Lesson 4 - One class to rule them all

Who am I?

Who am I?

- The youngest of the group

Who am I?

- The youngest of the group
- The **tallest** of blond teachers

Who am I?

- The youngest of the group
- The tallest of blond teachers

(In summaries… you can always let data say what you want!!)

Who am I?

- The youngest of the group
- The tallest of blond teachers

(In summaries… you can always let data say what you want!!)

… but… the most important is....

Who am I?

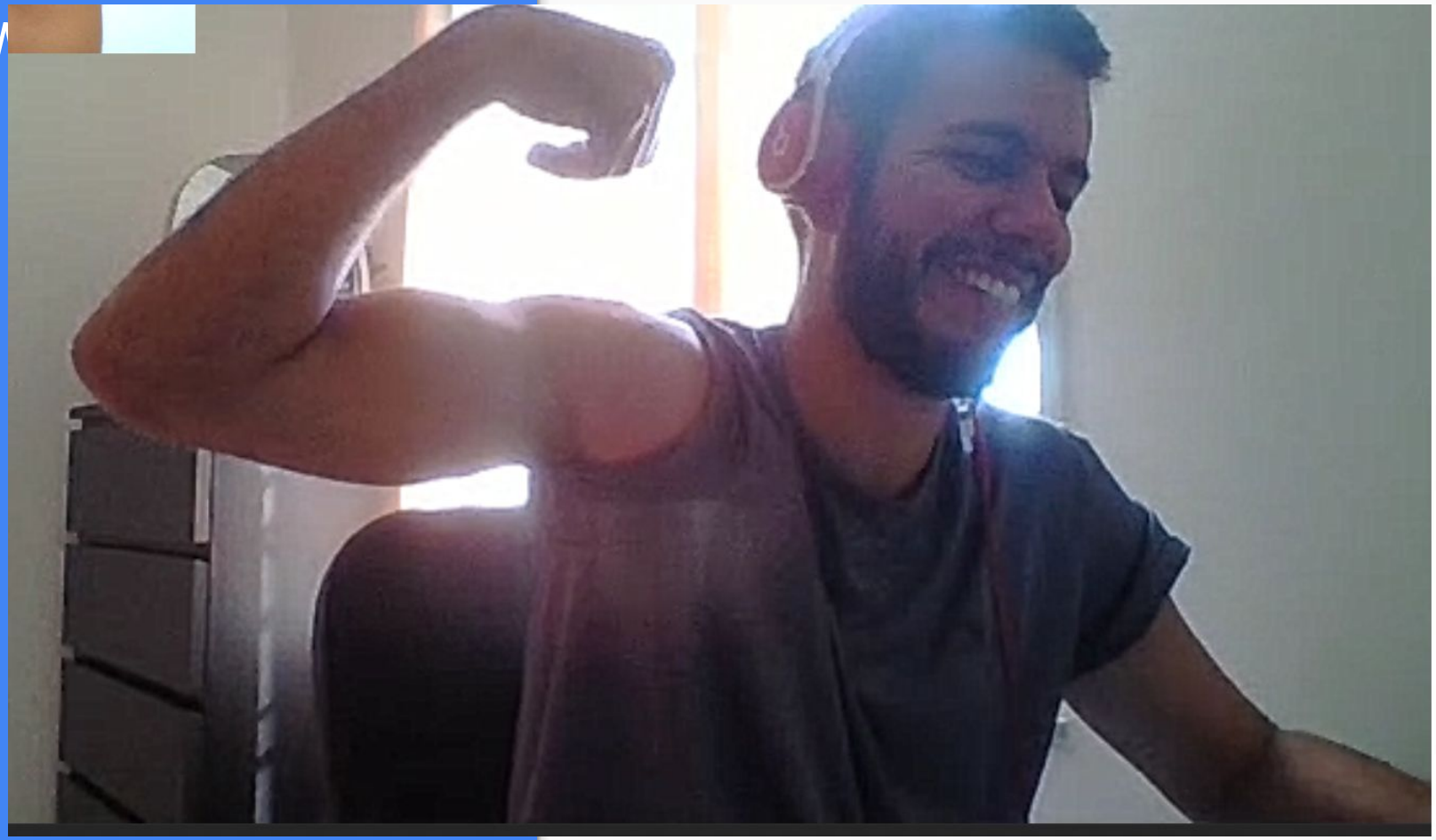
- The youngest of the group
- The tallest of blond teachers

(In summaries… you can always let data say what you want!!)

… but… the most important is....

- I won a Basketball match against Francesco

W



*n*ant!!)

Who am I?

- Bsc in Management Engineering
- MSc in Data Science
- Second year of master at UPC
- Internship at Eurecat

- PhD… HOPE SO!

For any question feel free to ping me:

michele.gentili93@gmail.com

eurecat
Technology Centre of Catalonia

UPC
UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

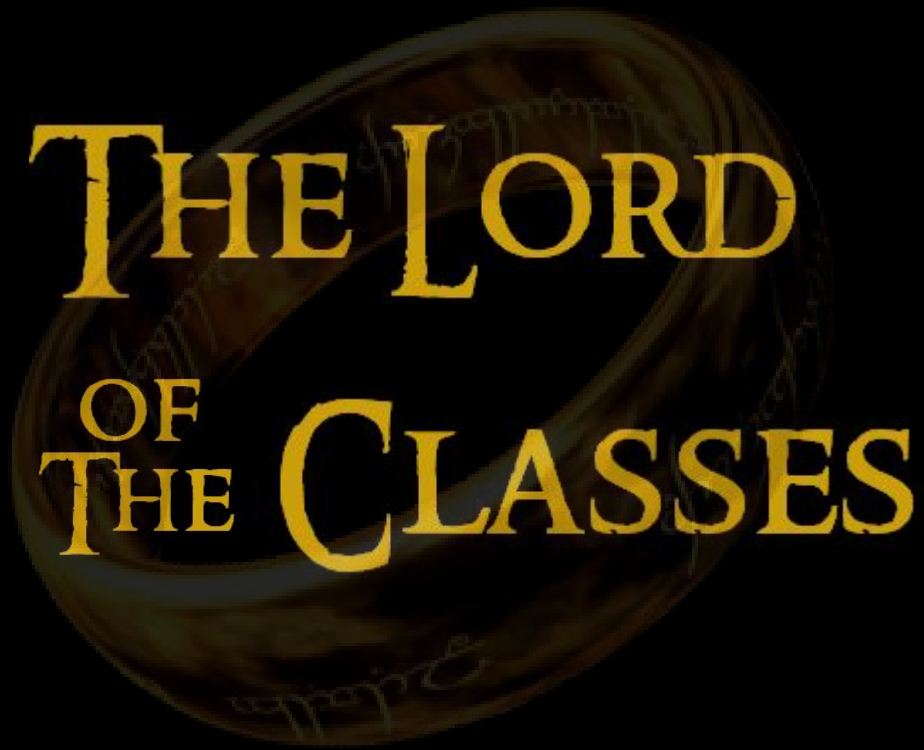
SAPIENZA
UNIVERSITÀ DI ROMA

PR 101
python crash course

Today’s goals

Today’s goals

Save the middle-earth


THE LORD
OF
THE CLASSES

The Problem

We want to help J.R.R. Tolkien writing *The Lord of the ring* in Python.

The Problem

We want to help J.R.R. Tolkien writing *The Lord of the ring* in Python.

We have to represent the middle earth, its characters and the war with what we have learnt so far:

- Dictionaries
- List
- Tuple
- etc...

The Problem

We want to help J.R.R. Tolkien writing *The Lord of the ring* in Python.

We have to represent the middle earth, its characters and the war with what we have learnt so far:

- Dictionaries
- List
- Tuple
- etc...

```
"""
Created on 05-09-1980
@author: J.R.R. Talkien
"""
gandalf = { 'Status': 10,
            'Heal': lambda x : x['Status']+1 }

#Gandalf heal himself
gandalf['Status'] = gandalf['Heal'](gandalf)
```

Exercise 1

Design how to represent:

- Characters
  - properties (list,dict etc…), actions (functions ..)
- The Middle earth
  - position of characters (matrix)

A possible solutions

Middle earth : a Matrix, where if two characters share the same coordinates, the have to fight

A possible solutions

Middle earth : a Matrix, where if two characters share the same coordinates, the have to fight

```
 8 # long and lat size of the middle earth
 9 land_size = 5
10 # middle earth is a matrix of land size^2 squares
11 middle_earth = [redacted]
12
```

```
In [128]: middle_earth
Out[128]:
[[0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0]]
```

A possible solutions

Middle earth : a Matrix, where if two characters share the same coordinates, the have to fight

```
 8 # long and lat size of the middle earth
 9 land_size = 5
10 # middle earth is a matrix of land_size^2 squares
11 middle_earth = [ [0]*land_size for x in range(land_size)]
12
```

```
In [128]: middle_earth
Out[128]:
[[0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0]]
```

A possible solutions

Gandalf : a dictionary with some specifics.

- Name
- Position in the matrix
- Status (5 life points)
- a dictionary with Attack and Defense values, (6,6)

Sauron : a dictionary with some specifics.

- Name
- Position in the matrix
- Status (5 life points)
- a dictionary with Attack and Defense values, (6,6)

Character collector : a dictionary containing the character variables as values and the name as keys.

A possible solutions

```
#create caracters as dict
gandalf = { 'Name': 'Gandalf',
            'Position': [0,0], #initial position in middle_earth
             'Status': 5, #Life points

             'Characteristics': { 'Attack': 6,
                                  'Defense': 6}
             }

sauron = { 'Name':'Sauron',
       'Position': [land_size-1,land_size-1],
             'Status': 5,

             'Characteristics': { 'Attack': 7,
                                  'Defense': 5,}
             }

# collect all the characters
characters_collection = [redacted]
```

A possible solutions

```
#create caracters as dict
gandalf = { 'Name': 'Gandalf',
            'Position': [0,0], #initial position in middle_earth
             'Status': 5, #Life points

             'Characteristics': { 'Attack': 6,
                                  'Defense': 6}
             }

sauron = { 'Name':'Sauron',
       'Position': [land_size-1,land_size-1],
             'Status': 5,

             'Characteristics': { 'Attack': 7,
                                  'Defense': 5,}
             }

# collect all the characters
characters_collection = {'Gandalf' : gandalf,
                         'Sauron': sauron}
```

A possible solutions

def Fight(character,x,y**)**: Given a character, fight against the character in position (x,y)

```
62 def fight(character,x,y):
63     '''
64     Fight among characters that are in the same position.
65     Damage = Attack score - Defense score
66     The character who moved last attack first
67 
68     @input: character: that has recently moved
69             x, y : position in the land
70     @output:
71         0: no deads
72         -1: if someone is dead
73     '''
74 
```

A possible solutions

```
75      #c1 is the character that moved last
76      c1 = character
77      #c2 is the character that already was in that position
78      c2 = characters_collection[middle_earth[x][y]]
79
80      #c1 attacks c2
81      c2['Status'] -= c1['Characteristics']['Attack'] - c2['Characteristics']['Defense']
82      # if c2 dies stop
83      if c2['Status']<1:
84          print c2['Name'], 'is dead'
85          return -1
86
87      #c2 attacks c1
88      c1['Status'] -= c2['Characteristics']['Attack'] - c1['Characteristics']['Defense']
89      # if c1 dies stop
90      if c2['Status']<1:
91          print c2['Name'], 'is dead'
92          return -1
93
94      print c1['Name'],' ', c1['Status']
95      print c2['Name'],' ', c2['Status']
96      return 0
```

A possible solutions

def Move(): a function that change the position of the character and check whether there's already someone else in the spot and in case engage the fight!

A possible solutions

def Move(): a function that change the position of the character and check whether there's already someone else in the spot and in case engage the fight!

```
35 def move(character):
36     '''
37         Move the character of his speed, at each step check wheater he hits someone.
38         In case of hits, the one who moved last, attack first
39 
40         @input: Character to move
41         @output: -1 someone is dead
42                  0 none is dead
43     '''
44     x,y = [redacted] #current coordinates
45     middle_earth[x][y]= 0 # free the cell
46 
47     # try to move of 1 cell, if it hits the boundary it stays
48     random_movement = np.random.randint(-1,2,2)
49     x = max([0,min([[redacted])
50     y = max([0,min([[redacted])
51 
52     #if the cell is already taken by a character attack
53     if middle_earth[x][y]!=0:
54         # if someone die, stop
55         if fight(character,x,y) ==-1:
56             return -1
57     middle_earth[x][y]= character['Name']
58     character['Position'] = [x,y]
59     return 0
60 
```

A possible solutions

The story: Let the character move until someone dies.

```
91 break_=False
92 iteration = 0
93 while True:
94     iteration+=1
95     #infinite loop, at each step move one by one all characters
96     for character in characters_collection.itervalues():
97         if move(character)==-1:
98             break_=True
99             break
100    if break_:
101        break
102
103    for line in middle_earth:
104        print line
105    print
106 print 'Total number of iteration: ',iteration
```

A possible solutions

The story: Let the character move until someone dies.

A possible solutions

Gandalf : a dictionary with some specifics.

- Name
- Position in the matrix
- Status (5 life points)
- a dictionary with Attack and Defense values, (6,6)

Sauron : a dictionary with some specifics.

- Name
- Position in the matrix
- Status (5 life points)
- a dictionary with Attack and Defense values, (6,6)

Something wrong ?

Can we actually save the middle earth coding in this way?

Something wrong ?

NO

Something wrong ?

Something wrong ?

- All variables share the same properties

Something wrong ?

- All variables share the same properties

Health

Something wrong ?

- All variables share the same properties

Health

Power

Something wrong ?

Health

- All variables share the same properties

Power

Actions

Something wrong ?

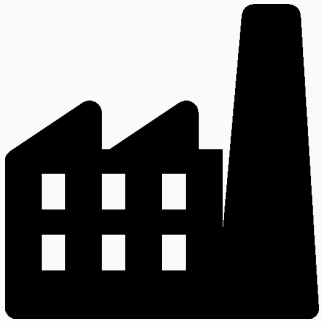
- All variables share the same properties
- We want them to interact

A possible solutions

A possible solutions

Let’s build a factory:

- it creates objects
- an object is an instance of a fixed structure

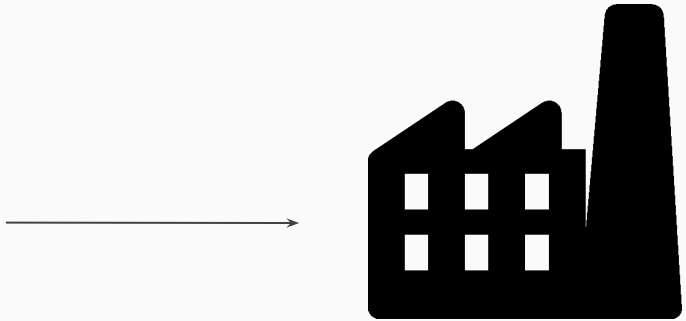


A possible solutions

Let’s build a factory:

- it creates objects
- an object is a realization on the same model

inputs

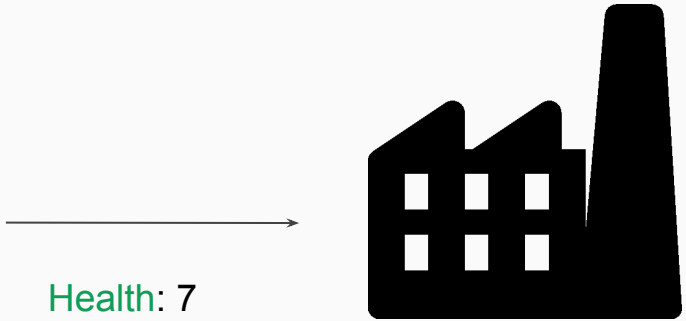


A possible solutions

Let’s build a factory:

- it creates objects
- an object is a realization on the same model

inputs



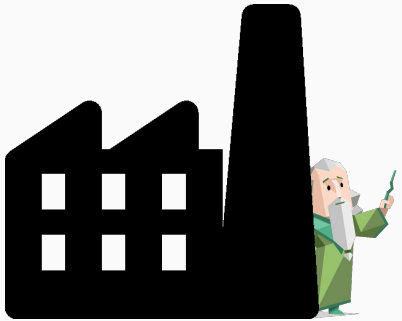
Health: 7
Power: 6
Defense: 7
Race: Wizard
Side: 1

A possible solutions

Let’s build a factory:

- it creates objects
- an object is a realization on the same model

inputs



Health: 7
Power: 6
Defense: 7
Race: Wizard
Side: 1

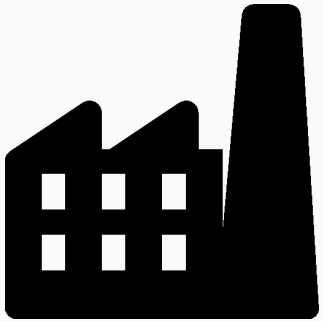
A possible solutions

Let’s build a factory:

- it creates objects
- an object is a realization on the same model

inputs

Health: 8
Power: 5
Defense: 5
Race: Human
Side: 1

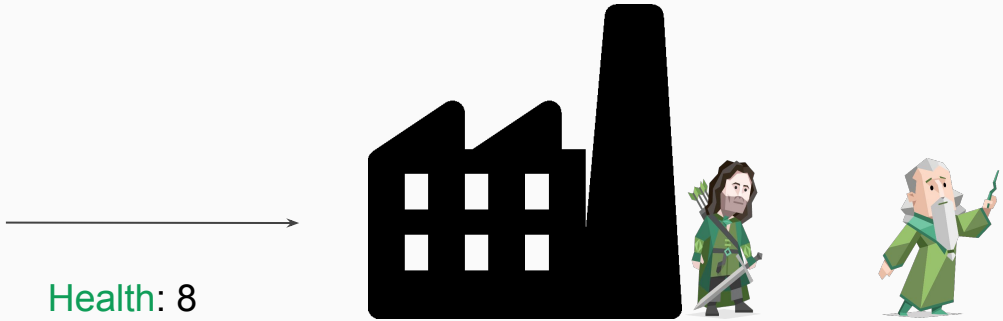


A possible solutions

Let’s build a factory:

- it creates objects
- an object is a realization on the same model

inputs



Health: 8
Power: 5
Defense: 5
Race: Human
Side: 1

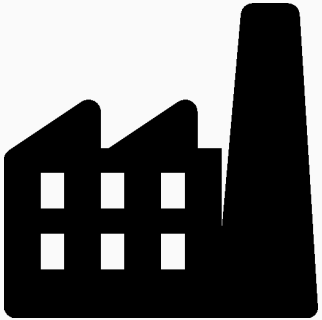
A possible solutions

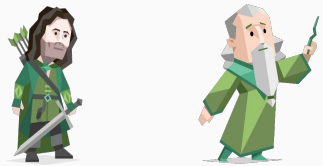
Let’s build a factory:

- it creates objects
- an object is a realization on the same model

inputs

Health: 10
Power: 10
Defense: 10
Race: Angel
Side: -1





A possible solutions

Let’s build a factory:

- it creates objects
- an object is a realization on the same model

inputs

Health: 10
Power: 10
Defense: 10
Race: Angel
Side: -1

A possible solutions

Let’s code

Afternoon Exercise

What : Create two classes, one representing characters and the other one representing the Battle

```
class Character:

    '''
        This class represents the character of LOTR. A character is defined by his health, his power, his defense,his race
        and his side.

        Attributes:
            @Name: name of the character
            @Health: is the life of a character, when it goes below 0, it's dead
            @Power: it's the strenght of the character, will be used to attack other characters
            @Defense: it's the shield of the character, will be used when receive attacks from other characters
            @Side: If it fights for Sauron is -1, against Sauron 1 and neutral 0
            @Position: Dictionary with x,y position
    '''
```

In [315]:
```python
import numpy as np

class Battle:
    '''
        This class represent a battle.
        The world it's a rectangle of shape = 2*bounds[0],2*bounds[1]

    '''

    def __init__(self, list_characters, bound_x,bound_y):
        self.list_characters = list_characters #list of characters objects
        self.bounds = {'x':bound_x,'y': bound_y} #longitude and latitude of the battlefield


    def move(self,character,x,y):
        '''
            This function will move the character. the new position is the algebric sum of the current plus the new one.
            When character hit the border, it bounces and moves backward of the remaining steps. Es border = 10, current =8,
            move 5, it will go to 7

            param:
                @character: the object who has to move
                @x: natural number, will move the character in x axe
                @y: natural number, will move the character in y axe

            return:
                1: moved
                -1: ERROR
        '''
```

```
def random_move(self,character):
    '''
        Let this character make a random move.
    '''

def print_matrix(self):
    '''
        This function just print a matrix full of zeros, containing in their position the characters, they will be prnted as
        their position. So that if there are two characters in the same position they will show up together.
        So if ch1 and ch2 are in [1,1] will be:
        [[0,0,0],
         [0,[ch1,ch2],0],
         [0,0,0]]

    '''
```

```
def copresence(self,character):
    '''
        This function check whether this character is in the same cell of someone

        param:
            @character: the character who's moving

        output:
            @list of character that are in that cell
    '''

def fight(self,char1,char2):
    '''
        This method do the fight among two characters from different side. Who start is random

        param:
            @char1,char2 the two characters involved in the fight
    '''

def still_fighting(self):
    '''
        Check if there are still people from opposite sides.

        output:
            True/False
    '''
```

Main:

```
aragorn = Character('Aragorn',health=8,power=6,defense=6,side=1)
gandalf = Character('Gandalf',health=10,power=6,defense=7,side=1)
frodo = Character('Frodo',health=3,power=6,defense=3,side=1)
gimly = Character('Gimly',health=7,power=6,defense=8,side=1)
sauron = Character('Sauron',health=9,power=10,defense=5,side=-1)

list_char = [aragorn,gandalf,sauron,frodo,gimly]
middle_earth = Battle(list_char,5,5)
```

Than write a loop that until Sauron is alive or all the others are dead, make them randomly move a and fighting

Bonus point (A free beer on Friday**)** : add 5 goblins and 2 elf warriors