Implémentation des aspect

Pour implémenter les aspects dans le projet nous avons décidé d'utiliser dans le projet le gestionnaire de dépendance maven. Cela nous a donc permis d'implémenter dans le projet la librairie AspectJ, grâce à cela, nous avons pu réaliser les aspects demandés dans le projet, à savoir toutes les vérifications de coup jouer dans la partie d'échec à la fois par le joueur humain mais aussi par le joueur IA.

AspectJ est une extension orientée aspect, pour le langage de programmation Java.
AspectJ est devenu le standard, du fait de son utilisation répandue, pour la Programmation orientée aspect en mettant l'accent sur la simplicité et la facilité de mise en œuvre pour les utilisateurs finaux.

Les aspects nous ont donc permis d'ajouter dans le projet toute la logique de vérification des coups comme :

- Ne pas sauter une pièce
- Ne pas bouger un pion qui n'est pas au même joueur
- Vérifier si le mouvement est faisable par une pièce en question
- Vérifier si le coup reste dans l'échiquier
- ….

Voici quelques messages d'erreur que nous avons implémenter lorsque les coups des joueurs ne sont pas valide :

```
1| T C P D R P C T
2| P P P P P P P P
3|
4|
5|
6|
7| p p p p p p p p
8| t c p d r p c t
  ----------------
   a b c d e f g h

Votre coup? <a2a4> a1a5
legal move
Vous êtes bloqués par une pièce
```

```
Votre coup? <a2a4> a0a5
Cette pièce n'est pas dnas l'échéquier
```

La programmation orientée aspect fonctionne comme ceci :

Nous établissons des points d'écoute appelés **PointCut**, qui nous permet d'exécuter du code lorsque le programme exécute une fonction mise en écoute, dans le cas de notre TP nous avons mis un **PointCut** sur la fonction makeMove de la classe Player. Étant donné que HumanPlayer et IAPlayer hérite de la classe Player, cela nous permet donc d'exécuter notre aspect lorsque l'un ou l'autre joueur réalise un mouvement. Lorsque cela se produit, nous vérifions si le coup est valide ou non.

Ainsi, grâce à la programmation aspect, nous avons séparé le code métier de la logique de vérification et de Log.

Les modifications que nous avons apporté au projet :

Dans la fonction **makeMove** du joueur IA et humain, nous avons pris la décision de rajouter en argument le Board de la partie pour que nous puissions y avoir accès dans l'aspect et y faire nos vérifications.

Dans la classe **Move** nous avons rajouté 2 booléens, un appelé **canMove**, qui nous permet de valider un coup ou non et de laisser le joueur rejouer un coup ci ce dernier n'est pas valide.
Un autre appelé **human,** qui nous permet de savoir quel joueur vient de jouer le dernier coup. Cette information nous aide dans nos vérifications.