La Plateforme

Jour 2 - Class en python

La programmation c'est class

Job 0

Créez une classe "Personne" avec les attributs "nom", "prenom". Ajoutez une méthode "**SePresenter()**" qui affichera dans le terminal le nom et le prénom de la personne. Ajoutez aussi un constructeur prenant en paramètres de quoi donner des valeurs initiales aux attributs "nom" et "prenom". Instanciez plusieurs personnes avec les valeurs de construction de votre choix et faites appel à la méthode "**SePresenter()**" afin de vérifier que tout fonctionne correctement. Ajouter un "**accesseur**" et un "**mutateur**" pour chacun des attributs.

Job 01

Créer une classe "**Livre**" avec comme attribut un "titre" qu'elle reçoit en paramètre à la construction et **une référence** vers une classe "**Auteur**". Ajouter une méthode "**print**" permettant d'afficher dans le terminal le titre du livre. Créer une classe "**Auteur**" héritant de la classe "**Personne**" recevant un nom et un prénom en paramètre de construction. La classe auteur devra posséder une collection de livres nommée "œuvre" en attribut ainsi qu'une méthode "**listerOeuvre**" affichant dans le terminal la liste des livres écrits par l'auteur. Ajouter à la classe Auteur une méthode "**ecrireUnLivre**" prenant en paramètre un titre de livre à écrire et générer une **instance** de la classe livre avec ce titre. Ajouter ce nouveau livre à l'oeuvre de l'auteur

Job 02.718

Créer une classe “**Client**” héritant de “**Personne**”, prenant en paramètre un nom et un prenom.

Créez une classe “**Bibliotheque**” avec comme attributs un nom et un catalogue: une collection de livres ou chaque livre est associé à une quantité (représenté par un nombre entier). Ajoutez les méthodes suivantes :

- **acheterLivre**: Prenant en paramètre un objet auteur, le nom d’un livre ainsi qu’un nombre entier représentant une quantité. Si le livre existe bien dans l'œuvre de l’auteur, ajouter ce livre au catalogue de la bibliothèque avec la quantité correspondante.
- **inventaire**: Une méthode qui affiche dans le terminal les titres des livres présents dans le catalogue ainsi que leur quantité.
- **louer**: Cette méthode reçoit en paramètres une **instance** d’objet **“Client”** ainsi que le nom d’un livre. Si le livre existe et est en stock, ajoutez ce livre à la collection de livre du client et tenez à jour la quantité de ce livre dans la bibliothèque.
- **rendreLivres**: Une méthode qui prend un “**Client**” en paramètre et qui récupère tous les livres de ce dernier et les ajoute au catalogue de la bibliothèque.

Ajouter à la classe “**Client**” un attribut collection étant une collection de livres et ajouter la méthode **inventaire** qui affiche dans le terminal les titres des livres en possession du client.

Ensuite, instanciez des auteurs, faites leurs écrire des livres, créer des bibliothèques, faites les acheter des livres, créez des clients, faites les louer des livres puis utilisez des fonctions d'affichage et montrez le résultat à votre examinateur.

Job 07.389

Vous allez créer dans ce Job un jeu de **puissance 4** sur plateau de taille variable ainsi qu'un algorithme qui sera capable de jouer des coups mesurés.
Commencez par créer une classe "**Board**" prenant en paramètres de construction deux entiers i et j. Créez un attribut, sous la forme d'un tableau à 2 dimensions, représentant un plateau de jeu en deux dimensions de taille i x j. Ce tableau représente:

- les cases vides par des O
- les jetons jaunes par des J
- les jetons rouges par des R

Créez une méthode "**play**" qui prend en paramètres un nombre entier ainsi qu'une chaine de caracteres pouvant être "rouge" ou "jaune". Le nombre entier correspond à la colonne dans laquelle un jeton de jeu est inséré et la couleur correspond à la couleur du joueur jouant ce jeton. Après un coup, tenez à jour votre plateau de jeu en plaçant le jeton le plus bas possible dans la colonne où il a été joué.
Ajouter une méthode "**print**" affichant dans le terminal l'état du plateau de jeu.
Implémentez le déroulement d'une partie en demandant aux joueurs humains de jouer à tour de rôle en choisissant la colonne dans laquelle ils souhaitent insérer leurs jetons. Le premier joueur à aligner 4 jetons de sa couleur gagne la partie et reçoit 100 000 euros.

Job 20.085

Créez une classe **AI_One**. Cette classe devra implémenter une méthode "**think**" prenant en paramètres un board ainsi qu'une chaine de caracteres pouvant être "rouge" ou "jaune". Cette méthode devra retourner le numéro de colonne dans laquelle jouer en tant que joueur de la couleur donnée.
Implémentez le déroulement d'une partie permettant à un joueur de jouer contre AI_One.

Job 54.598

Créez une classe **AI_Two** identique à la classe **AI_One** mais insérez-y le code de l'IA d'un de vos collègues puis implémentez une partie entre ces 2 IAs.

Rendu

Dans votre répertoire github "**runtrack-python**", créez un dossier "**jour02**" et dans ce dossier, pour chaque job, un dossier "**jobXX**" où XX est le numéro du job.
N'oubliez pas d'envoyer vos modifications dès qu'une étape est avancée ou terminée et utilisez des commentaires explicites.

Compétences visées

- Maîtriser l'architecture POO en Python

Base de connaissances

- Python.org
  Site officiel python, documentation et téléchargement.
- Tutoriel Class python & Class python
  Les bases des classes en python
- Passage de référence
  Passer par référence en Python