

# Problem-set: Cython at Trento Autumn School 2010

- From Python to Cython
- Wrapping C
- L-Systems
  - Hints

These files may be downloaded from <https://portal.g-node.org/python-autumnschool/materials:cython>

Please do explore beyond the problems given, and feel free to ask questions at any time.

## From Python to Cython

Consider the provided code, `fractal.py`, that computes pretty pictures. Optimise it, using Cython and the techniques discussed in the lecture. Your Cython code should be able to outperform the NumPy version (also provided)!

## Wrapping C

Implement a function in Cython that computes `sin(x)` on each element of a large floating point array. Compare the speed of this function to NumPy's built-in `sin` (which, by the way, can also operate on an entire array at a time). If it is faster, can you explain why?

## L-Systems

Implement an **L-system** in Cython. Try, for example, to build a Sierpinski Triangle or the Dragon Curve. If you want, you can first implement it in pure Python, and then add the type information later.

### Hints

- Use your Cython program to generate coordinates, then simply "connect the dots" using `matplotlib` (`import matplotlib.pyplot as plt; plt.plot(...)`).
- Cython can accelerate operations on non-numerical types too. For example, if you store your coordinates in a list, you can use:

```
cdef list L = []
```

For dictionaries, the type is `dict`.