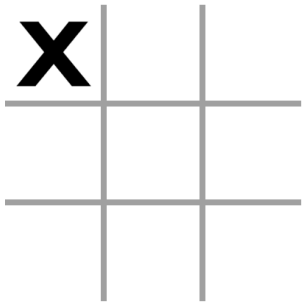
Nom, Prenom:_____ **Groupe:**_____

TP5 : Algorithme MinMax [2 heures]

Préparation

a. Déterminer le nombre de combinaisons possibles au Tic-Tac-Toe.

b. On considére la partie suivante, où faut-il jouer pour ne pas perdre ?



Travail pratique

Exercice 1

La classe Tic-Tac-Toe vous est fournie, acompagnée de sa documentation et d'un exemple d'utilisation.

a. Créer un projet et tester le programme d'exemple.

b. Recopier la fonction AIPlayer et compléter votre programme avec la fonction MinMax afin de sélectionner le meilleur coup avec une profondeur infinie.

```
void  AIPlayer(TicTacToe *Game)
{
    int Move;
    // Recherche le meilleur coup
    MinMax(Game,Game->CurrentPlayer(),&Move);
    // Joue le meilleur coup
    Game->Play(Move);
}
```

```
int MinMax(TicTacToe *Game,char Mark,int *ptBest_Move)
```

c. Modifier le programme de façon à utiliser le formalisme NegaMax.

```
int NegaMax(TicTacToe *Game,char Mark,int *ptBest_Move)
```

Nom, Prenom:_____ **Groupe:**_____

d. Accélerer votre joueur artificiel avec l'élagage Alpha-Béta.

```
int AlphaBeta(TicTacToe *Game,char Mark,int *ptBest_Move,int Alpha, int Beta)
```

e. Les fonctions MinMax, NegaMax et AlphaBeta sont des fonctions récursives. Déterminer la profondeur d'appel lorsque le joueur artificiel joue le premier coup.

MinMax	
NegaMax	
AlphaBéta	

f. Déterminer le poucentage de victoires, de défaites et de matchs nuls pour les configurations suivantes:

			Joueur 1		
Joueur 1	**Joueur 2**		**Victoires**	**Défaites**	**Matchs nuls**
IA	Aléatoire				
Aléatoire	IA				
IA	IA				