TP de Programmation avancée en Python : Série 4

Exercice1 :

Objectif :

Créer un système de gestion de paiement en ligne pour une application e-commerce. Ce système doit gérer différents types de paiments tels que le paiement par carte de credit, par PayPal et par virement bancaire.

1.--Classe abstraite `**Paiement**':

- Créer une classe abstraite **`Paiement`** avec les méthodes suivantes :

- `**validerPaiement()**' : valide les informations du paiement (par exemple, vérifier le solde, la validité de la carte, etc.).
- '**effectuerPaiement()**' : effectue le paiement après validation.
- '**afficherDetails()**' : affiche les détails du paiement (par exemple, le montant, le bénéficiaire).

2--Classes concrètes **`CarteCredit`, `PayPal`, 'VirementBancaire'** :

- Créer trois classes qui héritent de `**Paiement**` et qui implémentent les méthodes `**validerPaiement()**`, `**effectuerPaiement()**`et '**afficherDetails()**'.

- La classe `**CarteCredit**' (a les attributs solde et numero_carte) vérifie la validité de la carte et effectue le paiement.
- La classe `**PayPal**` (a les attributs solde et email) vérifie le compte PayPal.
- La classe `**VirementBancaire**` (a les attributs solde et banque) vérifie si le solde est suffisant pour effectuer le virement.

3.--Classe `**SystèmePaiement**`:

- Créer une classe `**SystèmePaiement**` qui permet de simuler des paiements et d'afficher les résultats.

- Cette classe doit afficher des messages appropriés pour chaque type de paiement.

4--Programme principal :

- Créer plusieurs objets : `**CarteCredit**`, `**PayPal**`, `**VirementBancaire**`.

- Regrouper-les dans une liste passée à `**SystèmePaiement**`.

- Simuler un paiement (montant au choix).

- Vérifier que le polymorphisme fonctionne : chaque méthode doit utiliser sa propre version de `**validerPaiement()**`, `**effectuerPaiement()**` et `**afficherDetails()**`.

Exercice 2 : Gestion des fichiers et répertoires avec **pathlib**

Consignes :1 –Créez un répertoire nommé MesProjets.

1. À l'intérieur, créez trois sous-répertoires : ProjetA, ProjetB, ProjetC (utilisez uniquement la méthode mkdir())
2. Dans le dossier ProjetA :
   - créez deux fichiers **notes.txt**, et **todo.txt**,
   - écrivez trois lignes de texte dans chacun des deux fichiers.
3. Toujours dans ProjetA, effectuez :
   - la lecture complète de notes.txt,
   - le comptage du nombre de lignes,
   - l'affichage du message : « Le fichier notes.txt contient X lignes. »
4. En utilisant une seule instruction, listez tous les fichiers présents dans *MesProjets/ProjetA*, puis n'affichez que ceux dont l'extension est .txt.
5. Dans le dossier ProjetB, créez un fichier ancien.txt, déplacez-le dans ProjetC, puis renommez-le en archive.txt. (Vous pouvez utiliser rename() ou replace().)
6. Supprimez :
   - le fichier todo.txt dans *ProjetA*,
   - le dossier ProjetC, uniquement s'il est vide. Sinon, affichez : « Impossible de supprimer ProjetC : le dossier n'est pas vide »

Exercice 3 :

On considère le problème: Trouver la fonction :

$$(P)\begin{cases} x'(t) = 2x(t)\left(1 - \dfrac{x(t)}{4}\right), & t \in]0,5], \\ x(0) = 1, \end{cases}$$

Ecrire un programme Python pour la résolution numérique de (P) et ploter la solution.