ENEL 645 – Python Bootcamp

Python 3 and efficient array processing with Numerical Python (NumPy)

Roberto Souza
Assistant Professor
Electrical and Computer Engineering
Schulich School of Engineering

January 2023

UNIVERSITY OF CALGARY

Outline

- Learning Objectives
- Python
    - What it is? And why use it?
    - Python variables, data types and syntax
    - Hands-on
- Numerical Python (NumPy)
    - NumPy array slicing
    - Hands-on
- Review of the Concepts
- Homework

Learning Objectives

- Introduction to Python:
  - Data types
  - Syntax

- Numerical programming in Python:
  - NumPy library
  - Efficient NumPy array processing

This is just an introduction! Students are advised to go through the first two tutorials in the class GitHub repository.

Tutorials:

- Tutorial 01: Introduction to Python
- Tutorial 02: Introduction to NumPy
- Tutorial 2.1: Avoid loops!

What is Python?

- Programming language created in 1991
  - Interpreted:
    -
    - Potentially slower performance. Avoid **explicit loops** at all cost!
  - High-level
    - Strong abstraction from the details of the computer
  - General-purpose
    - Applied on a range of domains: backend web development, scientific computing, data analysis, and **machine learning (ML)**

Why use Python?

Lasagne

Caffe

TensorFlow

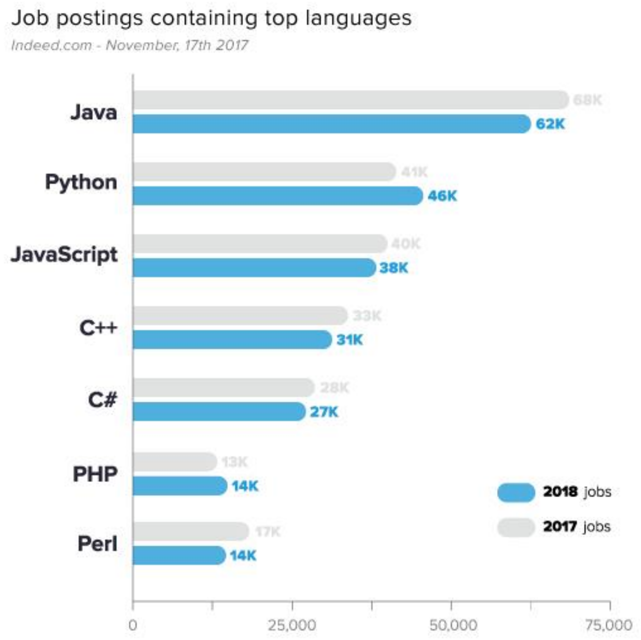
Caffe2

theano

K Keras

PYTORCH

- It is the ML language
- Preferred by data scientists and engineers - professions in high-demand



Source: https://www.codingdojo.com/

Why use Python?

- High productivity
- Programming paradigms
    - Object-oriented and structured programming
- Robust standard and non-standard libraries
    - NumPy, Pandas, OpenCV, Scikit-learn, Scikit-image, etc.
- Maintainable code
    - Readable and clean
- Installing libraries is easy
    - pip install *<library name>*
- Compatible with major platforms
- Large community
    - Fastest growth in past years compared to other languages

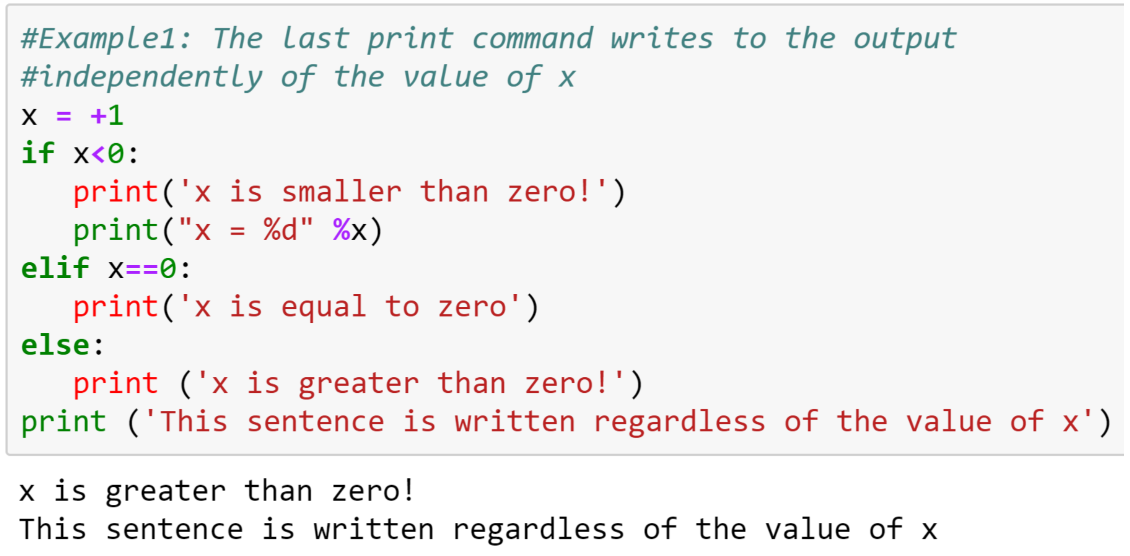
Python Variables and Data Types

- No need to pre-declare variables and their types

Variable type	Description	Syntax example
int	Integer variable	a = 103458
float	Floating point variable	pi = 3.14159265
bool	Boolean variable - True or False	a = False
complex	Complex number variable	c = 2+3j
str	UNICODE string	a = "Example"
list	Heterogeneous list (any type of elements)	my_list = [4,'me',1]
tuple	Heterogeneous tuple (values can't change)	my_tuple = (1,'I',2)
dict	Associative set of values	dic = {'me':1,'you':2}
set	Unordered collection of unique items	a = {5,2,3,1,4}

Syntax: Flow Control Statements

- Blocks, such as "if", "for" are delimited by **code indentation** and not delimiters like "{}" "BEGIN...END“
- Main flow control commands:
  - if/else
  - For
- Avoid **explicit loops** at all cost in Python!



```
#Example1: The last print command writes to the output
#independently of the value of x
x = +1
if x<0:
   print('x is smaller than zero!')
   print("x = %d" %x)
elif x==0:
   print('x is equal to zero')
else:
   print ('x is greater than zero!')
print ('This sentence is written regardless of the value of x')
```

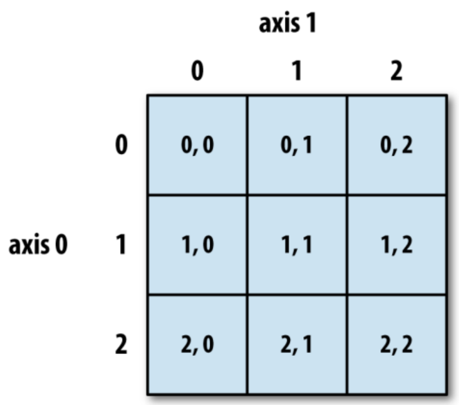
```
x is greater than zero!
This sentence is written regardless of the value of x
```

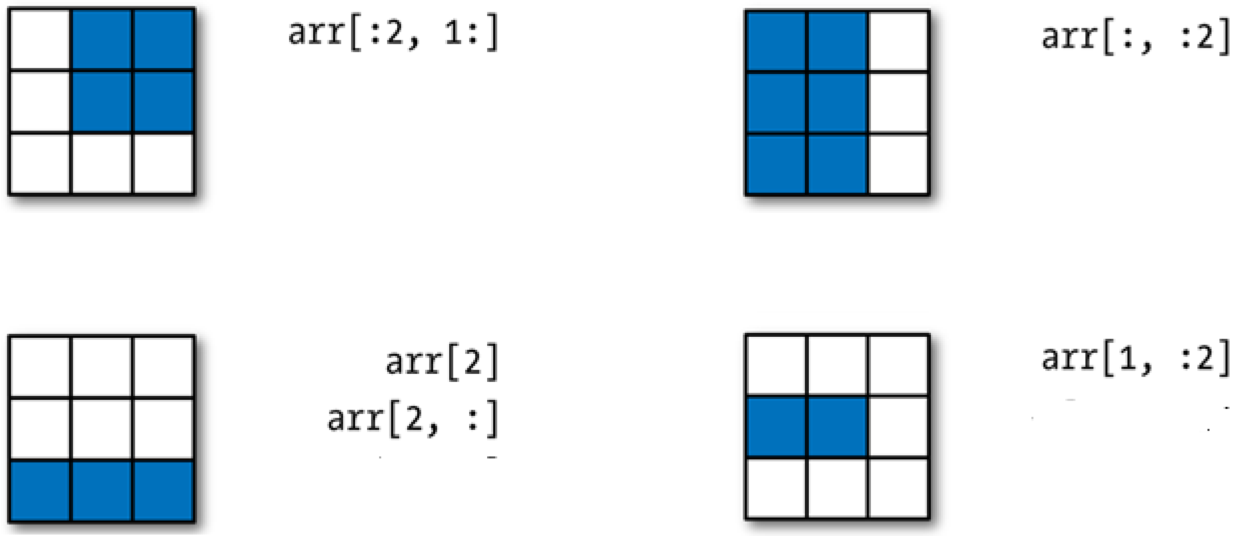
Numerical Python (NumPy)

- *NumPy* is a Python library
- It has functions for efficient multi-dimensional array processing
- It is the basis for scientific computing with Python
  - ML applications are all built upon NumPy arrays (*ndarray*)
- NumPy functions are compiled (i.e., faster)
  - Let the NumPy functions do the looping for you!

NumPy Slicing

- NumPy slicing
  - Compiled NumPy functions implicitly do the looping for you
  - Notation: [begin:end:step]
  - End element is not included





Summary

- Python is a high productivity language with a rich set of libraries, especially related to ML
- Flow control statements are defined by indentation
- NumPy array processing style can lead to efficient code
- Avoid **explicit loops** at all cost!

UNIVERSITY OF CALGARY

Thank you!

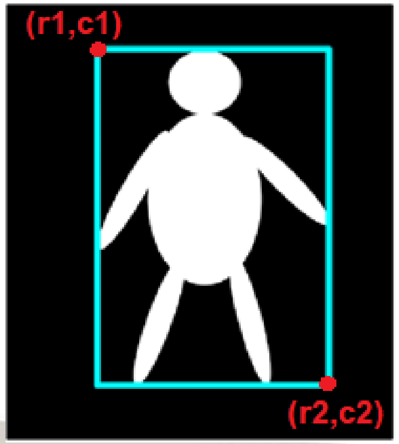
Questions?

Homework

- **Assignment:**

Write a function that receives as input a 2D Boolean NumPy array and outputs the coordinates of the minimal bounding-box that encloses all non-zero elements in the input array. Try to make your code clear and with fast execution.

Tip: search for the functions ***nonzero*** and ***where*** in the NumPy documentation.



- **Rules:**

1. All solutions will be visible to all students from initial development to the final code;
2. If you got your implementation idea from someone else, give credit to that person;
3. If you have a suggestion to improve a friend's code, leave a suggestion.