Python technical exercise #1 - Currency conversion

Introduction

Purpose of this exercise is to test candidate's problem-solving and software development skills on a real-world example. Exercise is split into 3 parts, which are meant to build on top of each other, producing a single feature.

Please note: goal of the exercise is to provide a *robust*, *tested* and *easily extensible* implementation of the described feature. In case the allotted time is not enough, please priorize quality over quantity (e.g. two high-quality tasks are better than three low-quality ones).

Guidelines

1. *All function signatures mentioned in the exercise have to be adhered to. Other functions can be introduced, depending on the need.*
2. *Each function mentioned below should be properly documented (docstring) and have a test case that covers it.*
3. *In case of invalid input, lack of data, or any other errors, inform the user by raising an exception.*

Implementation language: Python 3.7+
Duration: 4 hours
Output: ZIP archive with necessary `.py` files, together with a `README` file with instructions on how to execute the code.

Background

Currency conversion is one of the basic operations in the finance field - first step in any data transformation is normalizing the currency of the data to be processed.

Part 1 - Retrieving exchange rates

Background

Exchange rate data can be retrieved from the REST API of the European Central Bank, using the URL below:

`https://sdw-wsrest.ecb.europa.eu/service/data/EXR/M.GBP.EUR.SP00.A?detail=dataonly`

In the URL above, `GBP` is the source currency and `EUR` is the target one. These values can be replaced with any combination of currency codes (e.g. `PLN.EUR`).

Response, in XML format, contains multiple children of the following format:

```
<generic:Obs>
    <generic:ObsDimension value="1999-01"/>
    <generic:ObsValue value="0.7029125"/>
</generic:Obs>
```

Each of these represents the value of the exchange rate at a certain point in time.

Task

Implement the function with the following signature:

```
def get_exchange_rate(source: str, target: str = "EUR") -> pd.DataFrame
```

Function should fetch the exchange rate data from the appropriate URL and convert it to a pandas DataFrame, with columns `TIME_PERIOD` and `OBS_VALUE`, corresponding to values of `generic:ObsDimension` and `generic:ObsValue` tags from the XML. `OBS_VALUE` should be converted to `float`.

Example

Running: `get_exchange_rate("GBP")` should produce the following:

TIME_PERIOD	OBS_VALUE
1999-01	0.702913
1999-02	0.688505

...

Part 2 - Retrieving other data

Background

Other data of interest can be be retrieved from the REST API of the European Central Bank, using the URL below:

`https://sdw-wsrest.ecb.europa.eu/service/data/BP6/M.N.I8.W1.S1.S1.T.N.FA.F.F7.T.EUR._T.T.N?detail=dataonly`

In the URL above, `M.N.I8.W1.S1.S1.T.N.FA.F.F7.T.EUR._T.T.N` is the data identifier and can be replaced with any user-specified identifier.

Task

Implement the function with the following signature:

```
def get_raw_data(identifier: str) -> pd.DataFrame
```

Function should fetch the data from the appropriate URL and convert it to a pandas DataFrame. Format of the response XML, as well as the format of the resulting pandas DataFrame is same as in the part #1.

Example

Running:

`get_raw_data("M.N.I8.W1.S1.S1.T.N.FA.F.F7.T.EUR._T.T.N")`

should produce the following:

TIME_PERIOD	OBS_VALUE
1999-01	1427.666667
1999-02	379.666667

...

Part 3 - Data transformation

Task

Implement the function with the following signature:

```
def get_data(
    identifier: str,
    target_currency: Optional[str] = None
) -> pd.DataFrame
```

Function should retrieve the data using the function described in the part #2.

If the `target_currency` parameter is `None`, leave the resulting DataFrame as-is.
If the `target_currency` parameter is not `None`, convert the data from the source currency to the target one, defined by the `target_currency` parameter. Exchange rates for the currency conversion should be retrieved using the function described in part #1.

By convention, source currency is given by the 12th component of the identifier (0-based).

Example

Running:

`get_data("M.N.I8.W1.S1.S1.T.N.FA.F.F7.T.EUR._T.T.N", "GBP")`

should produce the following:

TIME_PERIOD	OBS_VALUE
1999-01	1003.52546
1999-02	261.402399

...