Lesson 3

- Motivational Demo: How great python is
- Type str: Strings in Python
- Input/Output and str Formatting
- for loop over str
- OPTIONAL: while loops

Data Types & Operators - Revisiting

- Data types:
    - int
    - float
    - string
    - bool
    - list
    - dict
    - ...

```
i = 10
pi = 3.14
hello = "Hello"
hi = 'Hi'
b = True
```

Type str: Strings in Python

- String Literal: A string literal is a sequence of characters. In Python, this type is called str. Strings in Python start and end with a single quotes (') or double quotes ("). A string can be made up of letters, numbers, and special characters. For example:

    If a string begins with a single quote, it must end with a single quote. The same applies to double-quoted strings. You can not mix the type of quotes.
- String Operators

```
str1 + str2 concatenate str1 and str1

str1 * int1 concatenate int1 copies of str1

int1 * str1 concatenate int1 copies of str1
```

    Note: concatenate means to join together The * and + operands obey by the standard precedence rules when used with strings. All other mathematical operators and operands result in a TypeError.

In [3]:

```
str = 'Hello, how are you, today?'
print(str)

print("Hello, how is Jane's life today")
```

```
Hello, how are you, today?
Hello how is Jane's life today
```

In [4]:

```
print('ab' + 'c')
print('a' * 5)
print(10 * 'bc')
```

```
abc
aaaaa
bcbcbcbcbcbcbcbcbcbc
```

Input/Output and str Formatting

- Python has a built-in function named print that displays messages to the user. For example, the following function call displays the string "hello":

      ```
      print("hello")
      hello
      ```

- The print function may also be called with a mathematical expression for an argument

      ```
      print(3 + 7 - 3)
      ```

- Finally, print can take in more than one argument. Each pair of arguments is separated by a comma and a space is inserted between them when they are displayed. For example:

      ```
      print("hello", "there")
      print("Anser=", 3)
      ```

- String Escape Sequences

  Python has a special character called an escape character: . When the escape character is used in a string, the character following the escape character is treated differently from normal. The escape character together with the character that follows it is an escape sequence. The table below contains some of Python's commonly used escape sequences.

      ```
      \n  newline (ASCII linefeed - LF)
      \t  tab (ASCII horizontal tab - TAB)
      \\  backslash (\)
      \'  single quote (')
      \"  double quote (")
      ```

In [8]:

```python
print('hello\nPython')
print('Your file is stored at C:\\Home\\UserA')
print('This is John\'s house')
```

```
hello
Python
Your file is stored at C:\Home\UserA
This is John's house
```

Excerise 1 - Ask user for his first name, family name and age. Say him 'Hi' properly

- your programm should ask the user 3 questions about

      name
      family name
      age

  and print following text:

      Hello <first name> <second name>!
      I know, that you are <put age here> years old.

  you can only use print() function once in your code

In [16]:

```python
first_name = input('Plese enter your first name:')
family_name = input('Plese enter your family name:')
age = input('Plese enter your age:')

print('\nHello', first_name, family_name, '!\n', 'I know, that you are', age, 'y
ears old')
```

```
Plese enter your first name:John
Plese enter your family name:Ivanov
Plese enter your age:15

Hello John Ivanov !
 I know, that you are 15 years old
```

For loop over strings

- For Loops

  The general form of a for loop over a string is:

      for variable in str:
          statements

In [19]:

```
string = 'abc'

for letter in string:
    print(letter)
```

a
b
c

Excerise2 - Calculate 'a' letters in a string

- write a programm which asks a user to input a word and calculates how many occurences of letter 'a' are there

  for this you will need 'for loop over strings' and 'if' operator

In [32]:

```
word = input('Please enter a word:')

number_of_a = 0

for letter in word:
    #if letter == 'a' or letter == 'A':
    if letter.lower() == 'a':
        number_of_a = number_of_a + 1
        print('another occurence of "a" found...')

print('I have counted', number_of_a, 'occurence(s) of "a" in the word', word)
```

Please enter a word:Abracadabra
another occurence of "a" found...
another occurence of "a" found...
another occurence of "a" found...
another occurence of "a" found...
another occurence of "a" found...
I have counted 5 occurence(s) of "a" in the word Abracadabra

while loops

The general form of a while loop:

```
while expression:
    statements
```

In [33]:

```python
a = 0

while a < 10:
    print(a)
    a = a + 1
```

```
0
1
2
3
4
5
6
7
8
9
```

In [22]:

```
['Y', 'e', 's', 't', 'e', 'r', 'd', 'a', 'y']
```

Homework