Introduction to Python Programming
The Basics

Goals:

The goal of this exercise sheet is to experiment and to gain hands-on experience with Python using both scripts and the interactive interpreter! In this Lab, you will practice around the important notions of variables, basic built-in types and Input/Output and Control Structures.

If you have any question about the syntax or how to write a part of your code, please use the "memo", the Python documentation or any other resources.

For this course and its practical parts, you will use the PyCharm Edu Integrated Development Environment (IDE).

Recommandations for each lab:

- Start PyCharm
- Select `Create New Project`
- Do not keep the **'untitled'** default name. For each lab, we strongly recommend you to create a new PyCharm project for each Lab. Thus, your first work here is to create a new project name for instance `SIP_Lab_1` in PyCharm
- Click on the small blue arrow to display the options
- Select `New environment using virtualenv`
- Keep the default path in `Location`
- Also keep the default `Base Interpreter`
  NOTE: check that it is not Python 2.x if an old version of Python was previously installed on your PC, as we will use Python 3 at CS and for SIP.

Additional resources:

- To create a new PyCharm project:
  `https://www.jetbrains.com/help/pycharm/creating-and-running-your-first-python-project.html`
- To execute chunks of code in the console:
  `https://www.jetbrains.com/help/pycharm/loading-code-from-editor-into-console.html`
- To install Python packages:
  `https://www.jetbrains.com/help/pycharm/installing-uninstalling-and-upgrading-packages.html`

1 Introduction

Exercice 1 : Your First Program : Hello World !

Write a Hello World program which consists in printing *"Hello, World! My name is YOUR_NAME"* to the console. First, write this program using the interactive interpreter. Next, put the body of the program into a file named `hello.py` (that has to be added to your PyCharm project), and run the Hello World program as a Python script.

2 Variables

To know : Variables

Variables are used to store information in a computer's memory so that we can refer to them later anywhere in the program. A variable is a name for an object reference. The = symbol, known as the assignment operator, is used to assign a value to a variable.

```
>>> a = 1234
```

The statement `a = 1234` creates an `int` object whose value is 1234; it then binds the variable `a` to this new `int` object.
A variable is defined by :

- A type (that is implicit in Python)
- An identifier which is a sequence of letters, digits, and underscores, the first of which is not a digit. Identifiers can not be keywords of the python language (see the memo or `https://docs.python.org/3/reference/lexical_analysis.html#identifiers`).

Naming conventions :

- Readability is very important : variable names should be lowercase, with words separated by underscores as necessary to improve readability.
- Descriptive names are very useful.
- Avoid using the lowercase letter 'l', uppercase 'O', and uppercase 'I'.

Some examples.

```
>>> myFirstVartiable = 5 # Affects 5 to the variable a
>>> myFirstVariable * 2 # Use the variable
10
>>> other_variable = myFirstVariable + 3.5 # Another variable
>>> print(other_variable) # Display a variable's content
8.5
>>> type(myFirstVariable) # Check a variable type
<class 'int'> # myFirstVariable is an int (integer)
>>> type(other_variable)
<class 'float'> # other_variable is a float (floating-point number)
```

Exercice 2 : Reading some code

We consider the code below :

```
prenom = "John"
nom = "Doe"
age= 45
```

```
male = True

prenom = "Max"
x = nom + prenom
y = age + male
z = nom + prenom + y
```

1. Draw on a paper sheet the evolution of the variable during the program (object reference, type, value of the referred object...)
2. One of the line of the code is not correct and will lead to the type of Error : `TypeError: must be str, not int`. Which one ? Why ? Try to correct it !
3. Write the correct instruction to display on the standard output the following message : `The male, named prenom nom has age years`

Exercice 3 : Swapping variables

Exchange the values of the variables `a` and `b`. Be careful not to lose any value! You can do it in one line or three lines.

Exercice 4 : A Mad Libs game in Python

Mad Libs[1] is a phrasal template word game where one player prompts others for a list of words to substitute for blanks in a story, before reading the – often comical or nonsensical – story aloud. The game is frequently played as a party game or as a pastime. The goal of this exercice is to build a simple Mad Libs game in Python. Your program must :

- Display that we are playing to the Mad Libs Game
- Ask to the user, using the `input`[2] command to ask to the user to define a `place`, an `adjective`, a `person` and an `animal`.
- Then to write two different generated Mad Libs using the given variable values.

Figure 1: Example of a Mad Libs game

[1] `http://www.madlibs.com/`
[2] `https://docs.python.org/3/library/functions.html#input`

3 Data Types

To know : Data Types

A data type is a set of values and a set of operations defined on those values. Many data types are built into the Python language. The table below presents the basic built-in data types in Python, i.e. integers (`int`), floating-point numbers (`float`), sequence of characters (`str`) and true-false values (`bool`).

type	*set of values*	*common operators*	*sample literals*
`int`	*integers*	`+ - * // % **`	`99 12 2147483647`
`float`	*floating-point numbers*	`+ - * / **`	`3.14 2.5 6.022e23`
`bool`	*true-false values*	`and or not`	`True False`
`str`	*sequences of characters*	`+`	`'AB' 'Hello' '2.5'`

Basic built-in data types

Figure 2: Basic built-in data types

3.1 Strings

Exercice 5 : Printing a Tic Tac Toe

Write a program using `print()` that, when runs, prints out a tic-tac-toe board with asterisk (*) characters using only one print instruction?

Indice : to create a multi-line string, use 3 single quotes.

Exercice 6 : Print 90210

Write a program that, using only `a`, `b`, `c` and `d` as defined as below, prints `90210`.

```
1 a= 90
2 b = "10"
3 c = "11"
4 d = 0
```

Exercice 7 : String Formatting

Write a Python program that accepts an integer n and computes the value of $n + nn + nnn$.

3.2 Numbers : Python as a calculator

To know : Numbers

In python, 4 built-in datatypes can be used to represent numbers : `int`, `bool`, `float` and `complex`. The code below gives some examples of basic operations that can be achieved with python numbers.

```
1 >>> # Comments are ignored by Python and start by a #
2 >>> # Python can be used as a powerful calculator
3 >>> (7+5)*2
4 24
```

```
>>> 3 / 4 # / is the division on real numbers
0.75
>>> 3 // 4 # // is the division on integers
0
>>> (6**10 % 11) # ** is the exponent operator, % the modulo operator
1
>>> 1+0.00000000000000001 # Be careful of the limited precision!
1.0
>>> 0.1 + 0.1 + 0.1
0.30000000000000004
>>> .4e7 + 4.2e-4 # scientific notation of float
4000000.00042
>>> (4+5j) # complex can also be defined in cartesian notation
>>> type(4+5j)
<class 'complex'>
>>> (4+5j)+ (1-8j)
(5-3j)
>>> (4+5j).real # real part of a complex
4.0
>>> (4+5j).imag # imaginary part of a complex
5.0
```

Exercice 8 : Python Challenge 1

The objective of this exercise is to solve the first challenge of the Python Challenge game. The challenge is here : `http://www.pythonchallenge.com/pc/def/0.html`.

Write the python instruction that enables to solve this challenge.

Exercice 9 : Fahrenheit to Celsius

Write a program that, given a temperature F in degrees Fahrenheit, prints the temperature in degrees Celsius. Your answer must be a float.

Exercice 10 : Guessing a value

What is the output of the following code :

```
z = 2
z = z**2**3
print(z)
x = 4
x = (x**2)**3 + 6 - z / 4 * 2
print(x)
```

Exercice 11 : Floating-Point Representation

Type the following instructions on your python console and explain the obtained results.

```
print(1.8e308)
print(1e-325)
```

Exercice 12 : Π

Write a program that assigns to a variable , the variable `pi` that equals the constant π (up to `float` precision, 14 digits here) using the `math` library (using `from math import pi`). Then, test if 22/7 is greater or lesser than it and write the relative error of this approximation ?

3.3 Booleans

To know : Booleans

Booleans is a type, consisting only of the values `True` and `False`. You can create a boolean expression (expression that produces a boolean) with comparison operators and with the specific operators `and`, `or` and `not`. Examples are given below.

```
>>> a = (3 <= 5) # Use ==, !=, <, <=, >, >= to compare numbers
>>> a
True
>>> type(a)
<class 'bool'> # a is a bool (Boolean)
>>> a == False # Be careful not to confuse = and ==
False
>>> not( (True or False) and True) # Boolean operators
False
```

3.4 Number systems

Exercice 13 : Number systems in python

The number systems refer to the numbers of symbols or characters used to represent any numerical value. By default, in python we use *decimal* litterals to represent numbers but other representations are possible as seen below.

```
>>> a= 0b1001111 # binary litteral for 79
>>> print(a)
79
>>> bin(79) # binary conversion of a decimal number
'0b1001111'
>>> int(bin(79),2) #decimal conversion of a binary number
79

>>>a=0o117 # octal litteral for 79
>>> print(a)
79
>>> oct(79) # octal conversion of a decimal number
'0o117'
>>> int(oct(a),8) #decimal conversion of an octal number
79

>>>a= 0x4f# hexadecimal litteral for 79
>>> print(a)
79
>>> hex(79) # octal conversion of a decimal number
'0x4f'
>>> int(hex(79),16) #decimal conversion of an hexadecimal number
79
```

Write a program that assigns four different variables assigned to the integer 25 in decimal, binary, octal and hexadecimal representations respectively, that displays them and that verifies that they correspond to the same value.

4 Statements Blocks and Control Structures

In programming, a sequence of statements is referred as *flow of control*. In python, indentation is meaningful and all the statements that are at the same level of indentation are in the same block of instructions. In imperative programming, **control structures** are used to execute blocks of statements according to certain conditions (**conditionals**) or to execute blocks of statements multiple times (**loops**). In python, the sequence of statements within the control structure needs to be indented one level further.

4.1 Conditional statements

Conditional statements are used to Execute a block of statements only if a certain condition is true (boolean tests). Otherwise, the statements are skipped. Examples are given below.

```
x = -3
a,b,c = -1,0,1
val = 0

# To start a condition structure, we use the keyword "if" followed by a boolean
    expression and a colon ":"
if x > c:
    val = "greater than 1" # The code to be executed needs to be indented

# We use the "elif" keyword to test another boolean expression
elif x > b:
    val = "close to 0"

# If none of the boolean expressions are true, we have the "else" keyword for default
     behaviour
else:
    val = "smaller than -1"
print(val)
```

Exercice 14 : Maximum

Write a program that asks to the user to enter two integers, that verifies the type of the input values and that converts them into integers and then that computes and prints the max between the two integers.

Exercice 15 : Rock Paper Scissors

The goal of this exercise is to write a program that plays to rock paper scissors with you. Two different versions can be written : a simple one that just plays to the game and a second one that also counts the score of the player and that takes into account the exit of the game.

4.2 Loops

A loop is a control structure used to repeat the same operation several times (a known or unknown number of times). In python we have two different types of loops `for` and `while`.

The `for` loop is used to go through the items of a collection (groups of objects). It is very useful for going through iterable objects of to iterate a known number of times.

Exercice 16 : Calculator

1. Write a `for` loop printing the value of $\sum_{n=1}^{100} \frac{1}{n^2}$.
2. The limit of $\sum_{n=1}^{k} \frac{1}{n^2}$ is $\frac{\pi^2}{6}$. Write a while loop block that computes the sum until reaching a precision of 10^{-6} using `from math import pi`.

Exercice 17 : Mysterious program

Explain what the following programs do. Give the expected results for the two programs for the entry 10.

```
n = int ( input (" Donnez un nombre : "))
for i in range (2, n):
    for j in range (2, i):
        if i % j == 0:
            break
    else:
        print(i)
```

```
n = int ( input (" Donnez un nombre : "))
for i in range (2, n):
    for j in range (2, i):
        if i % j == 0:
            continue
    else:
        print(i)
```

Exercice 18 : Fizz, Buzz, FizzBuzz!

An interview exercise for programmers.

Fizz buzz is a group word game for children to teach them about division in which players take turns to count incrementally, replacing any number divisible by three with the word "fizz", and any number divisible by five with

the word "buzz" and any number divisible by five and three by "fizzbuzz". This game has been transformed into an interview screening device for computer programmers[3]. Just test yourself !

Write a program that prints the numbers from 1 to 500. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.

Exercice 19 : The Euler Project - Problem number 1

Project Euler [4] is a series of challenging mathematical/computer programming problems that require more than just mathematical insights to solve. You will solve the problem number 1.

If we list all the natural numbers below 10 that are multiples of 3 or 5, we get 3, 5, 6 and 9. The sum of these multiples is 23.

Write a program that finds and writes the sum of all the multiples of 3 or 5 below 1000.

Exercice 20 : Armstrong Number

A positive integer of n digits is called an Armstrong number (or narcissistic number[5]) of order n (order is number of digits) if :

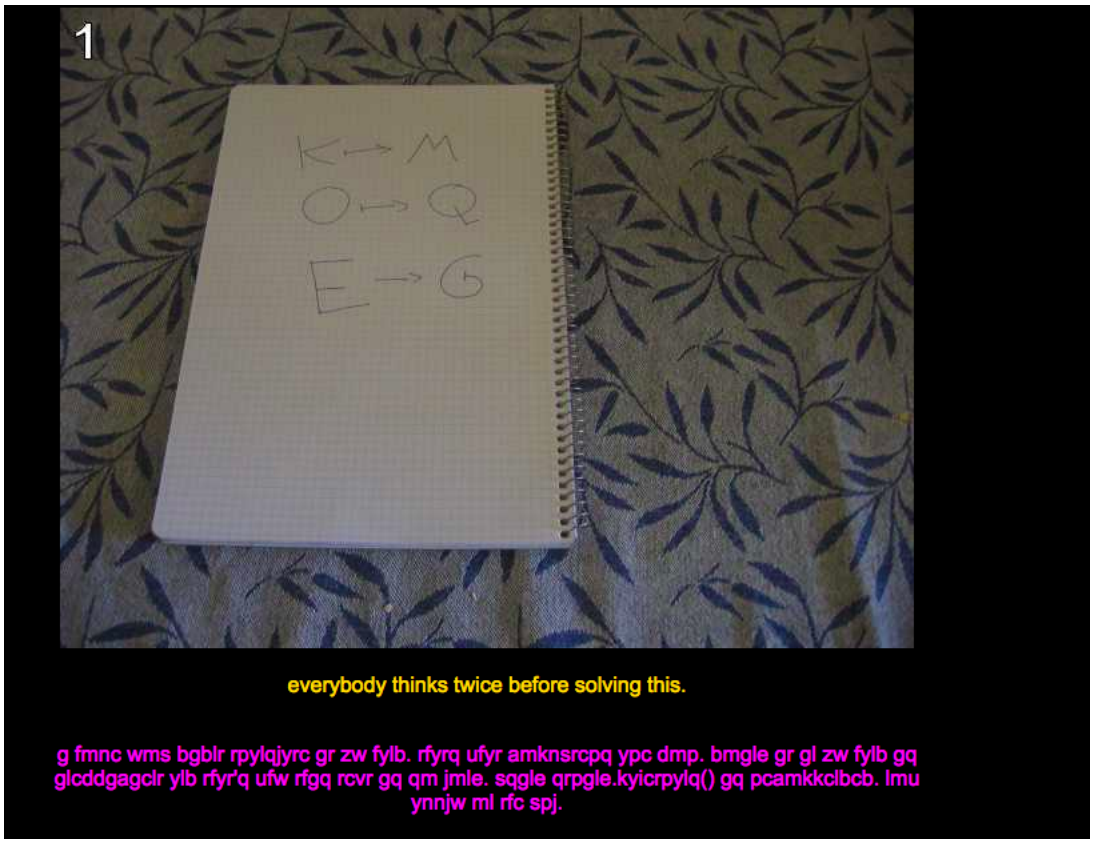
$$abcd... = a^n + b^n + c^n + d^n + ...$$

For instance, 153 is an Armstrong number since $153 = 3 * 3 * 3 + 5 * 5 * 5 + 1 * 1 * 1$

Write a program that checks Armstrong numbers in certain interval that will given by the user.

Exercice 21 : Python Challenge 2

he objective of this exercise is to solve the second challenge of the Python Challenge game. The challenge is here : `http://www.pythonchallenge.com/pc/def/map.html`.



[3] https://imranontech.com/2007/01/24/using-fizzbuzz-to-find-developers-who-grok-coding/
[4] https://projecteuler.net/
[5] https://en.wikipedia.org/wiki/Narcissistic_number

Exercice 22 : As far as possible !

You have finished the LAB, try to go as far as possible into the Python Challenge !