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?

   IPython Shell is more powerful than the default Python shell because it offers features like tab-completion, inline help, command history, and better output formatting, making coding, testing, and debugging faster and easier.

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?**
int	Represents whole numbers (eg: 5,-3,0)	scalar
float	Represents decimal numbers (e.g., 3.14, -0.5)	scalar
str	A sequence of characters, used for text (e.g., "hello")	Non-scalar
list	An ordered collection of elements, which can be of different types	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.

Lists are **mutable**, meaning their contents can be changed (items can be added, removed, or modified). They are defined using square brackets `[]`.
Tuples are **immutable**, meaning once they are created, their contents cannot be changed. They are defined using parentheses `()`.

Because of this, tuples are generally faster and can be used as keys in dictionaries (if they contain only immutable elements), while lists cannot.

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.

For a language-learning app using flashcards, the **most suitable data structure is a dictionary**.

Justification:

- A **dictionary** allows you to store each flashcard as an object with clear **key-value pairs**:
  - `'word'`: the vocabulary word (string)
  - `'definition'`: its meaning (string)
  - `'category'`: part of speech like noun, verb, etc. (string)
- Dictionaries are **flexible**, **readable**, and easy to **update or extend** (e.g., adding examples, pronunciation).
- Compared to tuples (which are immutable) or lists (which rely on index positions and can be confusing), dictionaries provide **structured data** and support **scalable development**.

Example structure:

flashcard = {

```
  "word": "run",
  "definition": "to move swiftly on foot",
  "category": "verb"
}
```