Sommaire :

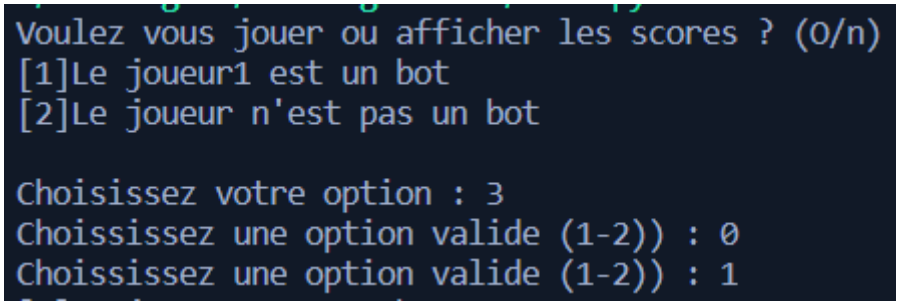
Conception générale :

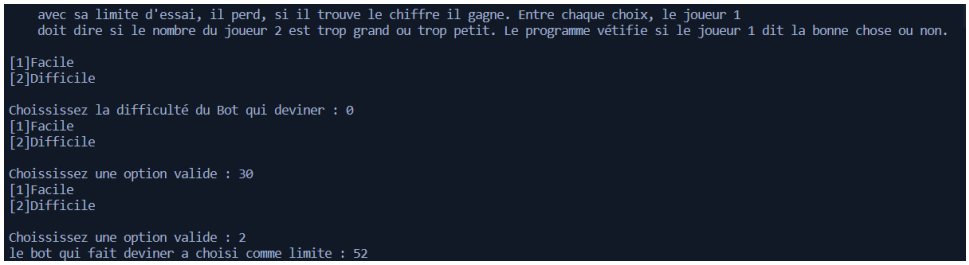
Lors de cette sae, nous avons dû concevoir des bots afin de pouvoir les faire jouer sur notre sae précédemment conçu lors de la S1.01. La consigne étant d'implémenter des bots pouvant suivre différents niveaux de difficultés, nous avons décidé d'implémenter deux difficultés différentes sur chaque jeu, une facile à base de jeu aléatoire et une difficile qui sera dans ce cas très compliqué de gagner voire impossible ! Lors de cette sae la principale difficulté que nous avons rencontrée est la conception du bot difficile du morpion, de savoir comment il serait capable de gérer et identifier les cas de victoires et de défaites possibles. Nous avons donc décidé de vérifier manuellement tous les cas de victoire possible.

Pour l'implémentation de la demande de bot et de difficulté nous avons décidé de demander la difficulté des bots à chaque début de partie afin de pouvoir changer de difficulté entre chaque partie.

```python
def menu_demande_bot(numero_joueur:int) -> str:
    saisie : int
    print(f"[1]Le joueur{numero_joueur} est un bot\n[2]Le joueur n'est pas un bot\n")
    saisie = int(input("Choisissez votre option : "))
    while saisie < 1 or saisie > 2 :
        saisie = int(input("Choisissez une option valide (1-2)) : "))
    if saisie == 1 :
        return "bot"
    elif saisie == 2 :
        return "humain"
    else :
        return ("indisponible")
```

```python
def menu_demande_difficulte(bot : str) -> str:
    saisi : int
    print("[1]Facile\n[2]Difficile\n")
    saisi = int(input(f"Choississez la difficulté du {bot} : "))
    while saisi < 1 or saisi > 2 :
        print("[1]Facile\n[2]Difficile\n")
        saisi = int(input("Choississez une option valide : "))

    if saisi == 1 :
        return "facile"
    elif saisi == 1 :
        return "difficile"
    else :
        return "indisponible"
```


Voulez vous jouer ou afficher les scores ? (O/n)
[1]Le joueur1 est un bot
[2]Le joueur n'est pas un bot
Choisissez votre option : 3
Choississez une option valide (1-2)) : 0
Choississez une option valide (1-2)) : 1


avec sa limite d'essai, il perd, si il trouve le chiffre il gagne. Entre chaque choix, le joueur 1
doit dire si le nombre du joueur 2 est trop grand ou trop petit. Le programme vétifie si le joueur 1 dit la bonne chose ou non.
[1]Facile
[2]Difficile
Choississez la difficulté du Bot qui deviner : 0
[1]Facile
[2]Difficile
Choississez une option valide : 30
[1]Facile
[2]Difficile
Choississez une option valide : 2
le bot qui fait deviner a choisi comme limite : 52

Devinettes :

Dans ce jeu, 2 bots peuvent jouer l'un contre l'autre, mais seulement le bot qui devine à 2 niveaux de difficultés car une difficulté plus élever sur le choit de la limite du nombre à deviner, le nombre à deviner et la limite d'essai serai inutile. La difficulté facile du bot qui devine va prendre un chiffre aléatoire entre 0 et la limite, puis prendra des nombres supérieurs si le nombre choisit et inférieur au nombre à deviner, et inferieur dans le cas inverse. La difficulté difficile elle utilise une méthode dichotomique pour trouver le nombre, c'est à dire qu'il va choisir la moitié de l'intervalle, puis si le nombre a deviné est plus grand, la borne inferieur devient ce nombre et dans le cas inverse c'est la borne supérieure qui devient ce nombre.

On peut également faire un bot contre humain ou les difficultés restes les mêmes. De plus, il est possible de choisir qui devine et qui fait deviner entre le bot et l'humain qui joue, si 2 bots jouent entre eux, ce sera choisie aléatoirement.

Si on fait deviner le bot en version facile, sur 5 parties en partant sur un intervalle de 1 à 50 et sur 6 essais, le bot 3 parties, en mode difficile avec les mêmes valeurs de test il gagne toutes les parties car 6 essais pour lui seront forcément nécessaire en utilisant la dichotomie

Jeux de tests :

Bot contre bot:

```
Devinette en cours de lancement...

Vous allez devoir choisir un joueur qui fera deviner un nombre, il choisira l'intervalle de choix, le nombre à deviner
    et le nombre d'essais auquel le joueur 2 à le droit. Si le joueur 2 ne trouve pas le nombre
    avec sa limite d'essai, il perd, si il trouve le chiffre il gagne. Entre chaque choix, le joueur 1
    doit dire si le nombre du joueur 2 est trop grand ou trop petit. Le programme vétifie si le joueur 1 dit la bonne chose ou non.

[1]Facile
[2]Difficile

Choississez la difficulté du Bot qui devine :
```

Demande de la difficulté (seulement du bot qui devine)

```
le bot qui fait deviner a choisi comme limite : 32
Il a également choisi comme limite d'essai : 5
nb a deviner :  13
```

On voit que le bot peut en effet choisir les 3 variables nécessaire pour jouer au jeu. Elles sont affichées afin de pouvoir analyser les données

```
Choississez la difficulté du Bot qui devine : 1
le bot qui fait deviner a choisi comme limite : 84
Il a également choisi comme limite d'essai : 8
nb a deviner :  38



Le bot qui devine joue et choisi 63 .
C'est plus petit !
Le bot qui devine joue et choisi 37 .
C'est plus grand !
Le bot qui devine joue et choisi 50 .
C'est plus petit !
Le bot qui devine joue et choisi 38 .
Le bot qui devine a gagné !
```

Ici, le bot qui devine a gagné avec la difficulté en facile, et choisit bien un nombre plus petit si c'est plus petit, et un nombre plus grand si c'est plus grand.

```
le bot qui fait deviner a choisi comme limite : 25
Il a également choisi comme limite d'essai : 7
nb a deviner :  11



Le bot qui devine joue et choisi 12 .
C'est plus petit !
C'est donc plus petit bot !
Le bot qui devine joue et choisi 6 .
C'est plus grand !
C'est donc plus grand bot !
Le bot qui devine joue et choisi 9 .
C'est plus grand !
C'est donc plus grand bot !
Le bot qui devine joue et choisi 10 .
C'est plus grand !
C'est donc plus grand bot !
Le bot qui devine joue et choisi 11 .
Le bot qui devine a gagné !
```

Ici, la difficulté est difficile, donc on voit qu'il choisit bien la moitié de l'intervalle [0 ;25] (12 ici) et la méthode dichotomique fonctionne bien.

Humain vs bot :

```
clement doit choisir une limite pour l'intervalle de valeur ( de 0 a cette limite ) : 100
clement doit choisir le nombre compris entre 0 et 100 : 50
Veuillez également choisir une limite d'essais : 5



Le bot qui devine joue et choisi 96 .
clement , le nombre est [1]plus grand ou [2]plus petit ? 2
Le bot qui devine joue et choisi 70 .
clement , le nombre est [1]plus grand ou [2]plus petit ? 2
Le bot qui devine joue et choisi 4 .
clement , le nombre est [1]plus grand ou [2]plus petit ? 1
Le bot qui devine joue et choisi 39 .
clement , le nombre est [1]plus grand ou [2]plus petit ? 1
Le bot qui devine joue et choisi 59 .
clement , le nombre est [1]plus grand ou [2]plus petit ? 2
clement a gagné !
```

Ici, la difficulté est de 1 pour le bot, et quand un humain dit plus grand il choisit bien plus grand, plus petit si l'humain dit plus petit

```
clement doit choisir une limite pour l'intervalle de valeur ( de 0 a cette limite ) : 100
clement doit choisir le nombre compris entre 0 et 100 : 51
Veuillez également choisir une limite d'essais : 7



Le bot qui devine joue et choisi 50 .
clement , le nombre est [1]plus grand ou [2]plus petit ? 1
C'est donc plus grand bot !
Le bot qui devine joue et choisi 75 .
clement , le nombre est [1]plus grand ou [2]plus petit ? 2
C'est donc plus petit bot !
Le bot qui devine joue et choisi 62 .
clement , le nombre est [1]plus grand ou [2]plus petit ? 2
C'est donc plus petit bot !
Le bot qui devine joue et choisi 56 .
clement , le nombre est [1]plus grand ou [2]plus petit ? 2
C'est donc plus petit bot !
Le bot qui devine joue et choisi 53 .
clement , le nombre est [1]plus grand ou [2]plus petit ? 2
C'est donc plus petit bot !
Le bot qui devine joue et choisi 51 .
```

Comme précédemment, le bot fonctionne bien suivant l'entrée de l'humain, et on peut voir avec cet exemple que le bot a besoin de 7 essais maximum pour gagner à coup sûr quand l'intervalle est compris entre 1 et 100.

Allumettes :

Conception du bot : Pour ces bots, nous avons tout d'abord fait choisir un chiffre aléatoire entre 1 et 3 au bot, un chiffre entre 1 et 2 quand il reste que 2 allumettes et 1 si le nombre d'allumettes est de 1. Ensuite, pour la version difficile nous avons utilisé une stratégie plus complexe ou si le bot parvient à avoir un nombre fatal en 1er, il a gagné, mais la règle s'applique également pour le joueur humain. Or, le joueur qui joue en premier peut tomber sur un nombre fatal très simplement en enlevant 3 allumettes, donc si le bot joue en 1er il a gagné. Ensuite, il suffit de jouer pour tomber sur un nombre fatal (multiple de 4 + 1), puis quand on y est arrivé, il suffit de jouer 4 – le nombre d'allumettes joué par l'autre joueur.

Jeux de tests :

```
[1]Bot 1
[2]clement
Choississez qui jouer en premier : 1
[1]Facile
[2]Difficile

Choississez la difficulté du bot 1 : 2
```

```
clement, à vous de jouer
[1]Retirer une allumette
[2]Retirer deux allumettes
[3]Retirer 3 allumettes
Quel est votre choix ? : 1
Le bot 1 a gagné
```

Nous voyons avec cet exemple que si le bot commence en premier, il gagne bel et bien.

```
[1]Bot 1
[2]clement
Choississez qui jouer en premier : 2
[1]Facile
[2]Difficile

Choississez la difficulté du bot 2 : 2
```

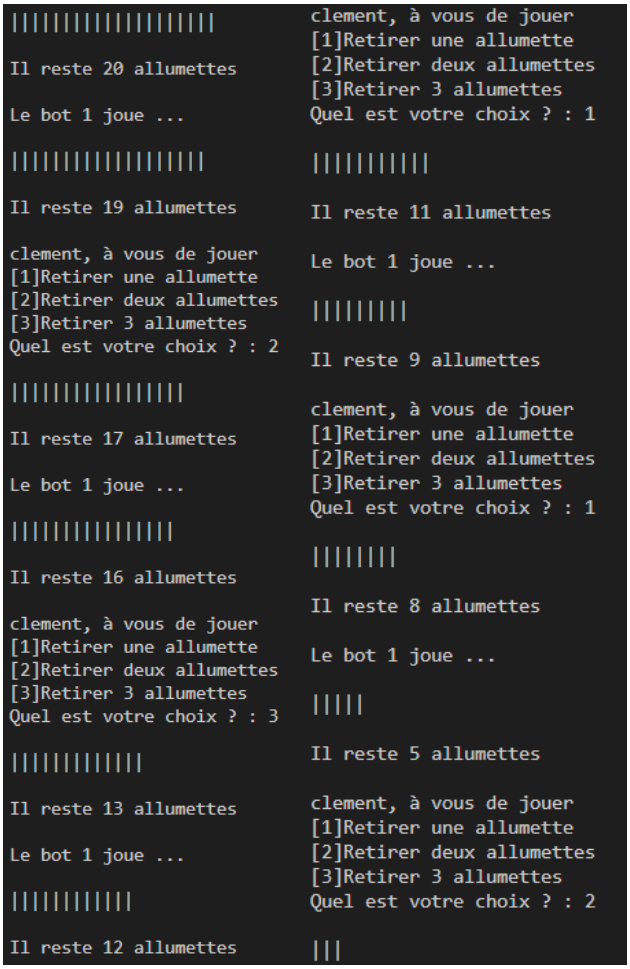
```
le bot 2 joue ...
Il ne reste aucunes allumettes, bravo clement, vous avez gagné
```

Dans cet exemple, on voit qui si le joueur joue en 1er, il peut gagner.

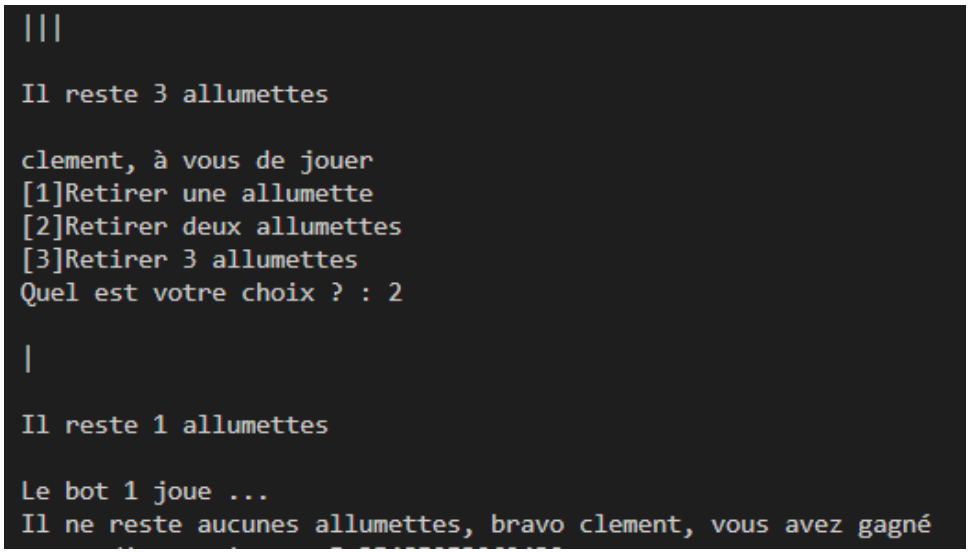
```
||||||||||||||||||||

Il reste 20 allumettes

Le bot 1 joue ...

||||||||||||||||

Il reste 17 allumettes

clement, à vous de jouer
[1]Retirer une allumette
[2]Retirer deux allumettes
[3]Retirer 3 allumettes
Quel est votre choix ? : 1

|||||||||||||||

Il reste 16 allumettes

Le bot 1 joue ...

|||||||||||||
```

```
Il reste 13 allumettes

clement, à vous de jouer
[1]Retirer une allumette
[2]Retirer deux allumettes
[3]Retirer 3 allumettes
Quel est votre choix ? : 3

|||||||||

Il reste 10 allumettes

Le bot 1 joue ...

||||||||

Il reste 9 allumettes

clement, à vous de jouer
[1]Retirer une allumette
[2]Retirer deux allumettes
[3]Retirer 3 allumettes
Quel est votre choix ? : 1

|||||||

Il reste 8 allumettes

Le bot 1 joue ...

|||||

Il reste 5 allumettes
```

On voit que le bot difficile retombe toujours sur un nombre fatal.



On voit que si le bot est en facile, il choisit bien des nombres aléatoires.



S’il reste qu’une allumette, le bot retire qu'une allumette.

Morpion :

Conception du bot : Pour implémenter le bot facile du morpion dans notre programme, nous avons décidé d'utiliser des randint, le bot ne jouerai qu'exclusivement avec des randint. Pour le joueur sur 10 parties il gagne 7 parties sur 10 environ. Pour le bot difficile, il est impossible de gagner, on peut juste faire des égalités ou perdre, ce qui fait que sur 10 parties le joueur perd 4 de parties. Le bot difficile quant à lui suit pour les 3 premiers tours un schéma très précis et répétitif. Après cela il vérifie à chaque fois les cas de victoires possibles que ce soit pour lui ou pour l'adversaire, s'il ne trouve aucun cas de victoire possible, il joue aléatoirement.

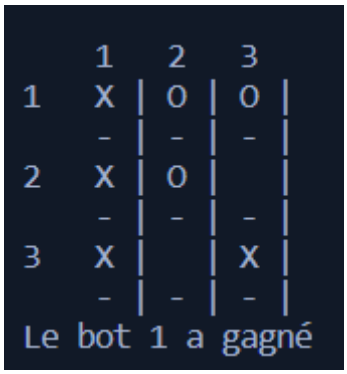
Le bot a été implémenté de tel manière qu'il ne puisse pas utiliser la fiabilisation utilisée pour les joueurs et qu'il ait sa propre fiabilisation afin d'éviter les répétitions.

```python
def bot_morpion1(tab: list[list[str]]) -> list[int]:
    """
    Cette fonction est la fonction qui permet au bot de jouer au morpion de façon "naïve", il ne joue qu'avec des randoms
    Args:
        tab (list[list[str]]): Tableau de jeu

    Returns:
        list[int]: l'indice 0 de la liste correspond à la ligne et l'indice 1 à la colonne
    """
    ligne: int
    ligne = morpion_ligne_colonne()
    colonne = morpion_ligne_colonne()
    while case_prise_morpion(tab, ligne-1, colonne-1): #Fiabilisation
        ligne = morpion_ligne_colonne()
        colonne = morpion_ligne_colonne()
    res = [ligne, colonne]
    return res
```

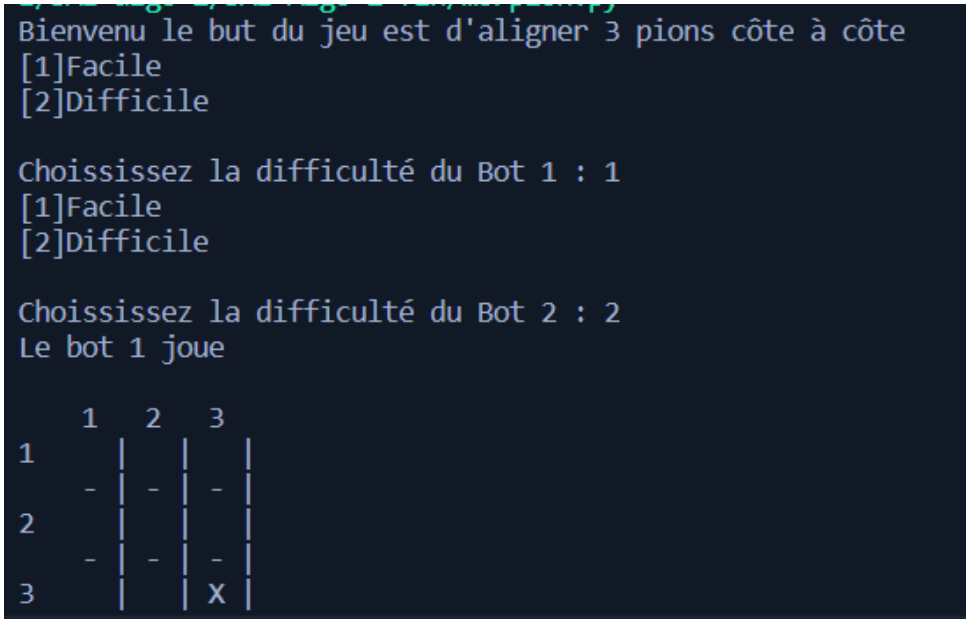
Cette fonction est utilisée par le bot difficile lorsqu'il ne trouve aucun cas de victoire ou de défaite possible, c'est donc là qu'on utilise la fiabilisation énoncée précédemment.

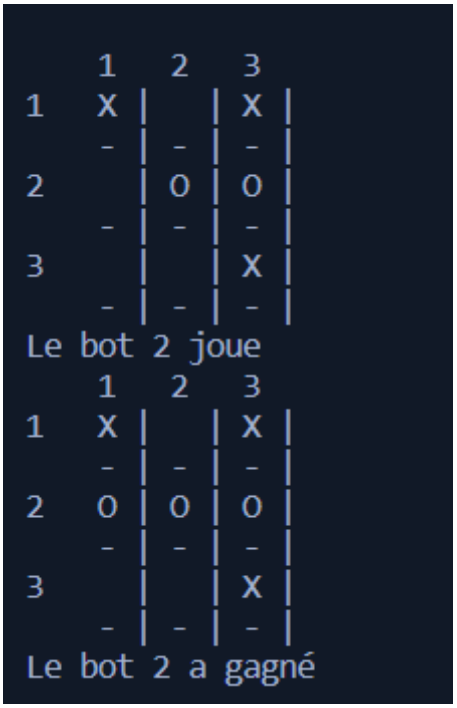
Jeux de tests :

Le résultat d'une partie de bot facile contre bot facile est aléatoire



Lors d'une partie entre un bot facile et un bot difficile, le bot difficile a de grandes chances de gagner





En effet, il n'a fallu que 5 tours au bot difficile pour gagner

Puissance 4 :

Pour le bot du puissance 4, nous avons eu le temps de faire le bot facile mais pas le bot difficile...

Le bot facile est conçu de tel manière à ce qu'il joue complètement aléatoirement tout en conservant sa propre fiabilisation comme dans les autres jeux.

L'issus d'une partie de bot contre joueur est donc presque certaine, en effet le joueur gagne très souvent, une partie de bot contre bot quant à elle est complètement aléatoire.

Comparaison des stratégies :

Pour comparer les différentes stratégies, nous avons décidé de comparer les temps d'exécution de chaque jeu en utilisant 2 bots l'un contre l'autre. Nous les avons fait jouer en facile contre facile, et difficile contre difficile.

Allumette :

Pour les bots faciles contre facile, nous avons trouvé environ 0.008978 seconde de temps d'exécution contre 0.015954 seconde pour le mode difficile contre difficile. L'algorithme aléatoire est donc plus rapide car il n'y a pas de stratégies, contrairement au bot difficile qui lui met en place une stratégie qui enlève souvent une seule allumette ce qui rend l'algorithme plus long.

Devinette :

Pour ce jeu, nous trouvons 0.001994 et 0.011966 seconde pour le mode facile, et 0.002986 seconde pour le mode difficile. L'algorithme aléatoire peut être plus ou moins rapide puisqu'il peut trouver en 1 ou 2 coups le bon nombre comme pas le trouver du tout. L'algorithme difficile quant à lui, sera toujours rapide puisque la méthode dichotomique est optimisée.

Morpion :

Le morpion en facile met entre 0.013962 et 0.026924 seconde pour s'exécuter, et 0.012975 seconde pour la difficulté difficile. Comme pour les devinettes, l'algorithme aléatoire peut gagner très rapidement ou pas du tout, mais l'algorithme difficile lui et globalement assez rapide, au dépend de la difficulté de l'autre bot. En difficile contre facile, on a 0.010965 et 0.024927 seconde, donc le temps d'exécution dépend de comment le bot facile va gêner le bot difficile.

Expérience utilisateur :

Allumettes :

Avec l'algo du bot difficile, il est compliqué pour un utilisateur de gagné à moins qu'il ne connaisse la stratégie pour gagner coup sûr. S'il la connait, il est également sûr de gagner.

Devinette :

Il est possible de gagner contre le bot qui fait deviner, mais très difficile de gagner contre le bot qui devine puisqu'en difficulté difficile si l'utilisateur met une limite d'essai trop importante, le bot est sûr de gagner.

Morpion :

Contre le bot difficile du morpion il est impossible de gagner, seulement faire des égalités puisqu'il peut se défendre et met en place une stratégie pour gagner. Le bot facile lui peut être très embêtant comme pas du tout

Pour prendre en compte l'expérience utilisateur, il faudrait expliquer les stratégies utilisées par les bots difficile afin qu'il puisse gagner contre certain ou au moins se défendre contre d'autre. Avec ces jeux, il est compliqué de prendre en compte l'expérience utilisateur puisque on a utilisé des algorithmes qui sont fait pour gagner à coup sûr ou ne jamais perdre, donc il faut connaitre les différentes stratégies pour avoir une bonne expérience.