Exercises - Part 4

October 19, 2020

0.0.1 Built-in Functions

```
# Example
fruits = ["apple", "kiwi","orange", "banana"]
print(len(fruits))
```

Exercise: 1. Assing the variable `numbers = [26473.014, -13474, -3669.4, 178934, 94371]` 2. Use the `min()` function to print the lowest number 3. Use the `max()` function to print the highest number 4. Use the `sum()` function the print the sum of all the numbers

Extra challenge: create a new list called `numbers_rounded` with the all the values from `numbers` rounded. You can use the `round()` function. The ouput of printing the list should be `[26473, -13474, -3669, 178934, 94371]`

```
[ ]: # Exercise 1
```

```
[ ]: # Exercise 2
```

```
[ ]: # Exercise 3
```

```
[ ]: # Exercise 4
```

```
[ ]: # Extra challenge
```

0.0.2 User Defined Functions 1

```
# Example
def convert_eur_to_usd(amount):
    usd = amount * 1.18
    return usd
```

```
# Example
round(3.14159265359, 4) # prints 3.1415
```

Exercise: 1. Create a function `convert_usd_to_eur` that converts an amount in USD to EUR 2. Use your function to calculate how much `$150` is in euros 3. Recreate the function to round the output of the function with two decimals 4. How much is `$150` in Euro's, rounded to the cent?

Extra challenge: instead of returning a numerical value, let the function return a string with the euro sign. So `print(convert_usd_to_eur(150))` should return `€127.12`

```
[ ]: # Exercise 1
```

```
[ ]: # Exercise 2
```

```
[ ]: # Exercise 3
```

```
[ ]: # Exercise 4
```

```
[ ]: # Extra challenge
```

0.0.3 User Defined Functions 2

```
# Example
def add(x, y=None):
    return x + y

print(add(4, 5))  # prints 9
```

Exercise: 1. Create a function `multiply` which takes `x` and `y` as inputs. The function should return `x` and `y` multiplied. 2. Call the function with `multiply(4, 5)`. It should return `20`. 3. Sometimes a user might not know the value of `y`. Give `y` a default value of `1`. 4. Call the function with `multiply(4)`. It should return `4` (instead of an error).

Extra challenge: entering more are arguments, like `multiply(4, 5, 6, 7)` will result in an error. Search the internet for a solution. Can you change the function so that it can deal with an unknown amount of arguments?

```
[ ]: # Exercise 1
```

```
[ ]: # Exercise 2
```

```
[ ]: # Exercise 3
```

```
[ ]: # Exercise 4
```

```
[ ]: # Extra challenge
```

0.0.4 Modules

```
# Example
import statistics
numbers = [1, 2, 3, 4, 5, 6]
statistics.mean(numbers)    # prints 3.5
```

Exercise: 1. Import the built-in math module and take a look at the documentation 2. Print the square root of `21000` 3. Use the **`math`** module to print the value of `pi`. Hint: see documentation, this is not a function, but a constant. So you dont need to call it with parentheses `()`.

Extra challenge: use the built-in collections module (hint: `from collctions import counter`) to find out which **two** letters occurs the most often in the word `antidisestablishmentarianism`

[ ]:
```
# Exercise 1
```

[ ]:
```
# Exercise 2
```

[ ]:
```
# Exercise 3
```

[ ]:
```
# Extra challenge
```

0.0.5 API Calls

```
# Example
import requests
url = "https://api.ratesapi.io/api/latest"
r = requests.get(url).json()
r
```

Exercise: 1. Pick your favourite API from the list below 2. Take a look at the documentation. Look for the URL where you can get the data. 3. Make an API call (similar to the example above) 4. Inpsect the results in `r` 5. *Optional*: to print only information you are interested in. You can use the `.keys()` and `.values()` methods to inspect a dictionary. If you retrieved a list of dictionaries, you could loop through the results. 6. Do the same exercise for another API in the list

List of public APIs:

Name	URL
Exchange Rates API	https://exchangeratesapi.io
Open Brewery DB	https://www.openbrewerydb.org
Open Library API	https://openlibrary.org/developers/api
Pokemon API	https://pokeapi.co
SpaceX API	https://github.com/r-spacex/SpaceX-API
Star Wars API	https://swapi.dev
Weather API	https://www.metaweather.com/api/

Extra challenge: Many online tools offer REST API access to their product. For example, HubSpot, Dropbox, Trello and many others. Search the internet for APIs of the tools you are using. Most often, you will need to authenticate with an API key. Explore the APIs documentation for the authentication process. Can you retrieve your data from these tools?

[ ]:
```
# Exercise
```

[ ]:

[ ]:

[ ]:

[ ]: