ÉCOLE D'INGÉNIEURS DU LITTORAL

Rapport de projet
Puissance 4

Réalisé par :

Issam Salghat - TD1-TP2

Massine Mountasser - TD2-TP3

Encadré par:

Alboody Ahed
Fonlupt Cyril

Présentation

Puissance 4 est un jeu de société créé en 1974 par Howard Wexler sous la marque Milton Bradley. C'est un jeu de stratégie en 2 dimensions où deux joueurs s'affrontent à tour de rôle en tentant d'aligner quatres pions de la même couleur en horizontale, verticale ou en diagonale.

La table est sous forme de grille de six lignes et sept colonnes où les joueurs vont glisser les pions de leur couleur. Une partie est remportée par le premier joueur qui aligne quatre pions et une partie est déclarée nulle si aucun des joueurs n'arrive à les aligner et que toutes les cases sont pleines.

Dans le cadre de notre formation en programmation orientée objet en première année du cycle d'ingénieur, nous allons réaliser ce jeu en langage Java.

Règles

Début de partie

Pour commencer une partie de puissance 4, on désigne le joueur qui commence.

Déroulement de la partie

Il met un de ses jetons de couleur dans l'une des colonnes de son choix. Le jeton se retrouve en bas de la colonne. Le deuxième joueur insère à son tour son jeton, de l'autre couleur dans la colonne de son choix. Les joueurs répètent la manœuvre jusqu'à ce qu'un joueur aligne 4 pions.

Fin de la partie

Le premier joueur à aligner 4 pions, que ce soit horizontalement, verticalement ou en diagonale, remporte la partie.

Répartition des tâches:

Planning de travail - Binôme

Modélisation du déroulement - Massine

Diagrammes UML - Issam

La classe Grille - Massine

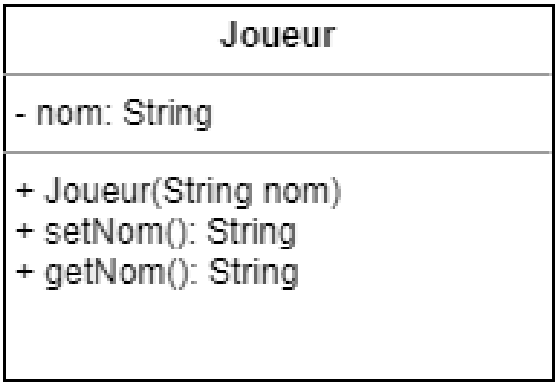
La classe Joueur - Issam

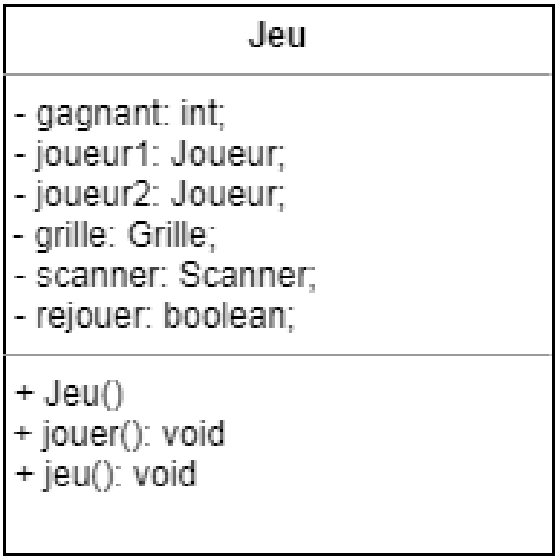
La classe Jeu - Binôme

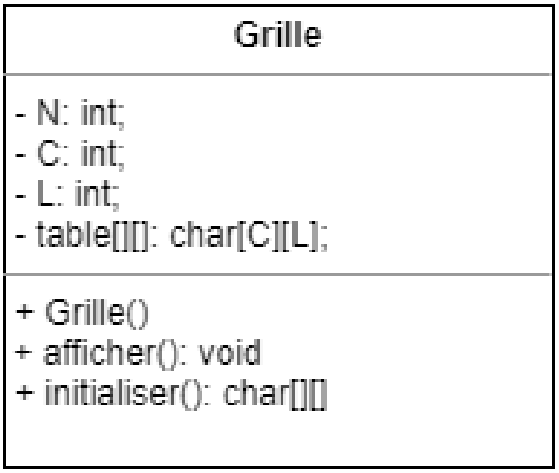
Main - Issam

Réalisation rapport - Massine

Diagrammes des classes UML:







Classes & Méthodes de Classes:

➔ Joueur

Cette classe permet de créer un joueur à partir du nom entré.

```java
public  class  Joueur
{
    private  String nom;
    static Scanner scanner = new Scanner(System.in);

    public Joueur (String nom)
    {
        this.nom = nom;
    }


    public  String getNom(){
        return nom;
    }

    public static String setNom() {
        System.out.println("Entrez votre nom: ");
        String nomJ=scanner.nextLine();
        return nomJ;
    }
}
```

➔ Grille

Cette classe permet de:

- créer une grille de jeu,
- initialiser la grille,
- afficher la grille.

```
public class Grille {
    public static int N=4;
    public static int C=7;
    public static int L=6;
    char[][] table = new char[C][L];

    public void Grille() {}
    public char[][] initialiser()
    {

        for(int x = 0 ; x < C ; x++)
            for(int y = 0 ; y < L ; y++)
                table[x][y] = '.';
        return table;
    }

    public void afficher() {
    System.out.println();

    for(int y = 0 ; y < L ; y++)
    {

        for(int x = 0 ; x < C ; x++){
            System.out.print(" " + table[x][y] + " ");
        }

        System.out.println();
    }

    System.out.println();

    for(int k = 1 ; k < 8 ; k++)System.out.print(" "+k+" ");
    System.out.println();
    }
}
```

➔ Jeu

C’est la classe qui gère le jeu en intégralité. Elle compte 2 méthodes:

- Jouer(): qui gère le déroulement d’une partie;

```java
public void jouer()
{
    grille.initialiser();
    for(int i = 1 ; i <= C*L ; i++)
    {
        grille.afficher();
        System.out.println("Tour de " +(i%2==1 ? joueur1.getNom(): joueur2.getNom() ));
        System.out.println("Entrez le numero de la colonne entre 1 et " + C );
        boolean placement = false;
        int colonne = -1;
        while(!placement){
            colonne = -1;
            String ligne = scanner.nextLine();
            try{
                colonne = Integer.valueOf(ligne);

                if(colonne >= 1 && colonne <= C){
                    if(grille.table[colonne - 1][0] != '.'){
                        System.out.println("Saisie invalide, rééssayez");
                    } else {
                        placement = true;
                    }
                } else {
                    System.out.println("Saisie invalide, rééssayez");
                }

            }catch(Exception e){System.out.println("Saisie invalide, rééssayez");}

        }
        int rang = L-1;
        while(grille.table[colonne - 1][rang] != '.'){
            rang--;
        }
        grille.table[colonne - 1][rang] = (i%2==1 ? 'R' : 'G');
```

Initialisation boule partie + gestion entrée joueur

```
char symbole = (i%2==1 ? 'R' : 'G');

int max = 0;
int x; int y;
int somme;

x = colonne-1; y = rang; somme=-1;
while(y >= 0 && x >= 0 && grille.table[x][y] == symbole){ y--; x--; somme++;}
x = colonne-1; y = rang;
while(y < L && x < C && grille.table[x][y] == symbole){ y++; x++; somme++;}
if(somme > max) max= somme;

x = colonne-1; y = rang; somme=-1;
while(y >= 0 && x < C && grille.table[x][y] == symbole){ y--; x++; somme++;}
x = colonne-1; y = rang;
while(y < L && x >= 0 && grille.table[x][y] == symbole){ y++; x--; somme++;}
if(somme > max) max= somme;

x = colonne-1; y = rang; somme=-1;
while(y >= 0 && grille.table[x][y] == symbole){ y--; somme++;}
y = rang;
while(y < L && grille.table[x][y] == symbole){ y++; somme++;}
if(somme > max) max= somme;

x = colonne-1; y = rang; somme=-1;
while(x >= 0 && grille.table[x][y] == symbole){ x--; somme++;}
x = colonne-1;
while(x < C && grille.table[x][y] == symbole){ x++; somme++;}
if(somme > max) max= somme;

if(max >= N)
{
    gagnant = (i%2==1 ? 1 : 2);
    i = C*L+1;
}
System.out.println("_____");
```

Gestion fin de partie

- Jeu(): méthode d'initialisation et fin du jeu

```
public void jeu()
{

    System.out.println("Joueur 1");
    joueur1= new Joueur(Joueur.setNom());


    System.out.println("Joueur 2");
    joueur2 = new Joueur(Joueur.setNom());
    while(rejouer) {
        Grille grille=new Grille();
        grille.initialiser();


        this.jouer();


        System.out.println();
        System.out.println("FIN DE PARTIE");
        System.out.println("_____________________");
        if(gagnant == 0)
            System.out.println("match null");
        if(gagnant == 1)
            System.out.println("Le vainqueur est "+joueur1.getNom());
        if(gagnant == 2)
            System.out.println("Le vainqueur est "+joueur2.getNom());
        System.out.println("_____________________");


        for(int k = 0 ; k < C+2+2*C ; k++)System.out.print('-');
        System.out.println();

        for(int y = 0 ; y < L ; y++){

            for(int x = 0 ; x < C ; x++){
                System.out.print(" " + grille.table[x][y] + " ");
            }

            System.out.println();
        }

        for(int k = 0 ; k < C+2+2*C ; k++)System.out.print('-');
        System.out.println();
        rejouer= false;
        System.out.println("Rejouer? Tapez 1 pour oui...");
        String again= scanner.nextLine();
        if(Integer.valueOf(again)==1) rejouer = true;
    }
}
```

➔ Main

Petite présentation visuelle du début, ainsi que l'appel à la méthode nouveau jeu.

```
public class Main
{

    public static void main(String[] args)
    {
        System.out.println("\n\n         ----Puissance 4----");
        System.out.println("         -----2 Joueurs-----");
        System.out.println("         --Nouvelle partie--");
        System.out.println("         --------------------\n\n");
        System.out.flush();


        Jeu nouveau=new Jeu();

        nouveau.jeu();
    }

}
```

Conclusion

Le travail sur ce projet nous a permis de mettre en œuvre nos apprentissages lors des cours et travaux pratiques, ainsi que la recherche de spécificités lors de l'écriture du programme. De plus, la bonne conception antérieure au codage est cruciale pour un travail efficace et propre.

Nous savons désormais qu'il est possible de modéliser un bon nombre de processus à l'aide de la programmation orientée objet, notamment en utilisant Java.