EXPLORE AI ACADEMY

Python

Module overview

Module timeline

Phase 1: Python basics

During weeks 1 to 4 we form a strong foundation by covering basic Python functionality and concepts.

This phase is aimed at getting to grips with the language itself, learning basic programming skills, getting comfortable with the Python environment, and generally finding our feet. This phase includes topics such as data types, operations, creating functions, learning about parameters, and object-oriented programming, all skills required for Data Science.

Phase 2: Applying Python to Data Science

Having established a good base in Python, weeks 5 to 8 will focus on using more advanced Python for Data Science problems.

While the content matter will continue to increase in complexity during this phase, we aim to keep the applications practical. Knowledge pieces and exercises will assist in developing programming skills and understanding, not only of the Python language, but of common Data Science problems.

Lesson structure: Building the knowledge base

Lesson overview	Each lesson starts with a **lesson overview**. It provides a concise view of the **topics** that will be covered, the **activity types**, as well as the **estimated time** to be spent on each lesson.
Video	The **videos** aim to introduce and **explain important concepts** in a fun and engaging way.
Walk-through	Walk-throughs are essentially **how-to videos**, showing how we could approach a **specific problem** or how to use certain commands. These videos are also used to illustrate interactions with the learning environment.
Examples	Examples are provided in the form of **Jupyter Notebooks**. These **notebooks** should be **downloaded** and then **run locally** to get the most out of them. Work through them **step by step**, **experimenting** with the code provided, thereby getting a better **understanding** of the concepts.
Markdown / Slides	These documents allow us to **dig deeper into theory** whenever necessary. They are also used to provide **links** to **external resources**, should the need arise.

Lesson structure: Knowledge testing

> In order to test newly acquired knowledge, we have sets of **Knowledge Questions** following certain activities, such as Examples, and Videos. In some lessons, we might also test **overall understanding** of concepts and their application, by using **graded Multiple Choice Questions.**

Knowledge Questions

Knowledge Questions are provided to keep us on our toes! These questions are generally **based on theory** or **new concepts** we've just discussed and should be seen as an opportunity to **self-assess** understanding.

Multiple Choice Questions

These questions are **normally graded** (unless specified otherwise) and will **test the understanding** of a whole lesson's contents. These questions are aimed to be **more practical**, so don't be surprised if writing some code is required in order to complete the assessment!

Lesson structure: Exercises and code challenges

> We can **develop our Python skills** by completing the **non-graded Exercise** notebooks in each lesson. **Code challenges** are **graded assignments** that need to be **submitted** to the learning platform once completed.

Exercise

Exercises are **non-graded** Jupyter Notebooks that contain **problem sets**. Each exercise is split into **two** sections, the first containing the **problem sets**, and the second section containing the **solutions**.

There are little nuggets of **information** and **handy tips** strewn between the problems and solutions, so don't be tempted to skip over them! It is **important** to know that there might be **multiple solutions** to a problem, and that the solutions provided are not necessarily the only answers.

Code challenge

Code challenges are **graded assignments** that aim to test skills and knowledge. They provide a real-world experience of having to **solve problems** with Python. Each code challenge comes with a clear **set of instructions** – read it carefully to see how the assignment should be submitted to retain the required format. Once submitted, the notebook is graded by our **autograder**, using unit tests on the code.

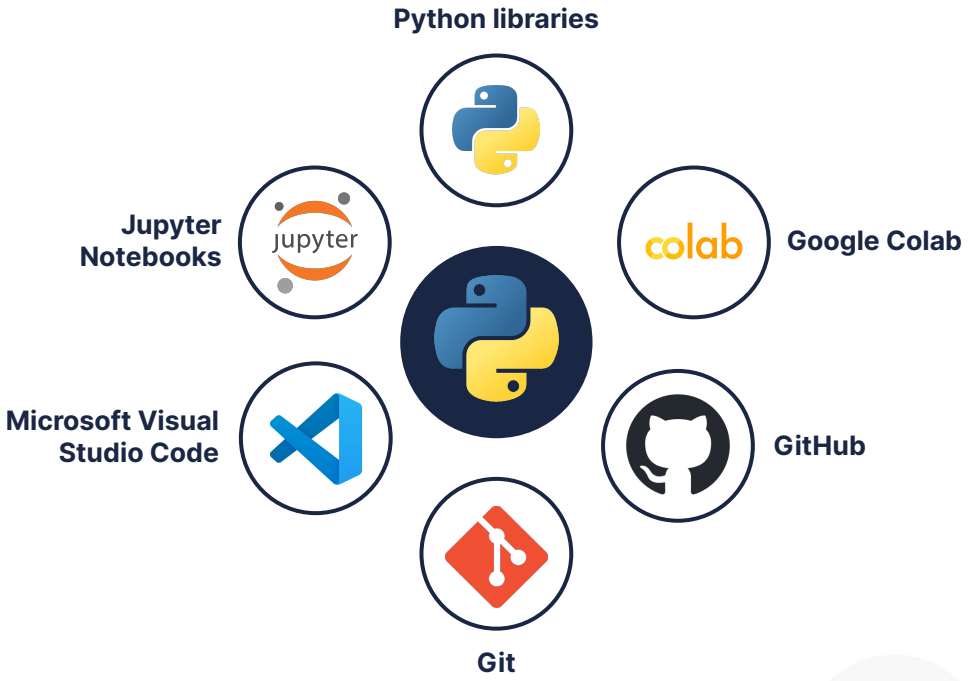
The first code challenge, *Farm management [Code challenge]*, should be seen as a practice round for code challenges, as it will not be graded by the autograder and does not need to be submitted to the learning platform.

Tools

Over the duration of this module, we'll introduce a variety of **tools** to write and help manage our Python code.

We'll look at how to use them and see which tools are useful in which situations.

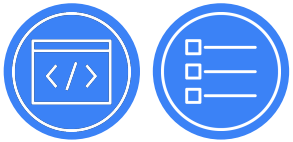
By the end of this module, we'll know which works best for us!



What to expect

> This module was built to **create a growing understanding of Python**, focusing on **practical problem solving**. It is however only the start of the Python journey!

Every week will contain a mixture of **knowledge pieces**, **exercises**, and **knowledge questions.**



There will only be **one graded assignment** per week – either in the form of a **Code challenge** or a set of **Multiple Choice Questions**.

It is impossible to cover all aspects of Python in a single module, so don't hesitate to use the **Python documentation** for further reading when necessary. **Good luck!**