Pre-Work

1. **What experiences have you had with coding and/or programming so far? What other experiences (programming-related or not) have you had that may help you as you progress through this course?**

Before the course, I didn't know anything specific about programming. In the Intro and Immersion course, I learned a lot about HTML, CSS, JavaScript, Node.JS, React, Angular, React Native, and a lot of packages that I used within them.

Programming is challenging at times, but I like challenges. For example, I like to play games on the hardest difficulty possible and earn all achievements in them. I used this kind of approach, to go through the course this far, and I'm enjoying it.

2. **What do you know about Python already? What do you want to know?**

I knew Python is a programming language, and that it is the best language to start to learn coding. Then, in the Immersion course, I learned, that Python is mostly used at the backend of an application because it needs to be transpiled for the browser to be able to read and utilize it.

I'd like to know how exactly Python works and what it is capable of.

3. **What challenges do you think may come up while you take this course? What will help you face them? Think of specific spaces, people, and times of day of the week that might be favorable to your facing challenges and growing. Plan for how to solve challenges that arise.**

The first challenge will be to get used to the new language we learn here, and how differently it will be used. My curiosity will possibly help me through this course, and the thought, that I would like to work as a developer.

At challenges in the tasks firstly I will check the documentation to find a solution to my problem. If it doesn't help, I will search for a similar problem someone had at Stack Overflow. If none of them works, I will ask my Mentor for some advice.

Exercise 1.1: Getting Started with Python

1. **In your own words, what is the difference between frontend and backend web development? If you were hired to work on backend programming for a web application, what kinds of operations would you be working on?**

Frontend development deals with the client side of an application. They work on how the application looks and feels to a user, how to send requests to the backend, and how to read the data returned from a request.

Backend development deals with the server side of an application. Their job is to elaborate on what data the application will use, how and where will the data be stored, and how will the client side communicate with the database.

I would be working on a REST API, which will communicate between the client side and the database, create models of how the data should be stored in the database (if the project works with a NoSQL database), validate requests, and let only valid data get stored, and last but not least, creating the logic, which allows only specific origins to communicate with the API.

2. **Imagine you're working as a full-stack developer in the near future. Your team is asking for your advice on whether to use JavaScript or Python for a project, and you think Python would be the better choice. How would you explain the similarities and differences between the two languages to your team? Drawing from what you learned in this Exercise, what reasons would you give to convince your team that Python is the better option?**

It is one of the easiest languages to learn and understand, and much easier to debug or document it due to its readability. It uses simple built-in package management, which is useful in the way we don't need to look for third-party libraries in our project and the most common web operations are available in it as well. It has one of the largest communities, and it's the best language for fast prototyping.

3. **Now that you've had an introduction to Python, write down 3 goals you have for yourself and your learning during this Achievement. You can reflect on the following questions if it helps you. What do you want to learn about Python? What do you want to get out of this Achievement? Where do you see yourself working on after you complete this Achievement?**

   1: Learn how to use Python effectively.
   2: Progress through the Exercise
   3: Continue with Achievement 2.

Exercise 1.2: Data Types in Python

1. **Imagine you're having a conversation with a future colleague about whether to use iPython Shell instead of Python's default shell. What**

reasons would you give to explain the benefits of using the iPython Shell over the default one?

- iPython uses color schemas to distinguish line of codes, suggestions, etc.
    - It gives more readability to the shell
- iPython has an autocompletion tool
    - It writes all the possible functions usable with the sequence, considering its data type. It can be chosen by pressing 'Tab' navigating to the right function with the arrow keys, then pressing 'Enter'

2. **Python has a host of different data types that allow you to store and organize information. List 4 examples of data types that Python recognizes, briefly define them, and indicate whether they are scalar or non-scalar.**

Data type	Definition	Scalar or Non-Scalar?
string	can store alphanumeric data with special characters surrounded by single or double quotes	Non-Scalar
boolean	can store either true or false, which can be used for example as statements to execute specific code lines	Scalar
integer	can store whole numbers in positive and negative directions as well	Scalar
list	Lists are able to store every kind of data type, even other lists, surrounded by square brackets '[ ]', and they are mutable.	Non-Scalar

3. **A frequent question at job interviews for Python developers is: what is the difference between lists and tuples in Python? Write down how you would response.**
   Tuples are ordered, immutable data storages surrounded by parentheses.
   Lists are ordered, mutable data storages surrounded by square brackets.

Tuples are processed and read faster, but they have the disadvantage of immutability. If we would want to modify its content, we have to add all the information again. They are most useful at working with a huge database.

Lists, on the other hand, have larger memory usage and are slower to read, but their content can be reassigned, modified, and new data can be added to it easily.

4. **In the task for this Exercise, you decided what you thought was the most suitable data structure for storing all the information for a recipe. Now, imagine you're creating a language-learning app that helps users memorize vocabulary through flashcards. Users can input vocabulary words, definitions, and their category (noun, verb, etc.) into the flashcards. They can then quiz themselves by flipping through the flashcards. Think about the necessary data types and what would be the most suitable data structure for this language-learning app. Between tuples, lists, and dictionaries, which would you choose? Think about their respective advantages and limitations, and where flexibility might be useful if you were to continue developing the language-learning app beyond vocabulary memorization.**

   Since flashcards don't usually need changes, tuples would be the best choice in my opinion. Due to its fast processing behavior, the application would run faster, and when a user would like to change something in a word's data, the frontend would fill out every input field, so the user can update data with minimum effort. When the development of the application continues, the small memory usage will come handy, for example, if a lot more languages are added to the application.