Nom : BLANCHON Prénom : Julien Groupe TP : B

Les « … » indiquent les endroits à compléter.

=====[Temps passé]==========

Ces informations de temps sont demandées à titre indicatif et ne sont pas prises en compte dans la notation du projet.

Toutes les durées sont à saisir en heures. Par exemple, si vous avez passé 1h45, il faudra indiquer 1.75. Si vous avez passé 2h30, il faudra indiquer 2.5.

- Temps passé sur la V1 (en h) : 7h
- Temps passé sur la V2 (en h) : 0h

=====[Questions]==========

Pourquoi l'exception OperationInterditeException ne peut pas être définie comme vérifiée par le compilateur ?

OperationInterditeException traduit une **erreur dans le code** il est donc cohérant de prendre une exception non vérifiée.

Expliquer ce qu'il faut faire pour ajouter un nouveau niveau de jeu, par exemple la stratégie lente (C13). Pour plus de précision, on numérotera les étapes à faire.

---

1. Crée la classe Lente implémentant Strategie

Listing 1: Lente.java

```
package allumettes;
/** Défini les propriétés de base de la stratégie "Lente"
 * d'un joueur du jeu des allumettes.
 * @author  Julien Blanchon
 * @version 1.0
 */
public class Lente implements Strategie {

    public static final String nom = "lente";

    /** Obtenir le nom de la stratégie : lente.
     * @return nom de la stratégie
     */
    @Override
    public String getNom() {
        return nom;
    }
```

```
    /** Obtenir le choix de la stratégie lente.
     * Toujours prendre 1 allumette.
     * @return nombre d'allumettes à prendre.
     */
    @Override
    public int getPrise(Jeu jeu, String nom) {
        return 1;
    }
}
```

2. Ajouter le choix de stratégie “lente” dans lors du choix utilisateur dans Jouer.java.

Listing 2: Jouer.java l32..47

```
switch (parts[1].toLowerCase()) {
    case Humain.nom : strategie = new Humain();
        break;
    case Naif.nom: strategie = new Naif();
        break;
    case Rapide.nom: strategie = new Rapide();
        break;
    case Tricheur.nom: strategie = new Tricheur();
        break;
    case Expert.nom: strategie = new Expert();
        break;
    case Lente.nom: strategie = new Lente();
        break;
    default:
        throw new ConfigurationException("Unexpected␣value:␣" + parts[1].toLower
}
```

3. Modifier la documentation d’afficherUsage.

Listing 3: Jouer.java l80..95

```
/** Afficher des indications sur la manière d'exécuter cette classe. */
public static void afficherUsage() {
    System.out.println("\n" + "Usage␣:"
        + "\n\t" + "java␣allumettes.Jouer␣joueur1␣joueur2"
        + "\n\t\t" + "joueur␣est␣de␣la␣forme␣nom@stratégie"
        + "\n\t\t" + "strategie␣=␣naif␣|␣rapide␣|␣expert␣|␣humain␣|␣tricheur"
        + "␣|␣lente"
        + "\n"
        + "\n\t" + "Exemple␣:"
        + "\n\t" + "␣␣␣␣java␣allumettes.Jouer␣Xavier@humain␣"
              + "Ordinateur@naif"
```

```
            + "\n"
            );
    }
```

Expliquer ce qui permet, dans votre conception, de changer dynamiquement (en cours d'exécution du programme) la stratégie d'un joueur (C14).

Pour changer la stratégie d'un joueur il suffit d'appeler la méthode Joueur#setStrategie((Strategie strategie)) de la poignée du joueur.

Par exemple l'arbitre peut changer la stratégie d'un joueur en 'lente si il triche:

Listing 4: Arbitre.java l50..52

```
if (!aTricher) {
    if (j_actuelle==this.joueur1){
        this.joueur1.setStrategie(new Lente());
    } else {
        this.joueur2.setStrategie(new Lente());
    }
}
```

=====[Explications]==========

Donner ici les explications supplémentaires utiles à la compréhension du travail rendu.

...
