```
##Couleurs

VIDE = 0
BLEU = 1
ROUGE = 2



## minimax

#Algorithme permettant, étant donné un plateau, de déterminer, si les deux
joueurs jouent aux mieux, quel joueur est certain de gagner

def minimax (plateau, couleur):

    plein = True

    for x in range (len(plateau[0])):

        for y in range(len(plateau[0])) :
            #on lance une boucle pour toutes les cases du plateau afin de
tester toutes les issues possibles

            if plateau[x][y] == VIDE :

                plateau[x][y] = couleur
                plein = False

                if minimax (plateau, autreCouleur(couleur)) == couleur :

                    plateau[x][y] = VIDE

                    return couleur

                plateau[x][y] = VIDE

    if plein and posGagnante(plateau, couleur):

            return couleur

    return autreCouleur(couleur)

## meilleur premier coup
#Renvoie, pour un plateau T et un couleur 'couleur', un coup permettant, si
le joueur 'couleur' joue de la meilleure façon possible, de gagner à coup
sûr. Sinon, si il n'existe pas de tel coup, renvoie la première case non
vide du plateau.

def coupGagnant(T, couleur):
    for x in range(len(T)):

        for y in range(len(T)):

            if T[x][y] == VIDE :

                T[x][y]=couleur

                if minimax(T,autreCouleur(couleur)) == couleur:

                    T[x][y] = VIDE
```

```
                    return [x,y]

                T[x][y] = VIDE

    for x in range(len(T)) :

        for y in range(len(T)) :

            if T[x][y] == VIDE :
                return [x,y]


## minimax pondéré
#Renvoie un noeud (ici une liste) composé des informations suivantes :
[couleur gagnante, nombre de victoires du joueur bleu, nombre de victoires
du joueur rouge]
#minimaxPond est une fonction récursive : on descend l'arbre de tous les
plateaux possibles, et à chaque feuille, on met à jour le noeud
intermédiaire selon si le joueur a gagné ou non. On fait ensuite remonter
successivement les noeuds jusqu'à arriver à la racine.

def minimaxPond (plateau, couleur):
    plein = True
    valeurNoeud = [autreCouleur(couleur), 0, 0]


    for x in range (len(plateau[0])):

        for y in range(len(plateau[0])) :

            if plateau[x][y] == VIDE :

                plateau[x][y] = couleur
                plein = False

                noeudInt = minimaxPond (plateau, autreCouleur(couleur))

                valeurNoeud[1] += noeudInt[1]
                valeurNoeud[2] += noeudInt[2]

                if noeudInt[0] == couleur :
                    valeurNoeud[0] = couleur

                plateau[x][y] = VIDE

    if posGagnante(plateau, couleur) and plein :

        L = [couleur, 0, 0]
        L[couleur] = 1

        return L

    elif plein :

        L = [autreCouleur(couleur), 0, 0]
        L[autreCouleur(couleur)] = 1

        return L
```

```
    return valeurNoeud

## meilleur coup pondéré

#renvoie le meilleur coup (coup qui donne au joueur le plus de plateaux
gagnants) pour le joueur 'couleur', en se servant du minimax pondéré

def meilleurCoup (plateau, couleur):

    maxNoeudsGagnants = 0
    meilleurePos = premCaseNonVide(plateau)

    for x in range (len(plateau)):

        for y in range (len(plateau)):

            if plateau[x][y] == VIDE :

                plateau[x][y] = couleur
                Noeud = minimaxPond(plateau, autreCouleur(couleur))
                plateau[x][y] = VIDE

                if Noeud[0] == couleur :

                    return [x, y]

                if Noeud[couleur] > maxNoeudsGagnants :

                    maxNoeudsGagnants = Noeud[couleur]
                    meilleurePos = [x,y]

    return meilleurePos
```