

OOP

September 20, 2019

1 Exercise 0

Implement a "Calculator" that is able to perform the 4 basic operation :add, subtract, multiply, divide The result should be something like the following

```
[1]: from Exercise_0 import *
if __name__=="__main__":
    c=Calculator()
    c.add(2,3)
    c.sub(2,3)
    c.mul(2,3)
    c.div(2,3)
```

```
2 + 3 = 5
2 - 3 = -1
2 * 3 = 6
2 / 3 = 0.667
```

2 Exercise 1

Implement a "Point" class that is able to represent 2-D points.
The class must contains the methods to obtain the results shown below

```
[2]: from Exercise_1 import *
if __name__=="__main__":
    a=Point(7,1)
    b=Point(1,1)
    print(a.distance(b))

    a.move(2,2)
    print(a)
```

```
6.0
(9,3)
```

Tips and tricks:

- The formula for the distance between two points is

$$d(P_1, P_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

- To use the square root function you need to import math and use math.sqrt like below

```
[19]: import math
print(math.sqrt(4))
```

2.0

3 Exercise 2

Implement a "Line" class that is able to represent 2-D lines. Using the class "Points" of the previous exercise and this one try to obtain the following results

```
[2]: from Exercise_2 import *
if __name__=="__main__":
    l=Line(3,2)
    print(l)

    a=Point(0,1)
    b=Point(2,2)
    l.line_from_points(a,b)

    l=Line(1,0)
    a=Point(1,5)
    print(l.distance(a))

    m=Line(-1,0)
    i=l.intersection(m)
    print(i)
```

Line: y=3x+2
2.82842712474619
(0.0,0.0)

Tips and tricks

- The formula to obtain the equation from 2 points is

$$y = (y_2 - y_1)/(x_2 - x_1)x - (y_2 - y_1)/(x_2 - x_1)x_1 + y_1$$

- The formula for the distance between a line and a point is

$$d(P, r) = \frac{|ax_P + by_P + c|}{\sqrt{a^2 + b^2}}$$

- the formula to find the intersection is

$$P\left(\frac{q_2 - q_1}{m_1 - m_2}, m_1 \frac{q_2 - q_1}{m_1 - m_2} + q_1\right)$$

4 Exercise 3

In this exercise we want to create a "Deck" class and a "Card" class to represent a poker deck of cards. The requirements are

1. The Deck should have a *deal()* method that pick a card (or multiple card from the deck) and remove them from the deck
2. The Deck should have a *shuffle()* method
3. The Card class should have a suit (Hearts, Diamonds, Clubs, Spades) and a value (A,2,3,4,5,6,7,8,9,10,J,Q,K)

Tips and tricks

- Pay attention to consider the case of trying to draw more cards than the actual number of cards in the deck.
- In the Python package *random* there is a function that may help you to shuffle the deck, look [here](#)

5 Exercise 4 (todo after the next theory lesson)

Create the classes "Contact" and "AddressBook" All the contacts are stored in a json file named "contacts.json" and are similar to

```
{
    "name": "Cassio",
    "surname": "Zen",
    "email": "cassiozen@gmail.com"
}
```

The "AddressBook" class must be able to read the content of the file and perform CRUD (Create, Read, Update, Delete). The Update is the most difficult so I suggest it to begin with the other. Below you can find an example

```
[24]: from Exercise_3 import *
      book=AddressBook()
      book.show()
```

Name:Cassio, Surname:Zen, mail:cassiozen@gmail.com

Name:Dan, Surname:Abramov, mail:gaearon@somewhere.com

Name:Pete, Surname:Hunt, mail:floydophone@somewhere.com

Name:Paul, Surname:Shannessy, mail:zpao@somewhere.com

Name:Ryan, Surname:Florence, mail:rpflorence@somewhere.com

Name:Sebastian, Surname:Markbage, mail:sebmarkbage@here.com

```
[25]: book.find_by_name('Dan')
```

I found the following results:

Name:Dan, Surname:Abramov, mail:gaearon@somewhere.com

```
[26]: book.remove_contact('Dan')
      book.show()
```

Name:Cassio, Surname:Zen, mail:cassiozen@gmail.com

Name:Pete, Surname:Hunt, mail:floydophone@somewhere.com

Name:Paul, Surname:Shannessy, mail:zpao@somewhere.com

Name:Ryan, Surname:Florence, mail:rpflorence@somewhere.com

Name:Sebastian, Surname:Markbage, mail:sebmarkbage@here.com

```
[27]: book.add_contact('Peter','Parker','notspiderman@marvel.com')
```

```
[28]: book.show()
```

Name:Cassio, Surname:Zen, mail:cassiozen@gmail.com

Name:Pete, Surname:Hunt, mail:floydophone@somewhere.com

Name:Paul, Surname:Shannessy, mail:zpao@somewhere.com

Name:Ryan, Surname:Florence, mail:rpflorence@somewhere.com

Name:Sebastian, Surname:Markbage, mail:sebmarkbage@here.com

Name:Peter, Surname:Parker, mail:notspiderman@marvel.com

Once you've done this you can now create a client to use the functions you implemented in a "user-friendly way" The result should be something like the following

Welcome to the application to manage your contacts
Press 's' tho show the list of contacts

```

Press 'n' to add a contact
Press 'f' to find a contact
Press 'd' to delete a contact
Press 'q' to quit
s
Name:Cassio, Surname:Zen, mail:cassiozen@gmail.com

Name:Dan, Surname:Abramov, mail:gaearon@somewhere.com

Name:Pete, Surname:Hunt, mail:floydophone@somewhere.com

Name:Paul, Surname:Shannessy, mail:zpao@somewhere.com

Name:Ryan, Surname:Florence, mail:rpflorence@somewhere.com

Name:Sebastian, Surname:Markbage, mail:sebmarkbage@here.com

Press 's' tho show the list of contacts
Press 'n' to add a contact
Press 'f' to find a contact
Press 'd' to delete a contact
Press 'q' to quit
g
Command not available
Press 's' tho show the list of contacts
Press 'n' to add a contact
Press 'f' to find a contact
Press 'd' to delete a contact
Press 'q' to quit
q

```

6 Exercise 5 (todo after the next theory lesson)

The file `playerNBA.json` contains the list of all the NBA player of this season with their stats and their bio. Each one looks like this:

```

{
  "pos": "G",
  "name": "Stephen Curry",
  "hgt": 75,
  "tid": 9,
  "injury": {
    "gamesRemaining": 0,
    "type": "Healthy"
  },
  "born": {
    "year": 1988,

```

```

        "loc": "Akron, OH"
    },
    "weight": 190,
    "ratings": [
        {
            "diq": 40,
            "endu": 74,
            "ins": 46,
            "pss": 74,
            "spd": 86,
            "tp": 95,
            "jmp": 88,
            "fg": 92,
            "stre": 50,
            "drb": 61,
            "dnk": 92,
            "oiq": 73,
            "reb": 49,
            "ft": 78,
            "hgt": 34
        }
    ],
    "draft": {
        "tid": 9,
        "pick": 7,
        "originalTid": 9,
        "year": 2009,
        "round": 1
    }
}

```

The first step is to analyze the data, so we need to create the function to evaluate: 1. The average **ratings** among the players 2. The average **height** and **weight** (in meters and kilograms while they're store in inches and pounds) 3. The average age

You should be able to obtain results like the one below

```

[2]: from Exercise_5 import *
import json
file_content=json.load(open('playerNBA.json'))

```

```

[2]: average_ratings(file_content)

```

```

[2]: {'diq': 40.86353211009176,
      'dnk': 60.25573394495405,
      'drb': 44.97018348623852,
      'endu': 41.286697247706385,
      'fg': 54.263761467889935,
      'ft': 51.10206422018341,

```

```
'hgt': 48.36353211009177,  
'ins': 45.02981651376154,  
'jmp': 53.35206422018351,  
'oiq': 46.380733944954095,  
'pss': 42.470183486238454,  
'reb': 48.62729357798164,  
'spd': 53.27752293577975,  
'stre': 46.24885321100917,  
'tp': 44.92316513761466}
```

```
[3]: average_height(file_content)
```

```
[3]: 2.0031959788690217
```

```
[2]: average_weight(file_content)
```

```
[2]: 97.63100959038039
```

```
[3]: average_age(file_content)
```

```
[3]: 26.212155963302752
```

```
[ ]:
```