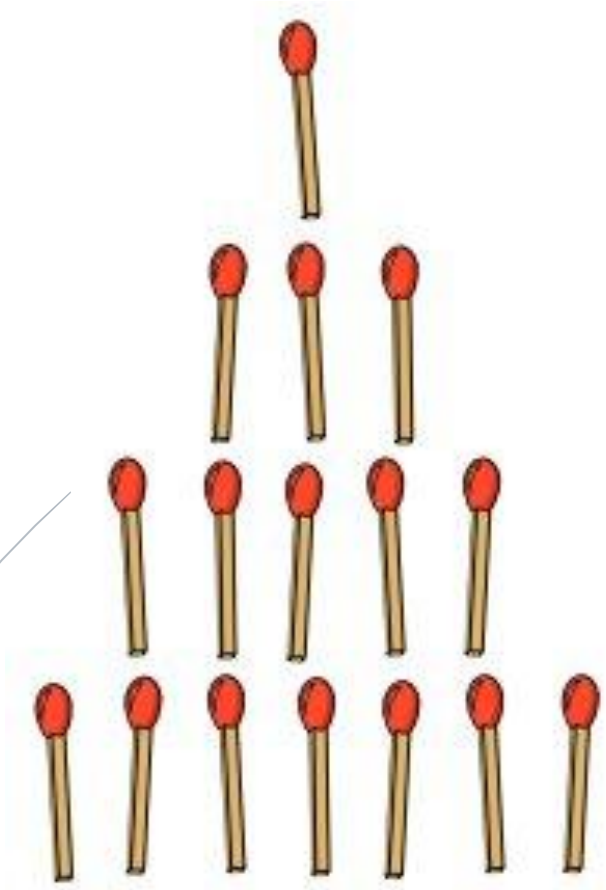
Jeu de Marienbad

SAE 1.01 :

Implémentation d'un besoin client



Enrick MANANJEAN

Killian AVRIL

GROUPE : 1B2

Jeu de Marienbad

Partie 1 : joueur contre joueur

Exemple d'exécution de la partie joueur contre joueur :

```
**Jeu de Marienbad**
Nom du joueur 1 : Enrick
Nom du joueur 2 : Killian
Choisissez le nombre de lignes : 4
1 : Enrick
2 : Killian
Qui va commençer? : 1
Enrick et Killian, C'est parti !

État actuel du jeu :
0 :  |
1 :  |  |  |
2 :  |  |  |  |  |
3 :  |  |  |  |  |  |  |
Enrick, c'est à toi de jouer.
Où veux-tu retirer des allumettes: 3
Combien : 5
Ça marche!

État actuel du jeu :
0 :  |
1 :  |  |  |
2 :  |  |  |  |  |
3 :  |  |
Killian, c'est à toi de jouer.
Où veux-tu retirer des allumettes: 2
Combien : 5
Ça marche!

État actuel du jeu :
0 :  |
1 :  |  |  |
2 :
3 :  |  |
Enrick, c'est à toi de jouer.
Où veux-tu retirer des allumettes: 3
Combien : 2
Ça marche!
```

```
État actuel du jeu :
0 :  |
1 :  |  |  |
2 :
3 :
Killian, c'est à toi de jouer.
Où veux-tu retirer des allumettes: 1
Combien : 2
Ça marche!

État actuel du jeu :
0 :  |
1 :  |
2 :
3 :
Enrick, c'est à toi de jouer.
Où veux-tu retirer des allumettes: 1
Combien : 1
Ça marche!

État actuel du jeu :
0 :  |
1 :
2 :
3 :
Killian, c'est à toi de jouer.
Où veux-tu retirer des allumettes: 0
Combien : 1
Ça marche!
Félicitations! Killian a gagné !


------------------
(program exited with code: 0)

Appuyez sur une touche pour continuer...
```

Exécution avec la méthode teste :

testJeu_allu() :

Permet de vérifier si le nombre d'allumette est bien retirer dans la ligne

```
*** testJeu_allu()
Test Jeu_allu(jeuInitial, niv=0, nb_allumettes=2)
ok
Test Jeu_allu(jeuInitial, niv=0, nb_allumettes=3)
ok
Test Jeu_allu(jeuInitial, niv=1, nb_allumettes=1)
ok
Test Jeu_allu(jeuInitial, niv=0, nb_allumettes=1)
ok
```

testJeuTermine() :

Permet de vérifier s'il y encore des allumettes

```
*** testJeuTermine()
Test jeuTermine(lignes=[0])
ok
Test jeuTermine(lignes=[0, 0, 0])
ok
Test jeuTermine(lignes=[1])
ok
Test jeuTermine(lignes=[0, 2, 0])
ok
Test jeuTermine(lignes=[1, 0, 0])
ok
```

testInitialiserJeu() :

Permet de vérifier si le tableau a bien été créé (le bon nombre de ligne avec le bon nombre d'allumettes)

```
*** testInitialiserJeu()
Test initialiserJeu(nbLignes=1)
ok
Test initialiserJeu(nbLignes=2)
ok
Test initialiserJeu(nbLignes=3)
ok
Test initialiserJeu(nbLignes=4)
ok
Test initialiserJeu(nbLignes=5)
ok
```

testJoueurCourant()

Permet de vérifier si les joueurs jouent bien chacun leur tour

```
*** testJoueurCourant()
Test joueurCourant(val=1, j1=Alice, j2=Bob)
ok
Test joueurCourant(val=2, j1=Alice, j2=Bob)
ok
Test joueurCourant(val=1, j1=John, j2=Jane)
ok
```

Partie 2 : joueur contre ordinateur

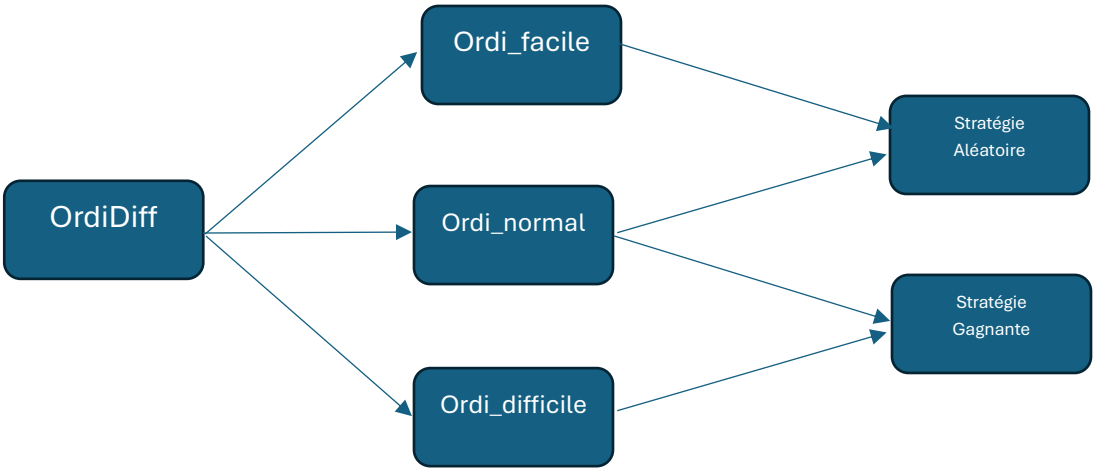
Exemples d'exécution de la partie joueur contre ordinateur
(pour chaque niveau de difficulté)

Ordinateur(facile).

```
**Jeu de Marienbad**
Votre nom : Enrick
1 : Facile
2 : Normale
3 : Difficile
Enrick, Choisissez le niveau de difficulté entre 1 et 3 : 1

**Facile**

Choisissez le nombre de lignes : 4
1 : Enrick
2 : ordi
Qui va commençer? : 1
Enrick c'est parti !

État actuel du jeu :
0 :  |
1 :  |  |  |
2 :  |  |  |  |  |
3 :  |  |  |  |  |  |  |
C'est à Enrick de jouer.
Où veux-tu retirer des allumettes: 3
Combien : 7
Ça marche!

État actuel du jeu :
0 :  |
1 :  |  |  |
2 :  |  |  |  |  |
3 :
C'est à ordi de jouer.
l'ordi a retirer 1 allumette(s) à la ligne 0

État actuel du jeu :
0 :
1 :  |  |  |
2 :  |  |  |  |  |
3 :
C'est à Enrick de jouer.
Où veux-tu retirer des allumettes: 2
Combien : 3
Ça marche!
```

```
État actuel du jeu :
0 :
1 :  |  |  |
2 :  |  |
3 :
C'est à ordi de jouer.
l'ordi a retirer 2 allumette(s) à la ligne 1

État actuel du jeu :
0 :
1 :  |
2 :  |  |
3 :
C'est à Enrick de jouer.
Où veux-tu retirer des allumettes: 2
Combien : 1
Ça marche!

État actuel du jeu :
0 :
1 :  |
2 :  |
3 :
C'est à ordi de jouer.
l'ordi a retirer 1 allumette(s) à la ligne 2

État actuel du jeu :
0 :
1 :  |
2 :
3 :
C'est à Enrick de jouer.
Où veux-tu retirer des allumettes: 1
Combien : 1
Ça marche!
Félicitations! Enrick a gagné !


------------------
(program exited with code: 0)

Appuyez sur une touche pour continuer...
```

Fonctionnement de l'ordinateur



teste sur initialisation et teste jeu terminer et joueur courant

1) Structure utilisée pour stocker le jeu.

Dans le code de la classe JeuMarienbad, la structure utilisée pour stocker l'état du jeu est un tableau d'entiers (int[] jeu).

{1, 3, 5, 7} exemple

Ce tableau permet de représenter le nombre de lignes et le nombre d'allumettes par ligne.

Le tableau jeu est créé dans la méthode initialiserJeu(int nbLignes), donc comme précisé avant chaque case du tableau représente une ligne et chaque ligne correspond a un nombre impaire d'allumette, déterminé par la formule 2 * i + 1, où i est l'index de la ligne.

Au fur et à mesure que la partie évolue, le tableau évolue donc aussi. Il est ainsi modifier dans la méthode Jeu_allu(). Le nombre d'allumettes dans la ligne choisie (index niv) est décrémenté de la valeur nb_allumettes.

Et pour finir la méthode jeuTermine() vérifie si toutes les valeurs du tableau sont à zéro, ce qui signifierait qu'il ne reste plus d'allumettes et que le jeu est terminé.

Conclusion, le tableau d'entiers int[] est la structure principale utilisée pour stocker l'état du jeu et suivre l'évolution du nombre d'allumettes sur chaque ligne.

2) Stratégie de l'ordinateur.

La stratégie de l'ordinateur est simple, son but est de gagner donc comment va-t-il le faire ? Il va tout simplement transformer le nombre d'allumettes par ligne en binaire.

Une fois cela fait il additionne toutes les valeurs d'une même colonne après ça il additionne toutes les colonnes entre elle. A ce moment la l'ordi vas soit essayer d'enlever toutes les allumettes de la même lignes (s'il en reste qu'une) sinon il vas faire en sorte de ne laisser qu'une nombre impaire pour qu'aux prochain tour il puisse gagner (s'il n'est toujours pas possible de gagner il recommence le procéder jusqu'à sa victoire)

3) Répartition du travail.

Le travail a été répartie de façon a ce qu'il soit de 50-50