# DESIGN PATTERNS – TP STRATEGY

## OBJECTIF

- Mettre en place le pattern **Strategy**

## INSTRUCTIONS

- Dans le projet **java-design-patterns**, ouvrez le répertoire **fr.diginamic.strategy**
- Dans ce package vous avez une classe **DemoTri** qui permet de tester 3 algorithmes de tri différents.
- C'est la méthode **exec** de la classe Tri qui contient ces 3 algorithmes.
- La méthode **exec** possède 2 paramètres :
    - le premier est le type de tri choisi
        - si typeTri==1 alors on applique le Bubble sort algorithme
        - si typeTri==2 alors on applique le Insertion sort algorithme
        - Si typeTri==3 alors on applique le selection sort algorithme
- Comme vous le constatez sans doute, cette méthode est très mal écrite.
- Refactorez cette méthode en utilisant le pattern Strategy :
    - Etape 1 : créez une interface **Strategy** avec une méthode **void trier** qui prend en paramètre un tableau d'entiers.
    - Etape 2 : créer 3 classes qui implémentent cette interface. Chacune de ces classes implémentent un algorithme de tri spécifique
    - Etape 3 : créer une classe **StrategyFactory** qui retourne une instance de l'une de ces classes en fonction d'un type.
    - Etape 4 : Pour le type de tri, utilisez une énumération.
    - **Etape 5 : refactorez maintenant la méthode exec afin d'utiliser le pattern strategy.**

### COMMITEZ SUR GITHUB


© Tous droits réservés à DIGINAMIC Formation