player **O** move at random; start another tournament and plot the new histogram of wins and draws

task 1.3: *connect four*

Get inspiration from looking at the code for *tic tac toe* and implement the game mechanics for *connect four* on a 6×7 board (if you are not familiar with this game and its mechanics, just look it up on wikipedia).
Realize proper functionality for random moves and game termination tests; have a tournament between two players moving at random (but of course according to the game rules) and try to collect statistics as to likely good moves; do you run into a problem? if so, think about what the underlying issue is; how could it be solved?
Realize a "beautiful" graphical user interface for *connect four* that allows for visualizing the progression of a game as well as for user inputs; **note:** should you have no idea whatsoever as to how to do this, simply google for "connect four python" and have a look at the many solutions you will find; you may also search for corresponding tutorials an YouTube; there is a wealth of information out there; please just use it!

task 1.4: *breakout*

Search the Web for python implementationss of the video game *breakout* (if you are not familiar with this game and its mechanics, just look it up on wikipedia).
An example of a particularly simple implementation is *bricka*.
Familiarize yourself with how *breakout* is typically implemented and play with the code you will find; for instance, see if you can increase the speed of the ball or the panel; **note:** your preparatory work on this task will come in handy for later projects.