Introduction to Scientific Python

Application to Oceanography

C. Troupin, E. Mason

SOCIB, IMEDEA

Cádiz, 27-29 January 2016

Table of contents

Table of contents

Introduction:

What? Why? How?

What is Python?

Programming language:

1. interpreted — instructions executed directly
2. dynamically typed — type checking at run-time
3. object-oriented — classes, objects, methods, ...
4. high-level — strong abstraction

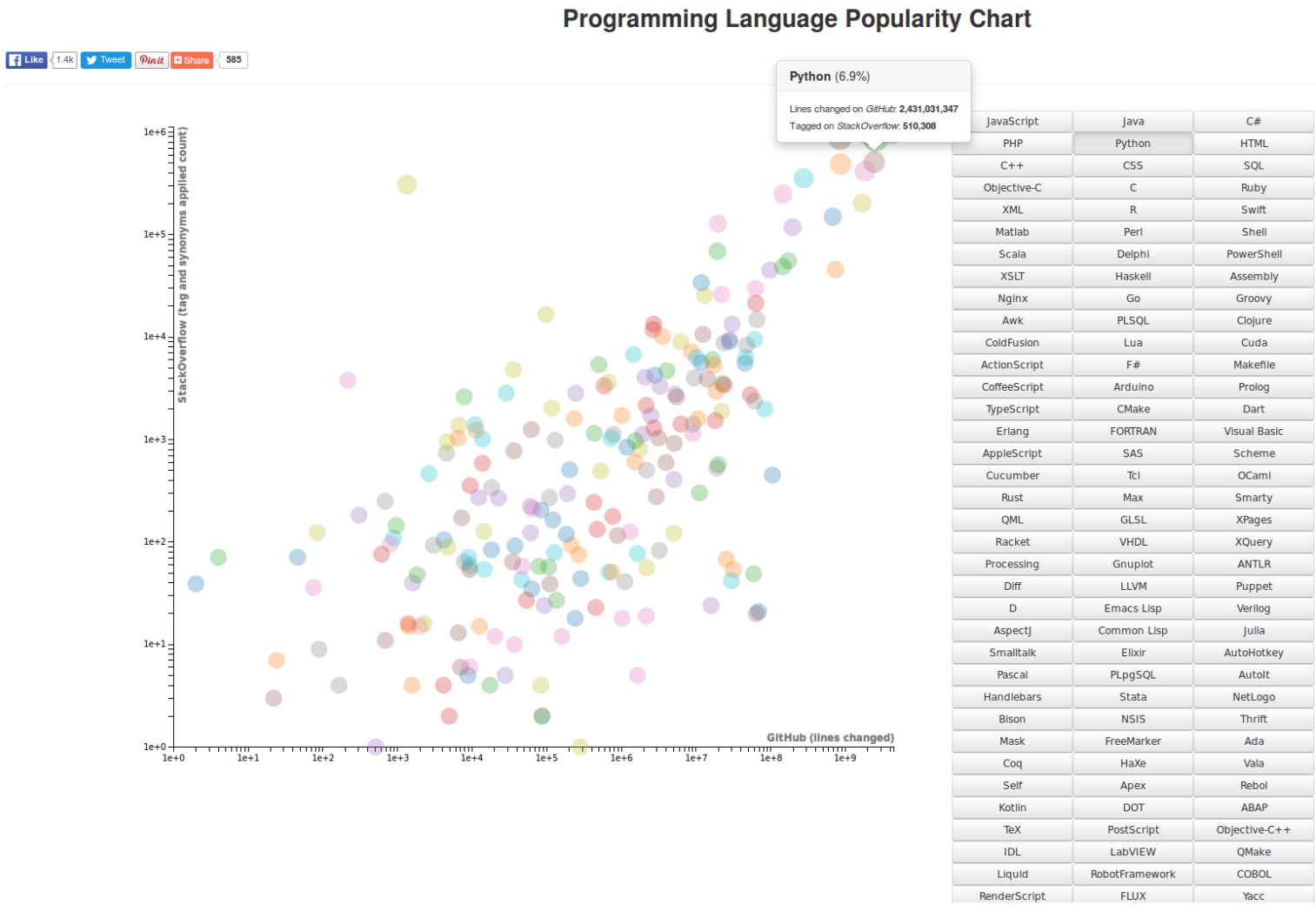
https://www.python.org

Why Python?

1. Simple, easy to learn syntax
2. Open
3. Large user community

doc, support, packages

Why Python?



Source: http://langpop.corger.nl/

Python vs. Matlab

Python	Matlab
General	
programming language	programming language + numerical computing environment
open	proprietary algorithms
general purpose	linear algebra
Indexing	
a[0]	a(1)
a[-1]	a(end)
a[::2]	a(1:2:end)
Functions	
a.max()	max(a)
a.shape()	size(a)

Numpy for Matlab users:
https://docs.scipy.org/doc/numpy-dev/user/numpy-for-matlab-users.html

Quick example: hello.py

Quick example: hello.py

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
'''
This function prints "Hello world"
'''

def hello():
  print "Hello world"
  return

def main():
  hello()

if __name__ == '__main__':
  main()
```

Quick example: hello.py

1. Try to document you code

Quick example: hello.py

1. Try to document you code
2. Use

```
# -*- coding: utf-8 -*-
```

   if you're using non-ascii characters

Quick example: hello.py

1. Try to document you code
2. Use

```
# -*- coding: utf-8 -*-
```

   if you're using non-ascii characters

3. In Python 3:

```
print("Hello world!")
```

Quick example: hello.py

1. Try to document you code
2. Use

```
# -*- coding: utf-8 -*-
```

   if you're using non-ascii characters
3. In Python 3:

```
print("Hello world!")
```

4. Indentation matters!

A few definitions

Object: Python's abstraction for data
https://docs.python.org/2/reference/datamodel.html#index-0

Function: series of statements which returns some value to a caller
https://docs.python.org/2/glossary.html#term-function

Module: file containing Python definitions and statements
https://docs.python.org/2/tutorial/modules.html

Class: logical grouping of data and methods (functions)
https://docs.python.org/2/tutorial/classes.html

Data types

1. Numbers

Example:

```
g = 9.81
h = 4.135667662e-15
```

Data types

1. Numbers
2. String

Example:

```
name = "Rickman"
s = "this is a string"
```

Data types

1. Numbers
2. String
3. List

Example:

```
list = ['one', 2, 'three', 'four', 5]
```

Data types

1. Numbers
2. String
3. List
4. Tuple

Example:

```
tuple1 = ('one', 2, 'three', 'four', 5)
tuple2 = (1, '1', 'one', [1, 2], (1, 2, 3))
```

Tuples are *immutable* (fixed value)

Data types

1. Numbers
2. String
3. List
4. Tuple
5. Dictionary

Example:

```
email = {'Evan': 'emason@imedea.uib-csic.es',\
         'Irene': 'irene.laiz@uca.es',\
         'Charles': 'ctroupin@socib.es'}
email['Irene']
'irene.laiz@uca.es'
```

Data types

1. Numbers
2. String
3. List
4. Tuple
5. Dictionary

More details:
https://docs.python.org/2/tutorial/datastructures.html#

Resources

Web:

- https://docs.python.org/2.7/tutorial/index.html
- https://developers.google.com/edu/python/introduction?hl=en
  tutorial + exercises
- http://www.python-course.eu
- http://www.learnpython.org online code execution
  https://pythonprogramming.net
- https://www.gitbook.com/book/djangogirls/djangogirls-tutorial

Resources

Learning platforms:

- Programming Foundations with Python Learn Object-Oriented Programming (7 weeks)
- Code Academy (13 hours)

Resources

Youtube:

- Python Beginner Tutorial (For Absolute Beginners) (4 parts)
- Google Python Class (7×30 minutes)
- Zero to Hero with Python (11 hours)

Resources

Books:

- *Learn Python the hard way*, Z.A. Shaw, 2013
  http://learnpythonthehardway.org/book/
- *Learning Python, 5th Edition*, M. Lutz, 2013
- *Python Programming: An Introduction to Computer Science*, J.M. Zelle, 2002

Complete list: https://wiki.python.org/moin/PythonBooks

Python 2 vs Python 3

Some differences:

- Print function
- Integer division
- Unicode
- ...

Python 3.x = present and future of the language

More details:

Python 2 or Python 3

Will Scientists Ever Move to Python 3?

Objectives of the course

1. Use Python to solve oceanography-related problems

Objectives of the course

1. Use Python to solve oceanography-related problems
2. Read/write various types of files

Objectives of the course

1. Use Python to solve oceanography-related problems
2. Read/write various types of files
3. Generate high-quality figures

Objectives of the course

1. Use Python to solve oceanography-related problems
2. Read/write various types of files
3. Generate high-quality figures
4. Find resources on the internet for specific problems

Objectives of the course

1. Use Python to solve oceanography-related problems
2. Read/write various types of files
3. Generate high-quality figures
4. Find resources on the internet for specific problems

What we won't (can't) do:
teach you how to be a good programmer

About the trainers

Evan Mason
Oceanographer
Post-doctoral researcher at IMEDEA
10-year experience with Python

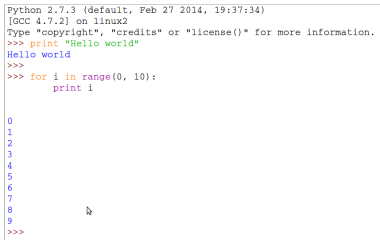
About the trainers

Evan Mason
Oceanographer
Post-doctoral researcher at IMEDEA
10-year experience with Python

Charles Troupin
Engineer, oceanographer
Head of SOCIB Data Center
2.5-year experience with Python

Structure of the course

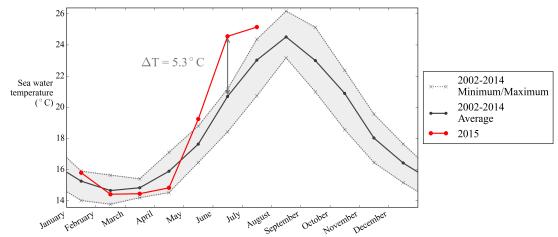
1. Reading/writing



Structure of the course

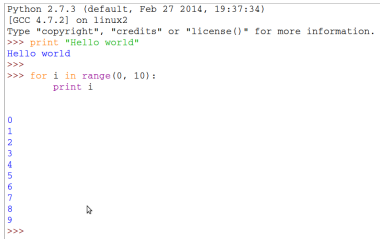
1. Reading/writing
2. Time series

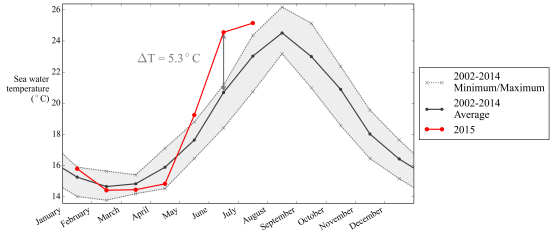
```
Python 2.7.3 (default, Feb 27 2014, 19:37:34)
[GCC 4.7.2] on linux2
Type "copyright", "credits" or "license()" for more information.
>>> print "Hello world"
Hello world
>>>
>>> for i in range(0, 10):
        print i


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

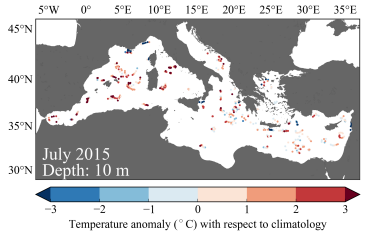


Structure of the course

1. Reading/writing
2. Time series
3. 2-D fields

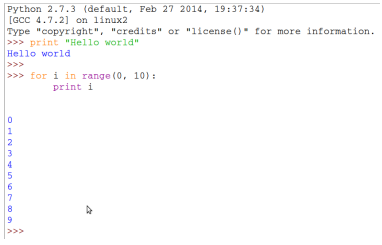


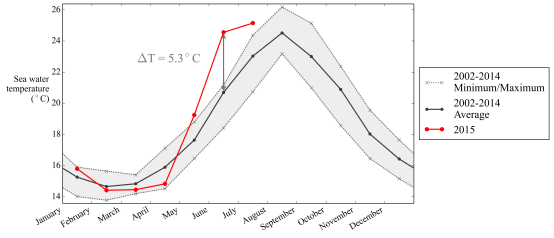


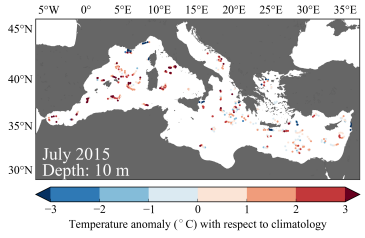


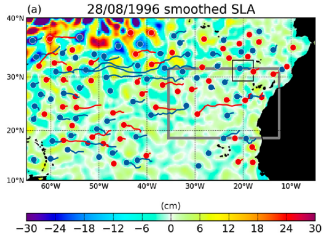
Structure of the course

1. Reading/writing
2. Time series
3. 2-D fields
4. Functions, classes, modules









Installation & use

Installing Python

Hard way: download source and compile:
https://www.python.org/downloads/

Installing Python

Hard way: download source and compile:
https://www.python.org/downloads/

Normal way: use installer or package:

- Windows:
  https://www.python.org/downloads/windows/
- Mac OS:
  https://www.python.org/downloads/mac-osx/
- Linux: package manager:
  python2.x and python2.x-dev packages

Installing Python

Hard way: download source and compile:
https://www.python.org/downloads/

Normal way: use installer or package:

- Windows:
  https://www.python.org/downloads/windows/
- Mac OS:
  https://www.python.org/downloads/mac-osx/
- Linux: package manager:
  python2.x and python2.x-dev packages

Easy way: Python distributions such as:

Anaconda	free
Enthought Canopy	free and commercial
Python(x,y)	free, Windows only

+ others

Installing modules

Example: SciPy (http://www.scipy.org/):
mathematics, science, and engineering

Installing modules

Example: SciPy (http://www.scipy.org/):
mathematics, science, and engineering

Easy way: Windows, Linux, Mac:
use Scientific Python distribution

Installing modules

Example: SciPy (http://www.scipy.org/):
mathematics, science, and engineering

Easy way: Windows, Linux, Mac:
use Scientific Python distribution

Intermediate: Linux, Mac: install package

- Linux: package manager
- Mac: MacPorts, Homebrew

Installing modules

Example: SciPy (http://www.scipy.org/):
mathematics, science, and engineering

Easy way: Windows, Linux, Mac:
use Scientific Python distribution

Intermediate: Linux, Mac: install package

- Linux: package manager
- Mac: MacPorts, Homebrew

Harder: build from source

```
$ python setup.py install
```

Installing modules

Example: SciPy (http://www.scipy.org/):
mathematics, science, and engineering

Easy way: Windows, Linux, Mac:
use Scientific Python distribution

Intermediate: Linux, Mac: install package

- Linux: package manager
- Mac: MacPorts, Homebrew

Harder: build from source

```
$ python setup.py install
```

Better: pip (https://pypi.python.org/pypi/pip)

Installing modules

Example: SciPy (http://www.scipy.org/):
mathematics, science, and engineering

Easy way: Windows, Linux, Mac:
use Scientific Python distribution

Intermediate: Linux, Mac: install package

- Linux: package manager
- Mac: MacPorts, Homebrew

Harder: build from source

```
$ python setup.py install
```

Better: pip (https://pypi.python.org/pypi/pip)

Avoid: mixing installation methods

Using `pip` to manage modules

`pip` = recommended tool for installing Python packages

Using `pip` to manage modules

`pip` = recommended tool for installing Python packages

Installation:

- Included in recent Python version
- Otherwise: download and run `get-pip.py`
  https://pip.pypa.io/en/stable/installing/#install-pip

```
$ python get-pip.py
```

Using `pip` to manage modules

`pip` = recommended tool for installing Python packages

Installation:

- Included in recent Python version
- Otherwise: download and run `get-pip.py`
  https://pip.pypa.io/en/stable/installing/#install-pip

```
$ python get-pip.py
```

Usage:

- Install latest version + dependencies:

```
$ pip install Package
```

- Specify exact version:

```
$ pip install Package==x.y.z
```

- Specify minimum version:

```
$ pip install 'Package>=x.y.z'
```

More about `pip`

- Uninstall packages:

```
$ pip uninstall
```

More about `pip`

- Uninstall packages:

```
$ pip uninstall
```

- List installed packages:

```
$ pip list
```

```
aptoncd (0.1.98-bzr117-1.2)
backports.ssl-match-hostname (3.4.0.2)
basemap (1.0.7)
...
xhtml2pdf (0.0.6)
zope.interface (3.6.1)
```

More about `pip`

- Uninstall packages:

```
$ pip uninstall
```

- List installed packages:

```
$ pip list
```

- Output installed packages in requirements format:

```
$ pip freeze
```

```
aptoncd===0.1.98-bzr117-1.2
backports.ssl-match-hostname==3.4.0.2
basemap==1.0.7

...
xhtml2pdf==0.0.6
zope.interface==3.6.1
```

More about `pip`

- Uninstall packages:

```
$ pip uninstall
```

- List installed packages:

```
$ pip list
```

- Output installed packages in requirements format:

```
$ pip freeze
```

- Show information about installed packages:

```
$ pip show Package
```

```
Metadata-Version: 1.1
Name: numpy
Version: 1.9.2
Summary: NumPy: array processing for numbers, strings, ...
...
Requires:
```

Running your code: several solutions

Edit, then run in a shell:

```
$ python mycode.py
```

or

```
$ mycode.py
```

if *shebang*

```
#!/usr/bin/python
```

is present at the 1st line

Running your code: several solutions

Interactive python (ipython)
Auto-completion, exploring objects, . . .

```
In [2]: string = 'Hello all'

In [3]: string.
string.capitalize   string.encode       string.format       ...
string.rstrip       string.strip        string.upper
...
string.startswith   string.translate
```

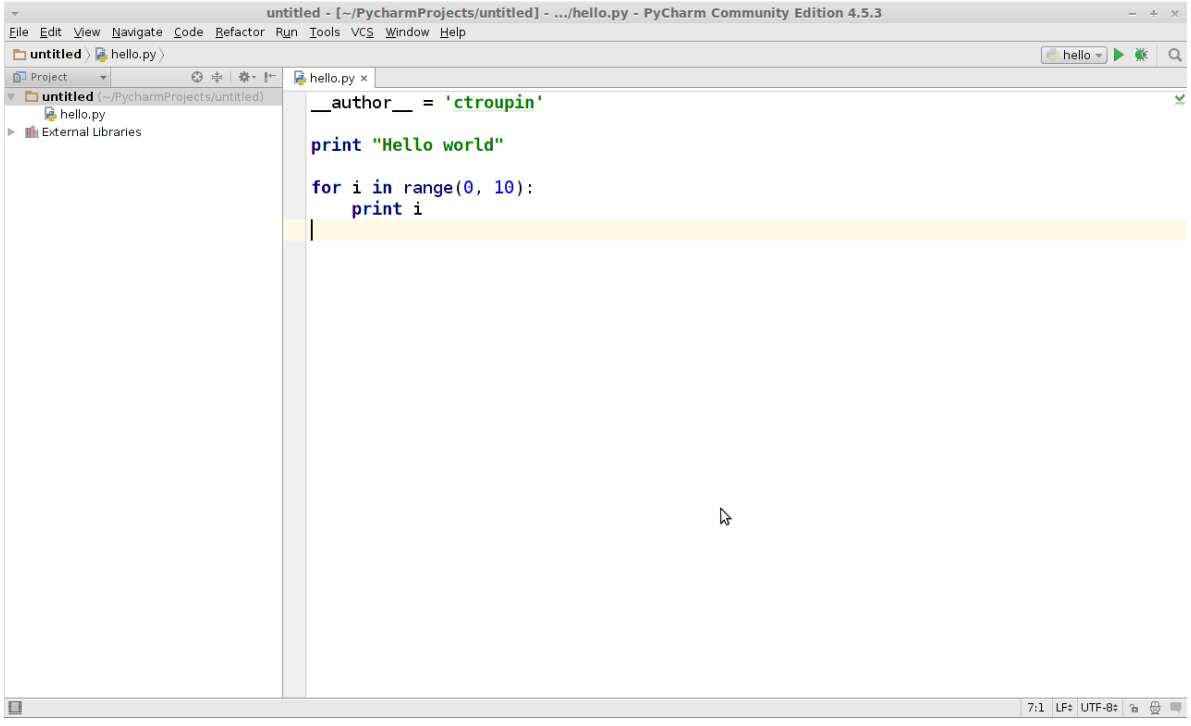
+ *magic* functions:

- `%run`: Run the named file inside IPython as a program
- `%timeit`: Time execution of a Python statement or expression
- `%who`: Print all interactive variables, with some minimal formatting

More: Built-in magic commands

Running your code: several solutions

Integrated Development Environment (IDE)
(editor + build automation tools + debugger)

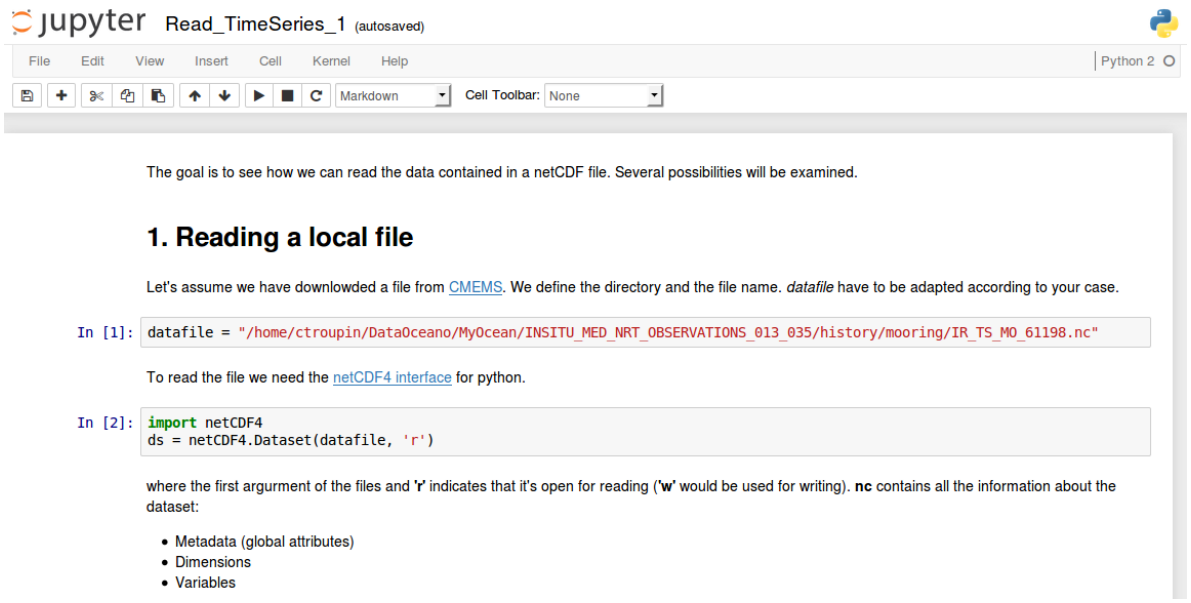


Examples: Atom, Eclipse, PyCharm, Idle, . . .
Complete list: https://wiki.python.org/moin/PythonEditors

Running your code: several solutions

ipython notebook

(interactive computational environment)



Rich text + command executions + figures + . . .

"*Data story telling*"

(see Programming in Python 2)

What do we work with?

Exercise 1: changecase.py

 :

Write a program that takes 2 arguments: the name and the surname, both written with a mix of upper and lowercase, and return the name with the first letter in uppercase and the surname with all the letters in uppercase.

Examples:

changecase allan rickman returns *Allan RICKMAN*

changecase aLlAn ricKmaN returns *Allan RICKMAN*

Tips: use the function sys.argv to be able to run the code as

```
$ changecase name surname
```