Python Programming Language

Throughout the whole specialization we will show you, from time to time, Python code snippets. We will also suggest you to solve a number of programming challenges. All of them are optional and are not formally required to pass the course.

Why on earth do we start the course with discussing a programming language? After all, this is a math (rather than programming) course!

That's true. But we believe that many pieces of code shown in this course will help you in many ways:
• They will show you a rich variety of applications of discrete math ideas in various branches of computer science.
• Code snippets can serve as interactive examples: you may want to tweak the given piece of code, run it, and see what happens.
• By trying to implement a particular idea, you are forced to understand every single detail of it.
• It is often easier to reason in terms of specific objects in programming rather than abstract mathematical concepts.
We have set up everything in a way that will allow you to run the code snippets used in this course even if you have never tried to write a program before. You don't even need to install or set up anything: everything can be run in the cloud, through your Internet browser. At the same time, we also provide instructions for those who would like to learn the basics of Python
while learning discrete math.

OK, let's do some programming while learning discrete math. But why Python instead of any other popular programming language?

Let us convince you that Python is an excellent choice for our purposes.

High-level language. It is particularly easy to start using Python (even if you haven't programmed before). The syntax is reader friendly (and close to a natural language). The code is compact: most of the pieces of code in this course are less than ten lines long!
Interactive mode. It can be used in an interactive mode (also known as REPL , for read-eval-print loop). This allows you to talk to your computer using Python as a language: the computer then reads your input, evaluates it, and prints the result. This way, you work stuff out and get instant feedback from the machine.
"Batteries included". The Python standard library offers a wide range of facilities, and many external libraries are available as well. In particular, this will allow us to generate a random sequence, plot a function, and draw a graph in just one line of code!
This (partly) explains why Python is often used for software prototyping, and in such areas as machine learning, data science, and web development.
Of course, advantages always come at the cost of some disadvantages. The high-levelness of Python makes it less flexible in performance tuning. This is OK for us, as we will only be using simple snippets of code where optimizing is not an issue.

OK, let's try! Where do I start?
Locally. To install Python on your machine, go to the Get Started section of python.org and follow the instructions. If you are new to Python, we encourage you to install PyCharm to start working with Python: this (free of charge) professional IDE will make the process of writing and running your code smoother and more efficient.
In the cloud. Alternatively, you may run all our code snippets from your Internet browser, without installing or configuring anything on your machine. To do this, visit the page github.com/alexanderskulikov/discrete-math-python-scripts and click the badge "Open in Colab". This will show you a list of notebooks that can be run in an interactive mode right in your browser (together with links to a tutorial on notebooks).