Python

#3

Agenda

- Functions
  - Function definition
  - Function arguments
    - Positional arguments
    - Keyword arguments
- Regular expressions

Who am I?

Michał Gałka

Contact:

galka.michal@gmail.com (e-mail or hangouts)

Use **[BioIT]** in the e-mail topic.

Who am I?

Michał Sarna

Contact:

michalsarna@gmail.com (e-mail or hangouts)

Use **[BioIT]** in the e-mail topic.

Who am I?

Marcin Górski

Contact:

mkgorski@gmail.com (e-mail or hangouts)

Use **[BioIT]** in the e-mail topic.

Who am I?

Andrzej Stasiak

Contact:

aandrzej.stasiak@gmail.com (e-mail or hangouts)

Use **[BioIT]** in the e-mail topic.

Functions

Functions

- Functions are the way to group code of a similar functionality.
  - You can later on reuse the code.
  - The code can be made more generic by adding parameters.
- Each function has its unique name.
- There are a lot of built-in functions in Python like: print, len

Calling a function

- To call a function you need to write its name and parameters in a parenthesis.
- If function doesn’t require any parameters you must place the empty parameters after the name to call it.

```
print('My text')
myfunc()
```

Function definition

- Do define your own function you need to use the keyword **def**.
- Each function has three parts:
  - ers (can be empty)
  - D

Function name

Parameters

```
def print_hello(name, surname):
    print('Hello {} {} !'.format(name, surname))
```

Function body

Function parameters

- In every function definition you need to place a list of parameters in parenthesis.
- Parameters are separated by a comma.
- If there are no parameters that function requires you must place the empty parenthesis.
- Parameters are available in the function body as variables.

```python
def print_hello(name, surname):
    print('Hello {} {} !'.format(name, surname))
```

Positional and keyword parameters

- Positional arguments are defined only by their name.
  - They are required during the function call.
- Keyword arguments are defined by their name and value.
  - They are optional during the function call.
  - If they are not present function will use their value from the definition.
    - The defined value is called a **default value**.

```
def print_n(text, n=5):
    for i in range(n):
        print(text)
```

Positional and keyword parameters

```python
def print_n(text, n=5):
    for i in range(n):
        print(text)
```

```python
print_n("my text")
print_n("my text", 10)
print_n("my text", n=10)
```

return

- Function can return a value (like in maths).
- Function returned value can be assigned to a variable or act as an variable.
- The value is returned with a **return** statement.

return

```python
def add(x, y):
    return x + y

z = add(2, 4)
print(z)

print(add(3, 5))

print(add(6, z))
```

Regular expressions

Regular expressions

- Regular expressions are patterns that describe strings.
- They make use of special characters to define symbols or group of symbols that match the expression.
- Regular expressions can describe matching strings or their parts.

Regular expressions

- There are several tools that may be used to design and verify regular expressions.
    - https://regex101.com/ - a tool to validate and run regular expressions.
    - https://regexper.com/ - a tool to visualize regular expressions.

Regular expression syntax basics

- . - any character
- a - specific character (in this case a letter a)
- abc - specific string (in this case abc)
- a|b - character a **or** b
- a* - 0 or more letters a
- a+ - 1 or more letters a
- .* - 0 or more characters