Exercise 1.2: Data Types in Python

Learning Goals

- Explain variables and data types in Python
- Summarize the use of objects in Python
- Create a data structure for your Recipe app

Reflection Questions

1. Imagine you're having a conversation with a future colleague about whether to use the 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?

    a. The iPython shell is more user-friendly and interactive than the default Python shell. It supports syntax highlighting, tab autocomplete so its easier to find available operations and complete commands, automatically indents your code, therefore the code written in iPython is generally more readable and its better managed.

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?
Integer	Whole numbers - negative, zero, and positive. The limit is infinity to both sides (or as much as the computer memory allows).	Scalar
Float	Decimal numbers - negative, zero, and positive. The limit is $10^{308} (or as much as the computer memory allows).	Scalar
Tuple	A linear array-like structure that's immutable. It can store multiple values of any type, e.g., water_level = (100, 98, 84, 89, 92, 101). It supports indexing, slicing. Adding new values either via creating a new version of the tuple or concatenaning 2 tuples.	Non-Scalar
String	Contains alphanumeric characters and symbols. Immutable in Python and needs to be surrounded by single or double quotes.	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 respond.

    a. Tuples are in essence immutable objects while lists are mutable. There is, however, one disadvantage that lists have over tuples, which is that they are slower to read and access, which makes a big difference especially with large data sets.

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.

    a. I'd suggest using dictionaries for storing different flashcards - they would have keys pertaining to different parts of a flashcard (word, definition, category). New keys can also be easily added to track a user's progress, preferences for each word, etc. Then, it could be useful to create lists for each word category that would store words of that category and would speed up sorting and selection of these words, or generally lists could help build other relationships in the app should it expand in the future.