POINT_ FAIBLE

a. Il existe le singleton et les enums, donc un peu complexe de faire les tests unitaire.
(il faut mettre **`initial( )`** **`@`**BeforeEach)

b. Il existe seulement deux bonus dans la version actuelle, le jeu est simple donc pour le Robot intelligent, il existe peu de progress à faire, il faut ajouter des bonus et compliquer le jeu.

c. Le Robot est conçu pour une interface implémenté par RobotSimple et RobotNormal, mais on trouve maintenant qu'il existe quelques communes méthodes. (Dupication de code)

d. Pour créer un nouveau jeu, il a besoin un Map<String,Level>.

POINT _FORT

a. On adopte Singleton (Table) et Enum (Invention, Inventeur…) donc il est facile de les utiliser dans les class différents.
b. Utiliser Loggers, avec qui on peut savoir clairement qui annonce les informations et on peut le mettre OFF si on veut.
c. Responsibility-driven design, moins de couplage.
d. Avoir une fondation solide, favoriser le développement après.
e. Les Robots peut difficilement faire du mal.
f. Si on veut ajouter un joueur réelle, il a seulement besoin de implémenter Robot.
g. On a une classe pour faire statistique de taux de victoires, on peut savoir clairement la capacité de Robot.
h. Plusieurs joueur, on peut décider le nombre de joueur.
i. Presque presque fini.

DÉTAIL_*CLASS List*

Fonctionner Le Jeu

- GameEngine.java
- PlayerConsole.java
- Table.java

Joueurs

- Robot *-Interface*
  - *RobotSimple.java*
  - *RobotNormal.java*

Enums

- **Invention**
- **Inventor**
- **Level**
- **PlayerColor**
- **Skill**
- **Ticket**

Main

- Statistics.java
- Main.java

DÉTAIL_*Fonctionner Le Jeu*

- Table.java

Principaux Méthodes

```
public static Table getInstance()
boolean putInvention(Invention invention)
List<Invention> getInventions()
List<Invention> getNotFinished()
void removeAll()
void removeFinished()
void initialInventions()
```

- Présenter les inventions valides dans le jeu
- Présenter les opération valides d'inventions

DÉTAIL_*Fonctionner Le Jeu*

- PlayerConsole.java

Principaux Méthodes

```
PlayerConsole(PlayerColor color)
```
- Créer une console selon PlayerColor

```
boolean send(Inventor inventor,
             Invention invention)
boolean setAllFree()

boolean useTicket(Ticket ticket)
```

- Comme l'interface entre Joueur et Jeu.
- Présente tous les actions valides.
- Limiter le nombre d'opération de joueur dans un round.
- Détecter les événements comme Ajouter Point ou Finir qqch et les annoncent à GameEngine.
- Avoir une liste d'inventeurs qui est initialisé selon le PlayerColor, des inventeurs de certain PlayerColor sont fixes.

DÉTAIL_*Fonctionner Le Jeu*

- GameEngine.java

Principaux Méthodes

```
GameEngine(Map<String, Level> build)
•    Map : { Robot Name, Robot Level }
•    Build Game

void gameStart()
•    Démarrer le jeu

List<PlayerColor> getWinner()

void initialGame()

Robot getRobot(PlayerColor color)
•    Trouver le Robot de certain couleur
```

- Créer les PlayerConsole de couleur diffèrent d'après le nombre de joueur.
- Ajouter les inventions random à la Table d'après le nombre de joueur.
- Ajouter les Tickets (Bonus) random aux inventions.
- Établir la liaison entre joueur(Robot) et PlayerConsole
- Possède et opère le Tableau de Score d'après l'annonce de PlayerConsole
- Distribuer les Tickets(Bonus) d'après l'annonce de PlayerConsole et l'information (taille de la contribution) présenté par certaine invention.
- Régler l'ordre de round et jeu.

DÉTAIL_*Enums, Robot, Main*

Enums

- *Invention*
  - void Initial()
  - 12 (première époque)
- *Inventor*
  - void initial()
  - 4*5=20

  (quatre inventeurs pour un couleur)
- *Level*
  - Normal
  - Simple
- *PlayerColor*
  - 5 couleurs (joueur)
- *Skill*
  - 4 competences
- *Ticket*
  - ADD A POINT
  - SET ALL FREE

Robot

- Simple
  - Faire tous les chose au hasard
- Normal
  - Pouvoir calculer la value de certain opération

Main

- Statistics.java
  - Lancement de jeu
- Main.java
  - Statistique de taux de victoires

À faire

Enums

- *Invention*
  - void Initial()
  - 12 (première époque)
- *Inventor*
  - void initial()
  - 4*5=20
  (quatre inventeurs pour un couleur)
- *Level*
  - Normal
  - Simple
- *PlayerColor*
  - 5 couleurs (joueur)
- *Skill*
  - 4 competences
- *Ticket*
  - ADD A POINT
  - SET ALL FREE

Trois époque avec 36 inventions

Améliorer les Robots

Cinq tickets