

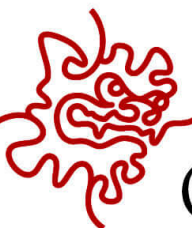


Skill Pill: Intro to Programming

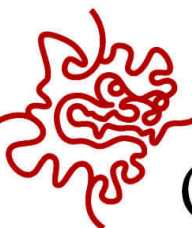
Spyder, variables and lists
Jeremie Gillet



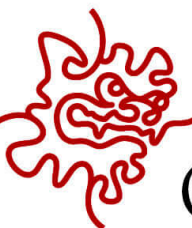
- Python's strengths and weaknesses
- Spyder IDE, scripts and console
- Variables
- Lists



- Easy to read, write and learn
- Very flexible
- Many libraries (scipy, numpy, matplotlib, data science, machine learning, games, web...)
- Well documented, all mistakes have been made
- Slow



- There are small visible differences between Python 2 and 3
- Currently, both versions co-exist
- Python 2 has been deprecated since January 2020, please don't use it
- From now on, Python = Python 3



- Anaconda is an open source distribution of Python and R for data science
- Anaconda manages thousands of packages
- Spyder is an open-source Integrated Development Environment (IDE) for scientific programming in Python
- Spyder helps you write Python code

Variable explorer, file explorer, help

Scripts

```

1
2 #Ex: give the value 3 to a, 5 to b and 0 to c
3 a, b, c = 3, 5, 0
4
5
6 #Ex: divide a by b and b by a
7 a/b
8 b/a
9
10 #Ex: same, with integer division
11 a//b
12 b//a
13
14 #Ex: divide a by c, with float and integer division
15 #a/c
16 #a//c
17
18 #Ex: calculate b modulo a and b modulo c
19 b%a
20 #b%c
21
22 #Ex: calculate b to the power a
23 b**a
24
25 #Ex: set a,b,c,d to 1,2,4,12 and find 1 or 2 ways to combine all 4 variables with +-*// to make
26 #example: a*(c-b)*d
27 a,b,c,d = 1,2,4,12
28 a*(c-b)*d
29 (d-c)*(a+b)
30 (a+b)*c+d
31 a*c*d//b
32
33 #Ex: set a to 7
34 a = 7
35
36 #Ex: check the type of a
37 type(a)
38
39 #Ex: without touching the "7" key on your keyboard, set b to the float value of 7
40 b = 1.0 * a
41
42 #Ex: check that b is a float
43 type(b)
44
45 #Ex: guess what the type of a+b should be
46 type(a+b)
47
48 #Ex: what is the largest int you can define?
49 #Ans: something big enough to use all your computer's memory

```

Name	Type	Size	Value
a	str	1	I like Skill Pills! I like Skill Pills! I like Skill Pills! I like Ski ...
b	float	1	7.0
c	int	1	0
d	int	1	12
fasta_ex	str	1	>sp Q9VB54 EGLY_MOUSE Pikichurin S=Mus mus ... OK=10000 Q=Egflam ...
s_ex5	str	1	# this is a comment line # this is another comment line
x	str	1	hi

File explorer Variable explorer Help

IPython console

Console 1/A

```

In [80]: ex5()
----- EXERCICE 5 -----
Traceback (most recent call last):

  File "<ipython-input-80-fe859a23b026>", line 1, in <module>
    ex5()

  File "/Users/jie/OneDrive - OIST/OIST/Events/Bootcamp 2018/Exercices/Day 1/Day 1 - Session 3b.py", line 84, in ex5
    splits = [line.split("\t") for line in s_ex5.split("\n") if line[0]!=""#"]

  File "/Users/jie/OneDrive - OIST/OIST/Events/Bootcamp 2018/Exercices/Day 1/Day 1 - Session 3b.py", line 84, in <listcomp>
    splits = [line.split("\t") for line in s_ex5.split("\n") if line[0]!=""#"]

IndexError: string index out of range

In [81]:

```

IPython console History log

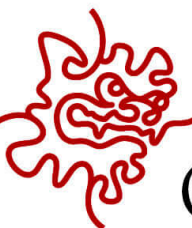
Permissions: RW End-of-lines: LF Encoding: ASCII Line: 1 Column: 1 Memory: 76 %

Console



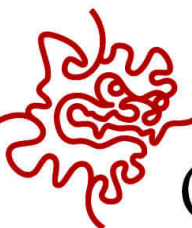
Let's try a script

- File > New File...
- Write `print("Hello World")`
- Click Run File (F5)
- Ok to the dialog box
- Look at the console



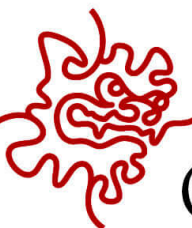
Let's try the console

- Click on the console
- Write `print("Hello World")`
- Press enter
- Try up arrow, ctrl+a, ctrl+e, escape



Just so you know...

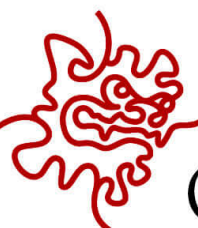
- We could do the same with less fancy tools
- Ex: TextEdit + terminal
- But: no user interface, syntax highlighting, autocomplete, saving variables...
- This is most likely how you would use Python scripts on Deigo



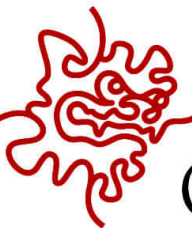
Spyder Shortcuts

- Shortcuts will save you seconds!
- Google “Spyder shortcuts” for more
- Mac people: Substitute Ctrl for Command

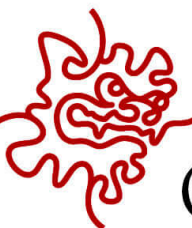
Ctrl + I	Toggle comments
Ctrl + 4 (5)	Add (remove) block comment
Ctrl + space / TAB	Code completion
F5	Run file
F9	Run selection



- Single line comments:
 - `#this is a comment`
- Multi-line comments:
 - `"""
these
are
comments
"""`
- Comment **EVERYTHING!!!!**



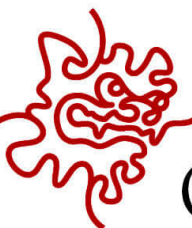
- Variables let you store values by giving them a name
 - `x = 2`
 - `my_string = "Hello"`
 - `a, b = 0, 1`
 - `y = x`
- There are rules for naming variables ([a-zA-z0-9_] only, cannot begin by a number, should be easy to understand...)



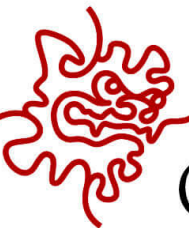
Types of Variables

Type	Python name	Possible values
Boolean	bool	True, False
Integer	int	..., -2, -1, 0, 1, 2, ...
Floating Point	float	..., 1E-8, 0.0, -0.001, ...
Text (String)	str	“”, “Hi”, ‘Mom’, ...
Complex	complex	3.1 + 7j, ...
...		

Check types by using `type ()`

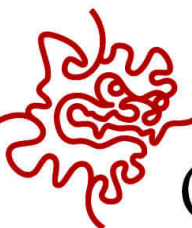


- Some variables can change types
 - `x = 2` `# will be an int`
 - `x = float(2)` `# will be a float`
 - `x = 1.0 * 2` `# will be a float`
 - `x = int(2.99)` `# will be 2`
 - `x = str(2)` `# will be a string`
 - `x = int("Hi")` `# will be an error`
 - `x = bool("Hi")` `# what will it be?`

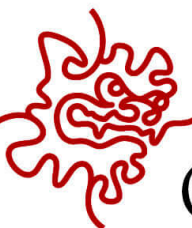


- There are two booleans (`bool`): `True` and `False`
- You can use them to make decisions
- You can manipulate them with boolean logic operators: `not`, `and`, `or`
- You can create them in various ways, including comparisons operators: `<`, `>`, `<=`, `==`, `!=`, ...

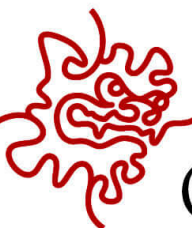
- Basic math: `+`, `-`, `*`, `/`, `//`, `**`, `%`
- Built-in functions: `abs()`, `min()`,
`max()`, `round()`, `bin()`, `sum()`,
...
- String operations: `+`, `*`



- The pattern `count = count + 1` is extremely common and is used to increase the value of a variable by 1
- It is equivalent to `count += 1`
- You can use that on many operators: `+=`, `-=`, `*=`, `/=`, ...



- `x = input("Type something: ")`
- Whichever you type until you press enter will be saved in `x`
- You can cast `x` into something else



Exercise

variables.py

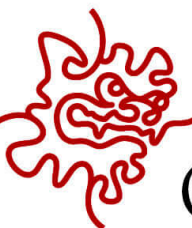
Test your answers in the console

There are three exercise levels:



Time: 20 minutes

- Lists are a type of data structure, in the family of iterators
- Lists can contain any type of values, numerical, strings, other lists...
- `my_list1 = [1, 2, 3]`
- `my_list2 = ["Hi", 3, [3.0, False], my_list1]`



The background of the slide is filled with a dense, scattered pattern of red and white capsules, resembling a field of pills or a rain of medicine.

Group Discussion

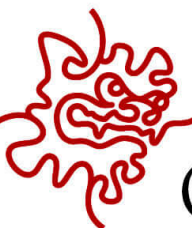
**What should you be able to
do with a list?**

3 minutes

Retrieving list elements

index	0	1	2
my_list =	[10,	20,	30]
index'	-3	-2	-1

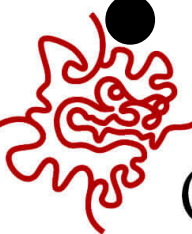
- `element = my_list[index]`
- The index must be integer, can be negative
- Indexing starts at 0
- For lists in lists: `my_list[i1][i2]`



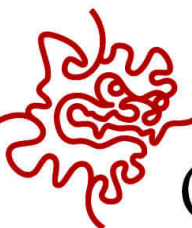
Slicing lists

index	0	1	2	3	4	5	6	
my_list	=	[1,	2,	3,	4,	5,	6,	7]

- Most general: `my_list[from : to : by]`
 - `my_list[1:6:2]` # will be `[2, 4, 6]`
- `to` is NOT inclusive
- If you omit values, then default to `[0: length of list: 1]`
 - `my_list[3:]` # will be `[4, 5, 6, 7]`
 - `my_list[:3]` # will be `[1, 2, 3]`
 - `my_list[::4]` # will be `[1, 5]`
 - `my_list[:]` # will be the same as `my_list`

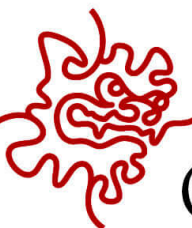


- `val in x` # checks if `x` contains `val`
- `del x[i]` # deletes the `i`-th element of `x`
- `x.append(val)` # adds `val` to the end of `x`
- `x.insert(i, x)` # inserts an element at index `i`
- `x.remove(val)` # removes the first occurrence of `val`
- `x.count(val)` # counts the number of time `val` is in `x`

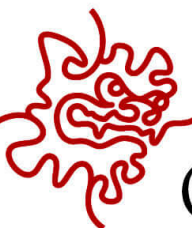


More list methods

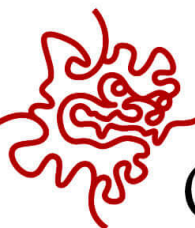
- `x + y` # concatenates `x` and `y`
- `len(x)` # Length of `x`
- `max(x)` # maximum value in `x` (`min` works too)
- `sum(x)` # sum of values in `x`
- `x.sort()` # sorts `x` in ascending order
- `sorted(x)` # returns a sorted version of `x`
- `x.index(val)` # returns index of first occurrence of `val` in `x`



- A tuple is simpler list (really, an iterator)
- `t = (1, 2, 3)`
- `t[1:3] # returns [2, 3]`
- `len(t) # returns 3`
- ...
- Tuples are immutable: cannot use `append`, `del`, `insert`, ...



- A range is an iterator, often used for loops
- `range(from, to, by)` # creates a range
- `list(range(1,5))` # returns [1, 2, 3, 4]



The background of the slide is filled with a dense, scattered pattern of red and white capsules, resembling a rain of pills.

Exercise

`lists.py`

Solve the problem within the script
and test your solution by running it

```
Time = Time[now:]
```