PROJECT 3: CHAOS GAMES

Creating fractals through a set of rules with random weights, and plotting these.

— Triangle, chaos game on a triangle

— Chaos Game, chaos game on an n-gon

— Barnsley Fern, affine transformation with a given set of functions and probabilities

— Variations, non-linear transformations on the space of the figures

Topics from IN1910:

— Classes and methods, for building the classes

— Using random numbers, when determining the next point in an iteration

— Codestyle and docstrings, to make code readable

— Optimisation, used numba to compile parts

PROJECT 3: SOLVING THE TASKS

— Pair programming, mostly taking turns per sub-task given and working on them in the order given
— We decided to work with numpy arrays instead of lists

Chaos Game:

— We chose to make a second iteration function outside of the class using numba to increase speed when doing many calculations
— For the save figure task we decided to create a make figure function with lists

Fern:

— We made a second faster class for fern in numba, although it doesn't make much difference unless we later decide to animate it for different inputs

Variations:

— For the optional animation we decided to do two types of transformations, Ax → Bx and Ax → Bax.

PROJECT 3: CHALLENGES

— Looked things up online if we were stuck on how to proceed

— Running different parts of the code with print statements to find errors

— Optimising Fern, it wasn't as easy to jit as the iteration in Chaos Game

Variations:

— Forgetting to scale Fern to fit the transformation

— Spiral variation looks to be incorrect, but we couldn't figure out why

— Some of the variations were too big, so we made an optional limit for x and y values

Animation:

— Figuring out how to store data sets for more than 2 methods in one animation in a way that let us plot it

— Adding colour to the animation, since it wouldn't let us use the color mapping we already had

PROJECT 3: TESTING

Testing on the go:

— Running the program for each new method to check for syntax errors

— Printing in different parts of a method when it didn't work, or to check that conditions were triggered

— Plotting for different values to see if the results are sensible, sometimes looking up images online to compare

Chaos Game unit tests:

— We found the formula for the lines going through the corners, then made two tests, one for starting points and one for iterations, to check that points where within the shape

— Checked if n-gon is equilateral and centered around origin

— Checked if errors are raised when they should be