

Asie – 2026 – sujet2 - Correction

Exercice 1 (6 points)

1. Indice de la case numéro 2

tab = [5, 3, 8, 0, 1, 2, 7, 6, 4] : la valeur 2 est à l'**indice 5**.

2. Grille après les deux instructions

tab[3] = tab[4] copie le 1 à l'indice 3, puis tab[4] = 0 place la case vide à l'indice 4 : tab vaut [5, 3, 8, 1, 0, 2, 7, 6, 4]. Le 1 a glissé vers la gauche :

5	3	8
1		2
7	6	4

3. Position gagnante

tab == [0, 1, 2, 3, 4, 5, 6, 7, 8]

👉 *Commentaire : en Python, == compare deux listes élément par élément.*

4. Méthode est_gagnant

```
def est_gagnant(self):  
    return self.tab == [0, 1, 2, 3, 4, 5, 6, 7, 8]
```

5. Méthode indice (lignes 3 et 5)

```
def indice(self, numero):  
    assert type(numero) == int, 'numero doit être entier'  
    assert 0 <= numero <= 8, 'numero de case non valide'           # ligne 3  
    i = 0  
    while self.tab[i] != numero:                                     # ligne 5  
        i = i + 1  
    return i
```

👉 *Commentaire : grâce à l'assertion de la ligne 3, le numéro est forcément présent dans tab : la boucle s'arrête avant de sortir du tableau. On accepte 0, puisque la méthode jouer cherche l'indice de la case vide avec self.indice(0).*

6. Méthode jouer (lignes 2 à 6)

```
def jouer(self, numero):  
    if self.est_possible(numero):  
        i = self.indice(numero)  
        j = self.indice(0)  
        self.tab[j] = numero  
        self.tab[i] = 0
```

👉 *Commentaire : la case numérotée prend la place de la case vide (indice j), et la case vide prend la sienne (indice i).*

7. Méthode melanger (lignes 4, 5, 6 et 9)

```
def melanger(self, n):  
    precedent = None  
    i = 0  
    while i < n:                                     # ligne 4  
        possibilites = self.coups_possibles()        # ligne 5  
        choix = random.choice(possibilites)          # ligne 6  
        if choix != precedent:  
            self.jouer(choix)  
            precedent = choix                         # ligne 9  
            i = i + 1
```

👉 *Commentaire : i ne compte que les coups réellement joués : si le hasard redonne le numéro précédent, on recommence le tirage.*

8. Résolution automatique de l'exemple

Après le mélange (1, 4, 5) et le coup du joueur (2), la pile contient, de la base au sommet : [1, 4, 5, 2]. On dépile et on joue chaque coup :

coup joué	pile après le coup (base → sommet)
2	[1, 4, 5]
5	[1, 4]
4	[1]
1	[] : le taquin est gagnant

Les numéros joués sont donc **2, 5, 4, 1** : les coups dans l'ordre inverse.

👉 *Commentaire : rejouer un numéro annule le coup précédent, puisque la case revient à sa place : on défait les coups du dernier au premier.*

9. Méthode resoudre

```
def resoudre(self):  
    self.mode_resolution = True  
    while not self.est_gagnant():  
        coup = self.pile.depiler()  
        self.jouer(coup)  
        print(coup)
```

10. Oubli du mode résolution

Sans passer en mode résolution, la méthode jouer **réempile** chaque coup qu'on vient de dépiler. On dépile x, on le joue, on réempile x ; au tour suivant on redépile x et on le rejoue, ce qui annule le coup précédent. La pile ne se vide jamais et le taquin oscille entre deux positions : la boucle while **ne se termine pas**.

11. Modification de jouer à partir de la ligne 8

```
if not self.mode_resolution:                                # ligne 7  
    if not self.pile.est_vide():  
        sommet = self.pile.depiler()  
        if sommet != numero:  
            self.pile.empiler(sommet)  
            self.pile.empiler(numero)  
    else:  
        self.pile.empiler(numero)
```

👉 *Commentaire : si le numéro joué est égal au sommet, le coup annule le précédent : on retire le sommet et on n'empile rien. Sinon, on remet le sommet puis on empile le nouveau coup par-dessus.*