TP Puissance 4 – POO C#

Version : 1.0

Ce TP est noté

A rendre un fichier compressé (zip, 7z, rar) contenant :

- Le projet Visual Studio contenant l'ensemble des fichiers de la solution et du projet.

Ce TP est à réaliser seul ou par groupe de deux.

Nommer le fichier nom.prenom.b2.zip ou nom1.nom2.b2.zip et mettre celui-ci dans le dossier de rendu Beecome prévu à cet effet ou envoyer le fichier à l'adresse nicolas.chevalier@reseau-cd.net .

Exercice : Création du jeu Puissance 4

Le but du TP est de créer une application utilisant les bases du langage C# et les notions de la programmation orienté objet (classe, propriété, méthode, constructeur, héritage, substitution, polymorphisme)

Avant de commencer le TP, créer un projet application TPPuissance4

Instructions

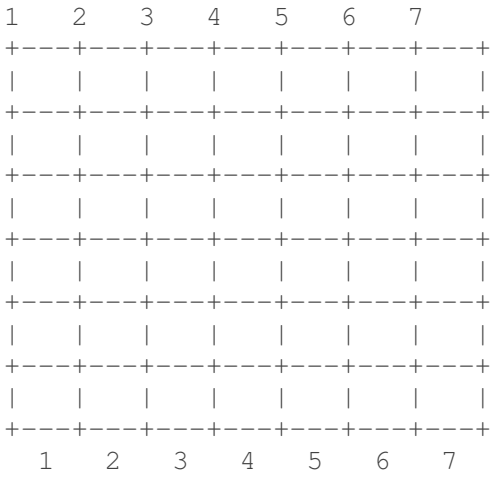
Le jeu Puissance 4 se joue à l'aide d'une grille verticale de sept colonnes sur six lignes. Chaque joueur dispose de vingt et un jetons d'une couleur (le plus souvent, rouge et jaune traduit dans notre jeu par les caractères « O » et « X ») et place ceux-ci au sein de la grille à tour de rôle.

Pour gagner le jeu, un joueur doit aligner quatre jetons verticalement, horizontalement ou en oblique. Il s'agit donc du même principe que le Morpion à une la différence que la grille est verticale ce qui signifie que les jetons tombent au fond de la colonne choisie par le joueur.

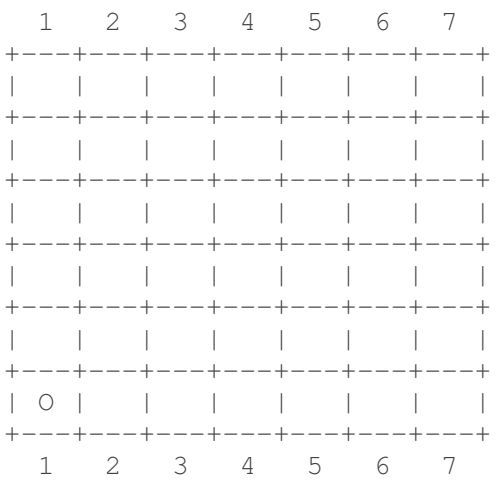
Vous trouverez la règle du jeu complète sur le site wikipedia : https://fr.wikipedia.org/wiki/Puissance_4.

Vous allez devoir réaliser le jeu Puissance 4 soit pour deux joueurs humains, soit pour un jour humain et un joueur géré par l'ordinateur. Il faudra donc proposer à l'utilisateur de choisir une partie entre deux joueurs ou entre un joueur et l'ordinateur. Le joueur devra mettre en place un algorithme simple d'intelligence artificielle qui est présenté par la suite.

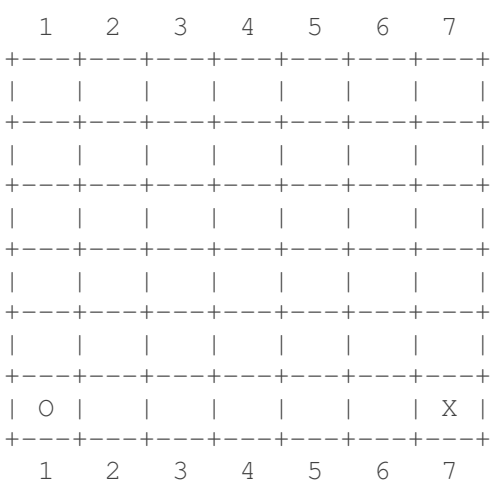
Dans le cas de deux joueur humain, le jeu devra ressembler à :




```
Joueur 1 : 1
```




```
Joueur 2 : 7
```



```
Joueur 1 :
```

….

Chaque joueur choisi à tour de rôle une colonne dans la grille ou placer son jeton. Si le joueur saisie autre chose, ou que la colonne indiquée est invalide, une nouvelle saisie lui sera demandé. A chaque tour, le programme devra vérifier si un joueur a gagné et si la colonne n'est pas déjà remplie. Dans ce cas le joueur devra rejouer. Le jeu s'arrête, soit quand un joueur a gagné, soit quand la grille est complète et qu'aucun joueur n'a réussi à gagner.

Joueur IA

Votre objectif sera de construire un petit système d'intelligence artificielle. Celle-ci permettra à un joueur de jouer seul contre l'ordinateur.

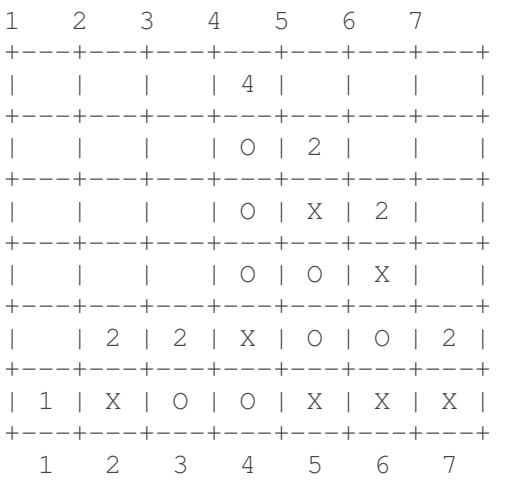
Afin de mettre en place le système d'intelligence artificielle, vous allez devoir réaliser un petit algorithme dont le principe est le suivant : pour chaque emplacement possible (il y en aura toujours au maximum sept), vous devez calculer combien de pièces composeraient la ligne si l'on joue à cet endroit sans tenir compte de notre couleur (autrement dit, le jeton que nous jouons est considéré comme étant des deux couleurs). Si une valeur est plus élevée que les autres, l'emplacement correspondant sera choisi. Si en revanche il y a plusieurs nombres égaux, une des cases sera choisie au hasard.

Cependant si lors de l'analyse, un coup permet à l'ordinateur de gagner, alors le traitement s'arrêtera et le mouvement permettant de gagner est immédiatement joué.

Par exemple, dans la grille ci-dessous, il est possible de jouer à sept endroits (indiqués à l'aide d'un point d'interrogation) :

```
1   2   3   4   5   6   7
+---+---+---+---+---+---+---+
|   |   |   | ? |   |   |   |
+---+---+---+---+---+---+---+
|   |   |   | O | ? |   |   |
+---+---+---+---+---+---+---+
|   |   |   | O | X | ? |   |
+---+---+---+---+---+---+---+
|   |   |   | O | O | X |   |
+---+---+---+---+---+---+---+
|   | ? | ? | X | O | O | ? |
+---+---+---+---+---+---+---+
| ? | X | O | O | X | X | X |
+---+---+---+---+---+---+---+
  1   2   3   4   5   6   7
```

Si on applique l'algorithme, on obtient les valeurs suivantes :



Le programme jouera donc dans la colonne quatre, cette dernière ayant la valeur la plus élevée.

Vous avez la possibilité d'amélioré cette algorithme, vous devez cependant au minimum mettre en place celui présenté précédemment.

Travail à faire :

Vous allez devoir concevoir l'application du jeu puissance 4 en programmation orienté objet :

1. Commencez par identifier les classes de l'application ainsi que les méthodes permettant de gérer une partie du jeu Puissance 4. Vous pouvez réaliser pour cela un diagramme UML en essayant de prévoir le fonctionnement de votre programme.
2. Implémentez les classes correspondantes en C# ainsi que le fonctionnement du jeu entre deux joueurs ou avec un joueur humain et l'ordinateur.
3. Testez en effectuant différentes parties pour vérifier le bon fonctionnement de votre application.

Consignes à respecter :

Les classes listées si dessous devront impérativement apparaitre dans votre programme.

Classe Jeu :

Cette classe est une classe de base permettant de représenter un jeu par son Nom. Prévoir les accesseurs ainsi que le constructeur de cette classe.

Classe Puissance4 :

Cette classe héritera de la classe Jeu et implémentera le jeu Puissance 4 avec une méthode Démarrer permettant de démarrer le jeu et de proposer au joueur de recommencer une nouvelle partie. Elle permettra aussi de demander au joueur si celui-ci veux jouer contre l'ordinateur ou contre un autre joueur humain. Elle indiquera enfin le résultat de la partie. Une boucle permettra de faire jouer les deux joueurs jusqu'à ce qu'il y ai un gagnant ou que la grille soit totalement remplie sans avoir de gagnant.

Classe Grille :

Cette classe permettra de gérer la grille du jeu. La grille sera constituée d'un tableau de char à 2 dimensions constituées de 6 lignes et 7 colonnes. Elle aura une méthode permettant d'afficher à chaque tour la grille et le choix effectué par chaque joueur. Elle devra avoir les méthodes suivantes :

- void Init() : permet d'initialiser la grille avec la valeur 0 dans chaque cellule.
- void Afficher() : permet d'afficher la grille avec le choix de chaque joueur
- char TestGagner() : retourne le joueur gagnant si un joueur à fait une ligne ou une diagonale de 4 jetons de suite.
- int GetLigne(int colonne) : permet de trouver la ligne disponible à partir d'une colonne.
- void Positionner(int ligne, int colonne, char jeton) : permet de positionner un jeton dans la grille suivant la ligne et la colonne.

Classe Joueur :

Cette classe est une classe de base permettant de représenter un joueur qui sera représenté par un Nom et un type de jeton (X ou O). Prévoir les accesseurs ainsi que le constructeur de cette classe. Elle devra avoir les méthodes suivantes :

- int GetColonne() : permet au joueur de choisir la colonne ou jouer et qui retourne la colonne choisie.
- void Jouer(Grille grille) : cette méthode sera réécrite dans les classe JoueurHumain et JoueurIA. Elle permet pour le JoueurHumain de choisir la colonne ou jouer et si celle-ci est valide de positionner le jeton dans la grille. Pour le JoueurIA elle permet de sélectionner une colonne ou jouer suivant l'algorithme IA et de positionner le jeton.

Classe JoueurHumain :

Cette classe héritera de la classe Joueur. Elle devra implémenter la méthode Jouer qui doit permettre à un joueur de choisir la colonne à jouer, de gérer les erreurs si un joueur choisi un mauvais numéro de colonne ou si la colonne choisie est déjà pleine, et de positionner le jeton.

Classe JoueurIA :

Cette classe héritera de la classe Joueur. Elle devra implémenter la méthode Jouer qui doit mettre en place l'algorithme IA afin de permettre à l'ordinateur de choisir une colonne ou jouer et de positionner le jeton.