Tests unitaires

Exercices

- 3 exercices à réaliser au total.

Mise en place du projet

- **Forkez** le projet ***demo-tests-unitaires***
- **Clonez-le** dans votre **workspace**.
- **Importez** le projet dans STS en tant que projet MAVEN
- Recherchez la dernière version de **junit** / **junit** sur Maven Central at ajoutez la dépendance correspondante dans votre pom.xml en scope test.
- **Attention**, vos classes de tests unitaires devront être placées dans le « source folder » **src/test/java**

Exercice 1

Package : fr.diginamic.enumerations
A tester : Saison

Dans cet exercice vous devez vérifier que la méthode **valueOfLibelle** dans l'énumération **Saison** fonctionne correctement :

- Créez une classe de tests unitaires afin de vérifier que les cas nominaux fonctionnement.
- Testez également les cas aux limites afin de vérifier si la méthode est robuste ou non.

Si vous trouvez un bug ou si les tests de robustesse ne sont pas concluants, corrigez Saison.

Commitez et Pushez sur GitHub

Exercice 2

Avant-Propos sur la Distance de Levenshtein

La **distance de Levenshtein** permet de calculer la « distance entre 2 mots », c'est-à-dire le nombre de lettres à changer pour passer d'un mot à un autre.

Cet algorithme permet par exemple à un moteur de recherche de vous proposer des résultats pertinents même si vous faites des erreurs d'orthographe. Supposons que vous tapiez le mot « Jva » au lieu de « Java », le moteur de recherche, ne connaissant pas ce mot, va rechercher tous les mots à une distance de Levenshtein de 1, comme « Java ».

Exemples de **distance de Levenshtein**:

1) Entre « chat » et « chats » la distance de Levenshtein vaut 1 car il faut :
   - **ajouter** une seule lettre (la lettre s) pour passer du mot « chat » au mot « chats ».
   - ou **supprimer** une seule lettre pour passer du mot « chats » à « chat ».

2) Entre « machins » et « machine » la distance de Levenshtein vaut 1 car il faut **remplacer** une seule lettre pour passer d'un mot à l'autre.

3) Entre « aviron » et « avion » la distance de Levenshtein vaut 1 car il faut **retirer** la lettre r du premier mot (ou ajouter selon le sens).

4) Entre « distance » et « instance » quelle est la distance ?

5) Entre « Chien » et « Chine » quelle est la distance ?

TACHES A REALISER

Package : fr.diginamic.utils
A tester : StringUtils

Cette classe a été trouvée sur internet mais nous souhaitons la tester correctement avant de l'intégrer à notre projet.

Réalisez une **classe** de tests unitaires qui permet de tester la classe **StringUtils** en respectant les instructions suivantes :

a) Couvrez les cas de tests nominaux en testant différents cas
b) Intéressez-vous également à la robustesse de cette classe. Que se passe t'il si on passe à cette classe des paramètres NULL ? Proposez un correctif pour rendre cette classe plus robuste.

COMMITEZ ET PUSHEZ SUR GITHUB

EXERCICE 3

Package : fr.diginamic.immobilier
A tester : Maison

Réalisez une **classe** de tests unitaires qui permet de tester la classe **Maison** en respectant les instructions suivantes :

a) Testez les différentes méthodes de cette classe
    a. Si vous trouvez des bugs, corrigez les
b) Intéressez-vous également à la robustesse de cette classe. Que se passe t'il si on passe à cette classe des paramètres NULL ? Proposez un correctif pour rendre cette classe plus robuste.

COMMITEZ ET PUSHEZ SUR GITHUB