TP Python : Gestion d'une bibliothèque

Objectif : Créer une application console en Python qui simule la gestion d'une bibliothèque en utilisant toutes les notions avancé : **heritage**, **abstraction**, **enum**, **gestions des exceptions**.

Taches :

1. Créer une classe abstraite `Document` qui représentera l'élément de base de votre bibliothèque. Elle contiendra :
    - un attribut de classe `nb_document` (int)
    - la propriété `titre` (String)
    - la propriété `anneePublication` (Int)
    - une méthode abstraite `afficherInfos()` qui sera redéfinie pour afficher les détails spécifique
    - une méthode de classe `afficherNbDocument()` qui affichera le nombre de document instancié
2. Créer une classe enum `Genre` qui permettra de classifier les livres par catégorie.
    - Ex: ROMAN, SCIENCE_FICTION, FANTASTIQUE
3. À partir de `Document`, créer deux classes :
    - Une classe `Livre` qui contiendra :
        - une propriété `auteur`: String,
        - une propriété `nbPages`: Int
        - une propriété `genre`: Genre (enum)
        - un **constructeur secondaire** (methode static) qui ne prend que le `titre`, l'`auteur` et le `genre` (nbPages sera à 100 et anneePublication à 0 par défaut)
        - une méthode static `compteurPages()` qui prend une liste de Livre et retourne le total de pages.
    - Une classe `Magazine` qui contiendra :
        - une propriété `numero`: Int
4. Créer une classe abstraite `Empruntable` que sera hérité par `Livre` uniquement contenant :
    - une propriété `estEmprunte`: Boolean,
    - une méthode abstraite `emprunter()`
    - une méthode abstraite `rendre()`
5. Créer une classe abstraite `Consultable` qui sera implémenté par `Livre` et `Magazine` contenant :
    - une méthode abstraite `consulter()` qui affiche : "Vous consultez ce document."
6. Créer 2 exceptions personnalisé : `DocumentDejaEmprunteException` et `DocumentNonEmprunteException`
7. Implémenter les méthodes `emprunter()` et `rendre()` dans `Livre` afin quelle consulte le booléen `estEmprunte` :
    - Si nous pouvons l'emprunter ou le rendre alors afficher un texte de succès.
    - Sinon renvoyé l'exception correspondante.

8. Dans la fonction principale :
    - Créez plusieurs livres (de genres différents) et magazines.
    - Stockez-les dans une liste.
    - Affichez la liste complète des documents avec afficherInfos().
    - Simulez plusieurs actions :
        - consultation d'un livre ou d'un magazine
        - emprunt d'un livre
        - tentative d'emprunt d'un livre déjà emprunté
        - tentative de rendre un livre non emprunté
        - retour d'un livre emprunté

Bonus :

- Chacune des classes doivent être dans des scripts à part, puis importer dans le script principal.
- Créer un IHM permettant à l'utilisateur d'effectuer toutes les actions précédentes jusqu'à ce qu'il quitte le programme :

```
====== GESTION BIBLIOTHEQUE ======
1. Consulter
2. Emprunt
3. Restitution
0. Quitter
```

- Nous pourrons également ajouter une option `Ajouter Document` et `Retirer Document`

Exemple :

```
--- Liste des documents ---
Livre: "The Witcher", Andrzej Sapkowski, 1993, 320 pages, Genre: FANTASTIQUE
Magazine: "Canard PC", n°420, 2023
Livre: "Harry Potter", J.K. Rowling, 0000, 100 pages, Genre: ROMAN

Consultation du magazine "Canard PC"...
Vous consultez ce document.

Tentative d'emprunt du livre "The Witcher"...
Emprunt réussi !

Tentative d'emprunt du livre "The Witcher" à nouveau...
ERREUR : Ce livre est déjà emprunté !

Tentative de rendre le livre "Harry Potter" sans l'avoir emprunté...
ERREUR : Ce document n'a pas été emprunté !
```

```
Retour du livre "The Witcher"...
Le livre est maintenant disponible.
```