La Plateforme

Day 2 - Class in python

Programming is class

Job 2.0

Create a "Person" class with the attributes "name", "firstname". Add a "**SePresenter()**" method that will display in the terminal the name and surname of the person. Also add a constructor taking as parameters to give initial values to the "name" and "firstname" attributes. Instantiate several people with the construction values of your choice and use the "**SePresenter()**" method to verify that everything is working correctly. Add an "**accessor**" and a "**mutator**" for each of the attributes.

Job 2.1

Create a "Book" class **with as an attribute a "title" that it receives as a parameter to the construction and a** reference **to an "**Author**" class**. Add a "**print**" method to display the title of the book in the terminal. Create an "**Author**" class inheriting from the **"Person**" class receiving a first and last name as a construction parameter. The author class must have a collection of books named "work" as an attribute as well as a method **"listerWork**" displaying in the terminal the list of books written by the author. Add to the Author class a "**writeABook"** method taking as a parameter a book title to write and generate an **instance** of the book class with this title. Add this new book to the author's work

Job 2.2

Create a "**Customer" class** inheriting from "**Person**", taking as a parameter a name and a first name.

Create a "Library**" class** with a name and a catalog as attributes: a collection of books where each book is associated with a quantity (represented by an integer). Add the following methods:

- **buyBook**: Taking as a parameter an author object, the name of a book and an integer representing a quantity. If the book does exist in the author's work, add this book to the library catalog with the corresponding quantity.
- **inventory**: A method that displays in the terminal the titles of the books present in the catalog as well as their quantity.
- **rent**: This method receives as parameters an **instance** of object **"Client"** as well as the name of a book. If the book exists and is in stock, add that book to the customer's book collection and keep the quantity of that book in the library up to date.
- **renderBooks**: A method that takes a "**Client**" as a parameter and retrieves all the books of the latter and adds them to the library catalog.

Add to the "**Client**" class a collection attribute that is a collection of books and add the **inventory** method that displays in the terminal the titles of the books in the possession of the client.

Then, instantiate authors, have them write books, create libraries, have them buy books, create customers, have them rent books and then use display features and show the result to your examiner.

Job 2.3

You will create in this Job a power game **4** on board of variable size as well as an algorithm that will be able to play measured moves.
Start by creating a "**Board**" class taking as construction parameters two integers i and j. Create an attribute, in the form of a 2-dimensional array, representing a two-dimensional game board of size i x j. This table represents:

- empty boxes with O's
- yellow tokens with J
- red tokens by R

Create a "**play**" method that takes as parameters an integer as well as a string of characters that can be "red" or "yellow". The integer is the column in which a game token is inserted and the color corresponds to the color of the player playing that token. After a move, keep your game board up to date by placing the chip as low as possible in the column where it was played.
Add a "**print**" method displaying in the terminal the status of the game board.
Implement the flow of a game by asking human players to take turns playing by choosing the column in which they want to insert their chips. The first player to line up 4 chips of his color wins the game and receives 100,000 euros.

Job 2.4

Create a AI_One class. This class will have to implement a "**think**" method taking as parameters a board as well as a chain of characters that can be "red" or "yellow". This method will have to return the column number in which to play as a player of the given color.
Implement the course of a game allowing a player to play against AI_One.

Job 2.5

Create a **class AI_Two** identical to the **AI_One** class but insert the AI code of one of your colleagues and then implement a part between these 2 IAs.

Rendering

In your github directory "**runtrack-python**", create a folder "**day02**" and in this folder, for each job, a folder "**jobXX**" **where XX is the number of the job.**
Don't forget to submit your changes as soon as a step is advanced or completed and use explicit comments.

Competencies targeted

- Mastering OOP architecture in Python

Knowledge Base

- Python.org
  Official python website, documentation and download.
- Tutoriel Class python & Class python
  The basics of classes in python
- Reference passage
  Go by reference in Python