INTRODUCTION TO PROGRAMMING

19/20 S2

Midterm

Rules

1. You can use internet or any other appointments you might have.
2. You MUST not copy from your colleagues.
3. You should deliver all your files in a zip file with your number.
4. Comment your code as much as possible
5. **Duration: 3h + 15 minutes (tolerance)**

Read everything with attention!

GROUP 1

Exercise 1 (1,5 points)

In this exercise you need to implement a function to convert Celsius to Kelvin or Fahrenheit. The function receives temperature and units for inputs. Temperature will be a char with values k for Kelvin and f for Fahrenheit. The function must handle the situation where the user inserts lower or upper case. After converting the temperature, the function must return the temperature plus the temperature units. The temperature units must be upper case always.
Test the function asking the user for the inputs and then call the function with the correct inputs.

e.g.

Insert the temperature you want to convert: 4
What are the units? F
the temperature is 39.2F

Insert the temperature you want to convert: 20
What are the units? k
the temperature is 293.15K

FORMULAS:
*C to F: (temperature in Celsius * 9/5)+32*
C to K: temperature in Celsius +273.15

Exercise 2 (2 points)

Implement a function that validates Portuguese postal codes.
A Portuguese postal code has the following structure:

- First four elements are digits
- Fifth element is a hyphen
- Last three elements are digits

The function name is postal_code_validator and it receives as input, a string with postal code.
The function validates in this order:

1. **Postal code size:** must be 8 characters
2. **If the four first elements are all digits**
3. **If the last 3 elements are all digits**
4. **If the separator is in the postal code string position 5 and is a hyphen ("-")**

Examples:
postal code: 2222-456
Output:
Postal code 2222-456 is ok

postal code: 22e2+46a
Output:
Postal Code must have a hyphen separating the number in the fifth position
The first four elements in postal code must be digits.
The last 3 elements in postal code must be digits.

postal code: 2242-46
Output:
Postal Code length is wrong.

Exercise 3 (2 points)

Implement a function to draw squares.
The function receives side_size as input. The minimum side size is 2.
To test the function, ask the user input for the square side.
Note: each asterisk in a square line is separated by an empty space.
e.g.

```
Insert the square side: 1
Minimal square size is 2.

Insert the square side: 2
* *
* *

Insert the square side: 4
* * * *
* * * *
* * * *
* * * *
```

```
Insert the square side: 6
* * * * * *
* * * * * *
* * * * * *
* * * * * *
* * * * * *
* * * * * *
```

GROUP 2

Dictionaries and lists

Question 01 (1 point)

Create a function to print an individual character from the character list. *(see dictionary.py file)*
The function should print "Villain" or "Avenger" in case the dictionary key " avenger" is False or True.
The command line output should be *Villain* or *Avenger*, followed by the character name.
The next lines should print the power and power values for each character and have a tab before so It can help the user understanding. Below an example for this is:

```
Avenger Iron Man
	strength: 80
	intelligence: 100
	magic: 0
	humor: 90
```

Question 02 (1 point)

Create a function that will filter the characters to print.
the function receives as input on of three possible values, "all", "avengers" or "villains".
If the input is "all", all the characters are printed. But if its "avengers", only the avengers will be printed. The same corresponding thing happens for villains.
an example for this is:

```
Characters list
Avenger Iron Man
	strength : 80
	Intelligence : 100
	magic : 0
	humor : 90
Avenger Captain America
	strength : 80
	intelligence : 90
	magic : 0
	humor : 50
Avenger Hulk
	strength : 95
	intelligence : 10
	magic : 10
	humor : 70
```

Question 03 (2 points)

Create a function to add a new character to the list. The function should receive has inputs the character name and a Boolean variable indicating if it is an avenger or not. The function must also validate if the new character being inserted already exists in the character list.
If it exists, a message saying "Character [NAME] already exists.", where [NAME] is the character name being inserted.
Once the character is valid, we will set all its power values to be a random number from 1 to 100. Once the character is added, the message "Character [NAME] created with success." where [NAME] is the character name.

If you don't know the Avengers (shame on you) here are some characters to add:

```
add_character("Ant-man", True)
add_character("Ant-man", True)
add_character("Hawkeye", True)
add_character("Red Skull", False)
add_character("Ultron", False)
```

In the example below, we created Ant-man twice and got the error message, Hawkeye, red skull and Ultron. All the characters with random values for its powers.

```
Character ant-man created with success.
Character ant-man already exists.
Character Hawkeye created with success.
Character Red Skull created with success.
Character Ultron created with success.

Characters list
Villain Thanos
        strength : 100
        intelligence : 80
        magic : 100
        humor : 10
Avenger Iron Man
        strength : 80
        intelligence : 100
        magic : 0
        humor : 90
Avenger Captain America
        strength : 80
        intelligence : 90
        magic : 0
        humor : 50
Avenger Hulk
        strength : 95
        intelligence : 10
        magic : 10
        humor : 70
Villain Loki
        strength : 70
        intelligence : 90
        magic : 90
        humor : 80
Avenger Ant-man
        strength : 65
        intelligence : 73
        magic : 76
        humor : 58
Avenger Hawkeye
        strength : 19
        intelligence : 44
        magic : 31
        humor : 77
Villain Red skull
        strength : 24
        intelligence : 27
        magic : 53
        humor : 49
Villain Ultron
        strength : 73
        intelligence : 57
        magic : 29
        humor : 43
```

Question 04 (1 point)

Create a function called *remove_not_funny*, that receives as input an integer that represents the humor level. What this function does is remove from the list all the characters with humor less or equal to the humor level set in the input.

Question 05 (2 points)

Create a function called battle. this function receives no inputs and what it does is select two random characters from the character list. The function must assure that two equal characters do not battle between them. After the players are selected, the function should announce the battle. The next step is to select randomly the power that will be used to battle between the characters. Once the power is selected, the function should compare the selected power and the character with the highest value wins. Pay attention to all possible outcomes.
In case of a tie, print the message "Incredible....we have a tie!".
Here is a possible scenario for the battle function:

```
A new battle is about to begin between...

player 1: Hawkeye
       Vs
player 2: Red Skull

The selected power is...
        magic

Hawkeye magic is 35
Red Skull magic is 16

THE WINNER IS...
Hawkeye!!!
```

GROUP 3

Objects

In this exercise we will create an **Animal class** with animal common characteristics, and then two other classes that will represent specific animals.
All animals must have a name, age and gender. All these attributes cannot be accessed from outside the class and should be private, unless you use specific functions for this.
the animal class has a function to return the number of meals each animal should have per day, and by default all animals should eat 3 times per day.
There are also two other animal classes, one to calculate the correspondent animal human years and another to print the animal information. These two classes are mandatory for all animals, but they are defined in different ways, depending on the animal.

A specific type of animal is the **Bird class**. The Bird class has all the Animal class attributes and adds two more, talk (Boolean to indicate If the bird talks or not) and a list to register the bird favorite words.
When adding new words to the favorite words list you should verify if the bird talks, and when you print those same favorite words, you should confirm of the bird has favorite words.
To define the function that calculates the bird's human years, you need to know that a bird's year corresponds to 8 human years.
Also, when printing the Bird class you should have this output:

```
[NAME] is [AGE] years old, but in human years it has [HUMAN YEARS]
```

[NAME] – The bird's name
[AGE] – The Bird's age
[HUMAN YEARS] – The Bird's human years

Another specific type of animal is the Dog class. The Dog class has the exact same attributes the Animal class has. In the Dog class, the function to get meals, returns different values than the one in the Animal class. In here, if the dog is younger than or has 3 years old, it has 5 meals per day. If the age is between 3 and smaller or equal to six, it has four meals per day, otherwise it has 3 meals per day.
To define the function that calculates the dog's human years, you need to know that a dog's year corresponds to 5 human years.
Also, when printing the Dog class you should have this output:

```
[NAME] is [AGE] years old, but in human years it has [HUMAN YEARS]
[NAME] also needs to eat [MEALS] times per day!
```

[NAME] – The Dog's name
[AGE] – The Dog's age
[HUMAN YEARS] – The Dog's human years
[MEALS] – The Dog's number of meals per day

Deliverables and testing

You should deliver each class in its own file, where the file name corresponds to the class name (ex. dog.py will have Dog class)
To test the classes, use the objects_test.py file. Put this file in the same folder where your classes are and use the above nomenclature to test your objects/classes.

To better understand the outputs of all your functions, when you run the objects_test.py file, you should see the following output:

```
piupiu's favorite words are:
	what’s up doc?
piupiu is 3 years old, but in human years it has 24

galaro does not talk

galaro has no favorite words yet
galaro is 6 years old, but in human years it has 48

laika is 6 years old, but in human years it has 30
laika also needs to eat 4 times per day!

bobi is 2 years old, but in human years it has 10
bobi also needs to eat 5 times per day!
```

Good luck!