

## TP Informatique 12

### Exercice 1

Ouvrir le fichier `nim_alea.py`, analyser le code puis jouer contre la machine. Commenter la stratégie suivie par la machine.

### Exercice 2

La stratégie gagnante au jeu de Nim consiste à laisser à chaque coup, si cela est possible, un tas de  $4k + 1$  bâtonnets avec  $k$  entier à son adversaire.

En s'inspirant du code de `nim_alea.py`, écrire une fonction `nim_expert(N)` d'argument  $N$  entier non nul le nombre initial de bâtonnets d'un jeu de Nim pour faire s'affronter un joueur  $J_0$  contre la machine  $J_1$  jouant de manière experte.

### Exercice 3

Ouvrir le fichier `morpion_alea.py`, analyser le code puis jouer contre la machine. Le joueur  $J_0$  saisira sa position dans la grille de jeu selon le format `x,y` : ligne, colonne.

Par exemple, si le joueur saisit :

J\_0 : x,y =2,1

il obtient la disposition :

		0		1		2
-----						
0						
-----						
1						
-----						
2				0		

Commenter la stratégie suivie par la machine.

### Exercice 4

En s'inspirant du code de `morpion_alea.py`, écrire une fonction `morpion_danger()` pour faire s'affronter un joueur  $J_0$  contre la machine  $J_1$  telle que la machine s'efforce d'éviter un échec sur le coup suivant.

On pourra écrire une fonction auxiliaire `rech_danger(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 liste éventuellement vide des cases où le joueur adverse peut réaliser un alignement horizontal, vertical ou diagonale de trois des ses motifs.

## Exercice 5

En s'inspirant du code de `morpion_danger.py`, écrire une fonction `morpion_victoire()` pour faire s'affronter un joueur  $J_0$  contre la machine  $J_1$  telle que la machine s'efforce de gagner ou sinon s'efforce d'éviter un échec sur le coup suivant.

En plus de la fonction `rech_danger`, on pourra écrire une autre fonction auxiliaire `rech_victoire(tab, libres, J_i)` avec les mêmes arguments que `rech_danger` et qui renvoie la liste éventuellement vide des cases où le joueur  $J_i$  peut réaliser un alignement horizontal, vertical ou diagonale de trois des ses motifs.