INFO-H-600 - Python
TP 5 - Object Oriented Introduction

For each exercice, a file containing the tests is provided. These tests are contained under the line `if __name__ == "__main__"`.

For instance:

```
# write your classes and functions here

if __name__=="__main__":
    # tests that have to run correctly
```

For these exercices, all attributes must be *hided* (declared starting with __). No object can modify the attribute of another object directly!

1 People

1.1 Basic class

Create a class representing people. Each person has the following arguments:

- an unique name (no two persons have the same name) represented by a string
- an age represented by an integer
- the cities where they live (we suppose each person can live in different cities)

Create this class, give it a constructor, and a method called `greetings` which make the object print a short presentation line.

1.2 Friends

When one person encounters another person, both become friends of each other.

- Add to the class of previous exercise a attribute containing its friends
- Create the necessary methods to update the friends of two persons meeting for the first time.
- Create the method `print_friends(self)` that asks to all the friends of the given person so execute the greetings method.

1.3 Rock Paper scissor

A person can play the *Rock Paper scissor* game against another person.

To represent the game, we will consider that the players play until one of them wins five rounds (and therefore wins the game). During each round, each player will select a natural number between 0 and 2 at random. Zero representing the paper, one the rock and two the cissor.

You should create the following methods and functions:

- `play_rpc(player, opponent)`: perfoms on game between the player and its opponent. Returns true if the player wins, false if the opponent win
- `play_rpc_round(my_choice, opponent_choice)`: returns `True` if the first player wins the round, false if its the opponent.

- `get_random_choice(self):` define what the person is going to play for this turn

If you want to obtain the same results as in the correction, you should only use the function `random.randint` once in the function `get_random_choice(self):`.

1.4 Cities

Create a class `City` representing the cities. Each city

- has an unique name.
- contains the persons which live their.

The class (and not the objects!) must contain a dictionary where each city object is associated with its name (the name is the key).

You should be able to acces it as follows:

```
City.get_city("Bruxelles")
```

If the city was not created yet, it should be created and added to the dictionary in the `get_city` method.

Modify the class People so that each people do not contain the names of the cities its living in, but the city object itself.

1.5 Rock Paper Scissor Tournament

A city can organise a Rock Paper Scissor tournament. In such a tournament each person living in the city will play against each other player *once*. For each victory, the player wins three points and the looser wins one. At the end of the tournament the player or players having the most points win the tournament. Each person must keep as attribute the total amount of tournaments it won.

2 Additional exercise

Write a function *remplace_letters(letters, old, new)* that recieves the name of 3 files as argument. The function must write in a new file (*new*) the content of the file *old*, where each letter has been replaced by "x" time another letter (respecting the case). The rules for remplacing are indicated in the *letters* file.

Example file `letters.txt`

```
A  E  3
E  O  2
I  U  3
P  R  4
U  I  1
Y  A  3
```

Indicates that you must therefore replace all the "a" and "A" in the *old* file by "eee" or "EEE" in the file *new*.

The call

```
>>> remplace_letters("letters.txt", "text.txt", "new_text.txt")
```

With the file `text.txt` :

```
Python is an interpreted, high-level and general-purpose
programming language.
Created by Guido van Rossum and first released in 1991,
Python’s design philosophy emphasizes code readability
with its notable use of significant whitespace.
```

Would produce the file `new_text.txt` with the following content:

```
RRRRaaathon uuus eeen uuuntoorrrrrrootood, huuugh-loovool eeend
goonooreeel-rrrrirrrrrosoo
rrrrrogreeemmuuung leeengieeegoo.
Crooeeetood baaa Giuuudo veeen Rossim eeend fuuurst roolooeeesood uuun 1991,
RRRRaaathon’s doosuuugn rrrrhuuulosorrrrhaaa oomrrrrheeesuuuzoos
codoo rooeeedeeebuuuluuutaaa
wuuuth uuuts noteeebloo isoo of suuugnuuufuuuceeent whuuutoosrrrreeecoo.
```