TD 1: Patron de conception (Le design pattern)

Exercice 1:

Nous allons concevoir une classe CompteBancaire qui permet de déposer ou retirer de l'argent sur un compte. Mais nous souhaiterions pouvoir afficher les opérations (effectuées ou refusées) dans la console en cas de problème. Pour cela, nous allons implémenter une classe distincte nommée Journalisation avec deux méthodes propres à 'utilisation de cette classe que sont *ajouterLog(string)* et *afficherLog()*.

1. **Comment garantir que notre programme va utiliser une seule et même instance de la classe Journalisation.**

Exercice 2:

Écrivez une classe abstraite Voiture avec trois sous-classes Fiat, Peugeot, Audi, et une factory qui crée la bonne voiture selon le budget de l'acheteur, sachant qu'une Fiat coûte moins de 10000 euros, une Peugeot entre 10000 et 19999 et un Audi 20000 ou plus.

Exercice 3:

L'objet Catalogue peut utiliser ces sous-classes concrètes pour instancier les produits. Pour chaque produit, nous disposons d'une classe abstraite, d'une sous-classe concrète décrivant la version du produit fonctionnant à l'essence et d'une sous-classe décrivant la version du produit fonctionnant à l'électricité. Pour l'objet scooter, il existe une classe abstraite Scooter et deux sous-classes concrètes ScooterÉlectricité et ScooterEssence.

L'objet Catalogue peut utiliser ces sous-classes concrètes pour instancier les produits. Cependant si, par la suite, de nouvelles familles de véhicules doivent être prises en compte par la suite (diesel ou mixte essence-électricité), les modifications à apporter à l'objet Catalogue peuvent être assez lourdes.

1. **Quel *design-pattern* doit-on utiliser ?**
2. **Implanter celui-ci.**

Exercice 4:

Un nouveau dispositif IoT (Internet of Things) de sécurité dédié à la maison. Equipé de plusieurs capteurs il écoute le bruit d'une maison, analyse l'environnement et notifie ses utilisateurs via une application, cette dernière peut être installée sur n'importe quel

plateforme mobile (iOS, Android) pour pouvoir être notifié si quelque chose sortant de l'ordinaire se passe.

1. **Quel design-pattern doit-on utiliser ?**
2. Implanter celui-ci.

Exercice 5:

L'entreprise HébergeKech offre trois services d'hébergement : perso, pro, et premium. Pour chaque offre, plusieurs services supplémentaires peuvent être ajoutés (SSL, support 24/24, sauvegardes, Google Adwords, mySQL). Vous devez concevoir une application qui permette de calculer facilement le coût mensuel du service, pour chaque offre en tenant compte de la combinaison des services supplémentaires.

3. Quel design-pattern doit-on utiliser ?
4. Implanter celui-ci.