Compte-rendu TP2

1.2

```
?- situation_initiale(S), joueur_initial(J).
```

Affiche l'état initial du tic-tac-toe et le nom du joueur qui joue en premier ici les X

```
?- situation_initiale(S), nth1(3,S,Lig), nth1(2,Lig,o)
```

met un O dans la ligne 3 colonne 2.

2.2

Pour vérifier les bons fonctionnement des 3 prédicats, ligne colonne et diagonale, j'ai fait des tests unitaires qui sont comme ceci :

```
test_unitaire_diag:-
    M=[[a,b,c], [d,e,f], [g,h,i]],
    T=[[a,b,c,d], [e,f,g,h], [i,j,k,l],[m,n,o,p]],
    diagonale([a,e,i],M),
    diagonale([c,e,g],M),
    diagonale([a,f,k,p],T),
    diagonale([d,g,j,m],T).

test_unitaire_ligne:-
    M=[[a,b,c], [d,e,f], [g,h,i]],
    T=[[a,b,c,d], [e,f,g,h], [i,j,k,l],[m,n,o,p]],
    ligne([a,b,c],M),
    ligne([d,e,f],M),
    ligne([g,h,i],M),
    ligne([a,b,c,d],T),
    ligne([e,f,g,h],T),
    ligne([i,j,k,l],T),
    ligne([m,n,o,p],T).

test_unitaire_colonne:-
    M=[[a,b,c], [d,e,f], [g,h,i]],
    T=[[a,b,c,d], [e,f,g,h], [i,j,k,l],[m,n,o,p]],
    colonne([a,d,g],M),
    colonne([b,e,h],M),
    colonne([c,f,i],M),
    colonne([a,e,i,m],T),
    colonne([b,f,j,n],T),
    colonne([c,g,k,o],T),
    colonne([d,h,l,p],T).
```

J’ai choisi de tester avec une matrice 3x3 puis avec une matrice 4x4, elles ont été initialisées dans le prédicat de test directement pour un gain de temps en test. Pour les lancer, il suffit de taper « test_unitaire_xxxx. » en remplaçant les xxxx par ligne colonne ou diagonale.

3.2

Le prédicat qui permet de connaître sous forme de liste l’ensemble des couples [Coord, Situation_Resultante] tels que chaque élément (couple) associe le coup d’un joueur et la situation qui en résulte à partir d’une situation donnée est le prédicat successeurs, il est primordial pour l’implémentation de l’algorithme negamax.

4.1

Pmax	1	2	3	4	5	6	7	8	9
Coup	[2, 2]	[2, 2]	[2, 2]	[2, 2]	[2, 2]	[2, 2]	[2, 2]	[1, 1]	Erreur
Gain	4	1	3	1	3	1	2	10000	Erreur

Pour Pmax = 9, on obtient une erreur, **stack limit exceeded**.

4.2

Pour éviter des développement inutiles due à des situations symétriques, il suffit de faire une rotation de la situation actuelle avant de l’évaluer pour vérifier qu’il ne s’agisse pas d’une symétrie d’une situation déjà évaluée

4.3

Pour passer au jeu du puissance 4, il faut changer les prédicats d’alignement pour le faire passer de 3 à 4 éléments. Il faut aussi rajouter la notion d’empilement qui est primordiale pour le puissance 4.

4.4

Pour améliorer l’algorithme en élaguant certains coups inutiles, on va comparer le meilleur coup de la branche que l’on souhaite développer avec le meilleur coup des branches précédemment développées. On ne la développera que si son coup est plus élevé que les autres.