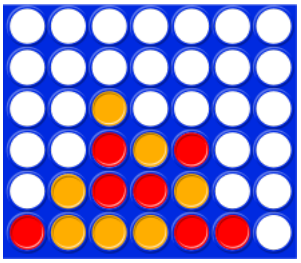
2023-2024 Université Paris 8
Algorithmique avancée Licence 3 Informatique

TP n°6 : IA pour le jeu de *Puissance 4*.

Le Puissance 4 est un jeu de stratégie à deux joueurs dans lequel le but est d'aligner une suite de 4 pions de même couleur sur une grille comptant 6 rangées et 7 colonnes. L'objectif de ce TP est de développer une IA capable de jouer au *Puissance 4* suivant un algorithme de type `minimax`, puis un algorithme de type `alpha-beta`.



Dans le fichier `puiss4.c`, vous trouverez toutes les fonctions de base du jeu:

- la fonction `joueur_gagne` permet de détecter si le joueur passé en paramètre a gagné la partie;
- les fonctions *colonne_ remplie* et *grille_ remplie* permettent respectivement de tester si une colonne est remplie (auquel cas on ne peut plus jouer dedans) où si la grille entière est remplie (auquel cas la partie doit se terminer).
- la fonction `pos_terminale` permet de tester si le jeu est en position terminale ou non.
- la fonction `jouer_jeton` permet d'ajouter un jeton correspondant au joueur en paramètre dans la colonne voulue; la fonction `retirer_jeton` permet l'action inverse.
- la fonction `inverser_joueur` permet d'échanger les deux joueurs, dans le cas où l'utilisateur choisirait de commencer en second;
- la fonction `afficher_jeu` permet d'afficher le tableau de jeu courant;
- la fonction `main` simule le jeu; elle demande d'abord à l'utilisateur de faire son choix pour commencer ou non, puis le jeu se réalise tour à tour entre l'utilisateur et l'IA que l'on joue à profondeur 6.

Dans ce TP, vous devrez:

1. implémenter une fonction permettant d'évaluer une position de jeu. Voici une proposition de prototype:

   ```
   int evaluation_position(int joueur, int jeu[6][7], int profondeur);
   ```

   Vous pouvez par exemple vous inspirer de cet article, ou écrire une fonction d'évaluation f telle que

   $$f(p) = \text{Score(joueur1)} - \text{Score(joueur2)}$$

   où le score est calculé selon la règle suivante: on ajoute 1 pour chaque pion du joueur, 5 pour chaque alignement de 2 pions, 50 pour chaque alignement de 3 pions et 1000 pour un alignement de 4 pions.

2. implémenter une IA jouant selon un algorithme `minimax`. Voici une proposition de prototype:

   ```
   int minimax(int jeu[6][7], int profondeur, int joueur)
   ```

   Notez qu'on pourra implémenter deux fonctions permettant de trouver respectivement le maximum des valeurs calculées par le joueur MIN à la profondeur précédente, et le minimum des valeurs calculées par le joueur MAX à la profondeur précédente.

3. implémenter une IA jouant selon un algorithme `alpha-beta`. Voici une proposition de prototype:

   ```
   int alpha-beta(int jeu[6][7], int profondeur, int joueur, int alpha,
                                 int beta);
   ```