# Python Exercise

*All exercises should be completed in Spyder unless anything else is specified – enjoy!*

## Exercise 0: Setting up the conda environment

0a)

To start the exercises, activate the conda environment *python_day2* from the command line.

*It is an environment containing Python and the Python package Biopython*

## Exercise 1: Print "Hello World"

1a)

Using Spyder, create a file called *myfirst.py*

1b)

Print the sentence:

```
"A python ate my mother-in-law, what should I do?"
```

1c)

Define three variables (VarA, VarB, VarC), and print the statement:

```
"Hello VarA, I am sorry I'm late. My VarB ate my VarC"
```

1d)

From the variable below, print only the positive part of the sentence using slicing:

```
VarD = "He's lazy, he's ugly, but he smells good"
```

## Exercise 2: Strings, Integers, and floats

2a)

Which type/class are the following: `"One"`, `1`, `"1"`, `1.0`

2b)

Use one of the integrated Python functions to convert the integer **3** into a float

2c)

Which type/class is True and False?

2d)

Test out the following statements in Spyder and note if they are defined as True or False:

1 + 1 == 2

2 + 4 == 5

1 == 1.0

2 != 2

## Exercise 3: Lists

3a)

Create a list of five things that should be in a burger, along with five of your favorite numbers (these can be floats and/or integers). Call that list *burger_list* and print it.

*Hint: "1" is not an integer*

3b)

Using indexing, print out the third, seventh, and tenth item *of burger_list*

## Exercise 4: If statements in Python

4a)

Make an *if statement*, that tests if any number of your choice is bigger than 68.

*Hint: Remember indentation!*

4b)

Make an *if statements*, that tests if the third, seventh and tenth element in *burger_list* is a string.

*Hint: Use indexing*

## Exercise 5: For loops in Python

5a)

Make a list of cute animals. Print every animal in that list using a *for loop*.

*Hint: Remember indentation*

5b)

Combining a *for loop* with an *if statement*, print every **string** (X) in *burger_list.*

Let the print statement be "X really goes in a burger!".

*Hint:* `type(X) == str:`

## Exercise 6: Open and write out a file

Download the file called **get_fasta_header.py** from EVA and place it in /home/student/BTG_intro/Scripts**.** Open the script in Spyder. Read through the script and note what you expect to happen once running it.

Add comments inside the script at all locations where you see a #.

Think about the following:

- What does the "a" mean on line 10?
- Why is `header` assigned as `line[1:]` and not just `line`?
- Why is fasta_headers.txt not closed?

Run the script from the command line. Is the outcome as you expected?

### Exercise 7: Biopython

7a)

Download the file called **get_fasta_length.py** from EVA and place it in /home/student/BTG_intro/Scripts. Open the script in Spyder. Read through the script and note what you expect to happen when executing it. Run the script from the command line and observe the output.

7b)

In Spyder, add # in the beginning of line 13 and 14 and note what you expect to happen when running the script now. Run the script from the command line and observe the output.

7c)

Using nano, replace line 5 with the following:

```
fasta_file = sys.argv[1] #Takes the first argument as input
```

Run the script from the command line and use **MRSA2.fasta** as argument for the script.

## Extra exercises

### While Loop

a)

Set a variable **sheep** to 0. Using a while loop, print the amount of **sheep** and increase the number of **sheep** by 1 until you reach 23.

b)

Create the duck list below:

```
duck_list=["duck", "duck", "duck", "duck", "duck", "duck", "duck", "goose",
"duck", "duck", "duck"]
```

Using a *for loop*, count the ducks and stop when you reach the goose. Print “Goose!” and the number of ducks you counted before the goose.

c) Repeat exercise b, but use a *while loop* instead of a for loop.
*Hint: Use* `len()`

## Making a function

a)

Improve the *if statement* from exercise 4a and make it into a function so that running higher_than(X) will return one of the following:

```
"X is not higher than 68"
"X is higher than 68"
"both numbers are 68"
```

b)

Improve the *function* from the previous exercise and make it take two inputs (integer or float) so that running *higher_than(X,Y)* will return one of the following:

```
"X is not higher than Y"
"X is higher than Y"
"Both numbers are X"
```

*(if X and Y are the same)*

## Sets

Using the set() constructor function, convert *duck_list* to set and observe the difference.

## Dictionaries

a)

Create a dictionary called *favorite_foods* with your top three favorite foods as keys and their corresponding deliciousness levels (on a scale of 1-10) as values. Print the dictionary to see your favorite foods and how delicious they are.

Choose one food and print its deliciousness level along with an explanatory message by retrieving the information from the dictionary.

*Example: `The deliciousness level of broccoli is 10`*

Modify the deliciousness level to make it even more delicious and make a new explanatory message.

*Example: `Actually, the deliciousness level of broccoli is more like 11`*

Add your least favorite food to the dictionary along with its corresponding deliciousness levels and update the explanatory message.

*Example: `I don't care for pizza; I only find its deliciousness level to be 3`*

b)

Make a dictionary called *my_dict* with a key called "number" and a key called "food" that each have an empty list as value.

Make a *for loop* that iterates through the *burger_list* you created in exercise 3a and append all strings to key: “food” and all int/floats to key: “number”.