

# Module 2: Algorithm and Data Structures Lab

David Leoni  
teaching assistant  
david.leoni [AT] unitn.it

University of Trento

23 Nov 2016  
v6.0

This work is licensed under a Creative Commons Attribution 4.0 License

## Today

- Implement ComplexNumber

## Next lab

- Monday 28th Nov, 16:00-18:00 Room 215

## Course website

### Theory

See Alberto Montresor website:

<http://cricca.disi.unitn.it/montresor/teaching/scientific-programming>  
(<http://cricca.disi.unitn.it/montresor/teaching/scientific-programming>)

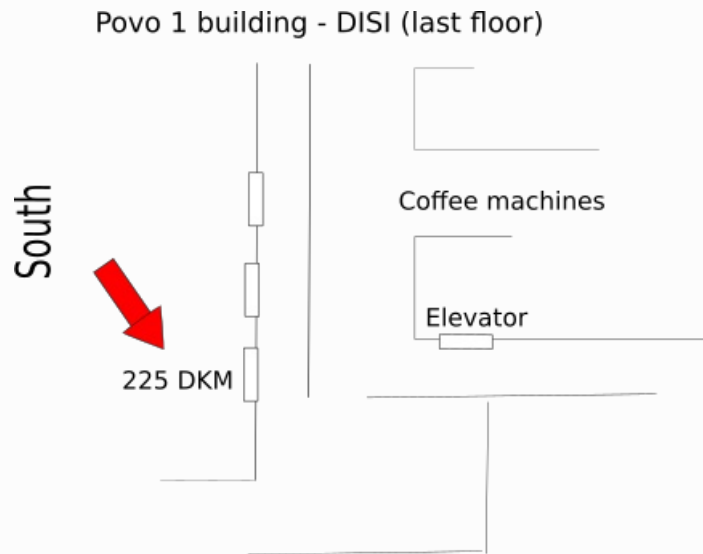
### Lab

[davidleoni.github.io/algolab](https://davidleoni.github.io/algolab) (<https://davidleoni.github.io/algolab>)

## Office hours

You can schedule a meeting by emailing me at david.leoni [AT] unitn.it , more or less I'm available every day until 19.00

Then you will find me in Povo 1 building at DISI, in room 225 DKM :



## Chapters

Worksheets are meant to be used online - pdf quality is not very good, if they result unreadable please tell me

0. [Testing \(testing.html\)](#) (pdf (pdf/testing.pdf))

1. [Lists \(lists.html\)](#) (pdf (pdf/lists.pdf))

2. [Data Structures \(data-structures.html\)](#) (pdf (pdf/data-structures.pdf))

## Commandments

**WARNING: If you don't follow the Commandments, bad things happen!**

### 1) You shall test!

To run tests, enter the following command in the terminal:

```
python -m unittest my-file
```

**WARNING:** In the call above, **DON'T** append the extension *.py* to *my-file*  
**WARNING:** Still, on the hard-disk the file **MUST** be named with a *.py* at the end, like *my-file.py*

**2. You shall also write on paper!**

**3. You shall copy *\*exactly the same\** function definitions as in the exercises!**

For example don't write :

```
def MY_selection_sort(A):
```

**4. You shall never ever reassign function parameters:**

```
def myfun(i, s, L, D):
```

```
    # You shall not do any of such evil, no matter what the type of the parameter is:
    i = 666          # basic types (int, float, ...)
    s = "666"        # strings
    L = [666]         # containers
    D = {"evil":666}  # dictionaries

    # For the sole case of composite parameters like lists or dictionaries,
    # you can write stuff like this IF AND ONLY IF the function specification
    # requires you to modify the parameter internal elements (i.e. sorting a list
    # or changing a dictionary field):

    L[4] = 2          # list
    D.my_field = 5     # dictionary
```

**5. You shall use *return* command only if you see written *return* in the pseudocode!**

If there is no `return` in the pseudocode, the function is intended to return `None`. In this case you don't even need to write `return None`, as Python will do it implicitly for you.

# Slides

## Lab 1 Slides

3 Nov 2016

### Lab Goals

- Going from theory taught by Prof. Alberto Montresor to implementation
- Pseudo code --> Python

### How

- Hands-on approach
- Performance considerations
- Focus on correct code
- Few Python functions

### Lab Midterm?

Probably not. Still, will provide exam example.

## Lab 2 Slides

Date: Nov 11th, 2016

- More practical than last time!
- Finish `selection_sort` and gap implementation
- midlab pause ;-)
- implement a Python class

## Lab 3 Slides

Nov 17th, 2016

- Recursion
  - `gap_rec`, `binary_search_rec`
- `binary_search_iter`
- Will give you more tests
- Write also on paper!
- Copy *exactly the same* function definitions!
  - For example don't write `def MY_selection_sort(A):`
- use `return` command *only* if you see written `return` in the pseudocode!
  - If there is no `return` in the pseudocode, the function is intended to return `None`. In this case you don't even need to write `return None`, as Python will do it implicitly for you.

## Lab 4 Slides

Nov 18th, 2016

- Divide et Impera
  - `binary_search_iter`
- Implement `ComplexNumber`

New Commandment:

You shall never ever reassign function parameters:

```
def myfun(L, i, s):  
  
    # You shall not do any of this evil:  
    L = [666]  
    i = 666  
    s = "666"
```

Previous commandments:

- You shall also write on paper!
- You shall copy exactly the same function definitions as in the exercises!
- For example don't write `def MY_selection_sort(A)`:
- You shall use `return` command only if you see `Written return` in the pseudocode!
- If there is no `return` in the pseudocode, the function is intended to return `None`. In this case you don't even need to write `return None`, as Python will do it implicitly for you.

## Resources

- Online book: [Problem Solving with Algorithms and Data Structures using Python](http://interactivepython.org/runestone/static/pythonds/index.html) (<http://interactivepython.org/runestone/static/pythonds/index.html>) by Brad Miller and David Ranum
- [Theory slides](http://cricca.disi.unitn.it/montresor/teaching/scientific-programming/slides/) (<http://cricca.disi.unitn.it/montresor/teaching/scientific-programming/slides/>) by Alberto Montresor
- Will try to be consistent with [other lab module notes](http://disi.unitn.it/~teso/courses/sciprog/index.html) (<http://disi.unitn.it/~teso/courses/sciprog/index.html>) of Stefano Teso and Toma Tebaldi
- [PythonTutor](http://www.pythontutor.com/visualize.html#mode=edit) (<http://www.pythontutor.com/visualize.html#mode=edit>), a visual virtual machine (*very useful!* can also be found in examples inside the book!)
- [Source code](https://github.com/DavidLeoni/algolab) (<https://github.com/DavidLeoni/algolab>) of these worksheets ([download zip](https://github.com/DavidLeoni/algolab/archive/master.zip) (<https://github.com/DavidLeoni/algolab/archive/master.zip>)), in [Jupyter Notebook](http://jupyter.org) (<http://jupyter.org>) format.
- The internet ....

## Editors

- [Jupyter Notebook](http://jupyter.org) (<http://jupyter.org>): Nice environment to execute Python commands and display results like graphs. Allows to include documentation in Markdown format
- [Spyder](https://pythonhosted.org/spyder/) (<https://pythonhosted.org/spyder/>): Should be a fine editor, although I haven't used it in a long time

In [6]:

In [6]: