ISM MP, Informatique
Année 2024/2025

Corrigé du TP Informatique 12

Exercice 1

La fonction principale est :

```
def nim_alea(N):
    tas,fini,J_i=N,N==1,1
    while not fini:
        [...]
        else:
            # la machine joue (mal)
            retrait=rd.randint(1,min(3,tas-1)+1)
            print("Retire= ",retrait)
        tas-=retrait
        fini=(tas==1)
    print("J_"+str(J_i)+" gagne")
```

On constate que la machine réalise un retrait d'un nombre aléatoire de bâtonnets suivant une loi uniforme dans l'ensemble des retraits possibles avec l'instruction `rd.randint(1,min(3,tas-1)+1)`. La machine n'a donc pas de stratégie et n'importe quel joueur peut la battre aisément.

Exercice 2

On saisit :

```
def nim_expert(N):
    tas,fini,J_i=N,N==1,1
    while not fini:
        J_i=adversaire(J_i)
        print("J_"+str(J_i)+"=",tas)
        if J_i==0:
            valide=False
            while not valide:
                retrait=int(input("Retire="))
                valide=coup_valide(tas,retrait)
                if not valide:
                    print("Coup invalide")
        else:
            r=tas%4
            if r==1:
                retrait=1
            else:
                retrait=(r-1)%4
            print("Retire= ",retrait)
        tas-=retrait
        fini=(tas==1)
    print("J_"+str(J_i)+" gagne")
```

Exercice 3

La fonction principale est :

```
def morpion_alea():
    """Partie de morpion contre la machine
    J_0 : joueur
    J_1 : machine"""
    tab=[[None]*3 for k in range(3)]
    libres=[(i,j) for i in range(3) for j in range(3)]
    nb,fini,J_i=0,False,1
    # J_0 commence (bascule de joueur en début de boucle)
    while not fini:
        J_i=adversaire(J_i)
        print('Coup=',nb)
        nb+=1
        aff(tab)
        if J_i==0:
            [...]
        else:
            ind=rd.randint(len(libres))
            i,j=libres[ind]
        tab[i][j]=J_i
        libres.remove((i,j))
        [...]
    if align_o or align_x:
        print("J_"+str(J_i)+" gagne")
    else:
        print("Match nul")
    aff(tab)
```

La variable `libres` recense, au cours de la partie, les positions disponibles pour les prochains coups. La machine, quand c'est son tour, tire aléatoirement suivant une loi uniforme une position parmi l'ensemble des positions disponibles avec l'instruction `rd.randint(len(libres))`. La machine n'a donc pas de stratégie et il est aisé de la battre.

Exercice 4

On recherche un danger potentiel (s'il y en a plusieurs, on ne peut en éviter qu'un). On saisit :

```
def rech_danger(tab,libres,J_i):
    adverse=adversaire(J_i)
    for (i,j) in libres:
        tab[i][j]=adverse
        test_alignement=alignement(tab,adverse)
        tab[i][j]=None
        if test_alignement:
            return (i,j)
```

puis

```
def morpion_danger():
    tab=[[None]*3 for k in range(3)]
    libres=[(i,j) for i in range(3) for j in range(3)]
    nb,fini,J_i=0,False,1
    while not fini:
        J_i=adversaire(J_i)
        print('Coup=',nb)
        nb+=1
        aff(tab)
        if J_i==0:
            valide=False
            while not valide:
                saisie=input("J_"+str(J_i)+" : x,y =")
                coords=saisie.split(",")
                i,j=int(coords[0]),int(coords[1])
                valide=coup_valide(tab,i,j)
                if not valide:
                    print("Coup invalide")
        else:
            danger=rech_danger(tab,libres,J_i)
            if danger!=None:
                i,j=danger
            else:
                ind=rd.randint(len(libres))
                i,j=libres[ind]
        tab[i][j]=J_i
        libres.remove((i,j))
        align_o=alignement(tab,0)
        align_x=alignement(tab,1)
        fini=align_o or align_x or nb==9
        print()
    if align_o or align_x:
        print("J_"+str(J_i)+" gagne")
    else:
        print("Match nul")
    aff(tab)
```

Exercice 5

On cherche la possibilité d'une victoire. On saisit :

```
def rech_victoire(tab,libres,J_i):
    for (i,j) in libres:
        tab[i][j]=J_i
        test_alignement=alignement(tab,J_i)
        tab[i][j]=None
        if test_alignement:
            return (i,j)
```

puis

```
def morpion_victoire():
    tab=[[None]*3 for k in range(3)]
    libres=[(i,j) for i in range(3) for j in range(3)]
    nb,fini,J_i=0,False,1
    while not fini:
        J_i=adversaire(J_i)
        print('Coup=',nb)
        nb+=1
        aff(tab)
        if J_i==0:
            valide=False
            while not valide:
                saisie=input("J_"+str(J_i)+" : x,y =")
                coords=saisie.split(",")
                i,j=int(coords[0]),int(coords[1])
                valide=coup_valide(tab,i,j)
                if not valide:
                    print("Coup invalide")
        else:
            victoire=rech_victoire(tab,libres,J_i)
            if victoire!=None:
                i,j=victoire
            else:
                danger=rech_danger(tab,libres,J_i)
                if danger!=None:
                    i,j=danger
                else:
                    ind=rd.randint(len(libres))
                    i,j=libres[ind]
        tab[i][j]=J_i
        libres.remove((i,j))
        align_o=alignement(tab,0)
        align_x=alignement(tab,1)
        fini=align_o or align_x or nb==9
        print()
    if align_o or align_x:
        print("J_"+str(J_i)+" gagne")
    else:
        print("Match nul")
    aff(tab)
```