ISM, Informatique
Année 2024/2025

Atelier python - Journée pédagogique

Exercice 1

Ouvrir le fichier `nim_minimax.py`.

1. Compléter la fonction `minimax(tas,J_i)` d'arguments `tas` un entier qui est le nombre de bâtonnets d'un jeu de Nim, `J_i` le numéro du joueur dont c'est le tour, qui renvoie la fonction d'évaluation de la configuration.
2. En s'inspirant de la fonction `nim_alea()`, compléter la fonction `nim_minimax()` pour faire s'affronter un joueur J_0 contre la machine J_1, celle-ci jouant en suivant l'algorithme minimax.
3. Défier la machine !

Exercice 2

Ouvrir le fichier `morpion_minimax.py`.

1. Compléter la fonction `minimax(tab,libres,J_i)` d'arguments `tab` une liste décrivant une grille de jeu, `libres` une liste contenant les positions disponibles pour les prochains coups, `J_i` le numéro du joueur dont c'est le tour, qui renvoie la fonction d'évaluation de la configuration.
2. En s'inspirant de la fonction `morpion_alea()`, compléter la fonction `morpion_minimax()` pour faire s'affronter un joueur J_0 contre la machine J_1, celle-ci jouant en suivant l'algorithme minimax.
3. Défier la machine ! (pour les plus motivés, faire s'affronter la machine contre elle-même).

Exercice 3

Prendre connaissance des règles du jeu Othello :
`https://fr.wikipedia.org/wiki/Othello_(jeu)`
puis ouvrir le fichier `othello_minimax.py`.

1. Compléter la fonction `minimax_depth(tab,J_i,p)` d'arguments `tab` une liste décrivant un plateau de jeu, `J_i` le numéro du joueur dont c'est le tour, `p` un niveau de profondeur, qui renvoie une estimation de la fonction d'évaluation de la configuration.
2. En s'inspirant de la fonction `othello_alea()`, compléter la fonction `othello_minimax_depth(p)` d'argument `p` entier pour faire s'affronter un joueur J_0 contre la machine J_1, celle-ci jouant en suivant l'algorithme minimax à un niveau de profondeur `p`.
3. Défier la machine ! Imaginer d'autre heuristiques que celle proposée et les tester.

Exercice 4

Ouvrir le fichier `puiss4_minimax.py`.

1. Compléter la fonction `minimax_depth(tab,col,J_i,p)` d'arguments `tab` une liste décrivant une grille de jeu, `col` une colonne, `J_i` le numéro du joueur dont c'est le tour, `p` un niveau de profondeur, qui renvoie une estimation de la fonction d'évaluation de la configuration.
2. En s'inspirant de la fonction `puiss4_alea()`, compléter la fonction `puiss4_minimax_depth(p)` d'argument `p` entier pour faire s'affronter un joueur J_0 contre la machine J_1, celle-ci jouant en suivant l'algorithme minimax à un niveau de profondeur `p`.
3. Défier la machine ! (pour les plus motivés, faire s'affronter la machine contre elle-même).