Exercice 1: Une grille de Takuzu, ou Binairo, est une grille carrée à n lignes et autant de colonnes avec n pair. Au départ, chaque case peut contenir 0, 1 ou être vide. En Python, une grille est codée par une liste de listes, les cases vides étant représentées par None. Par exemple, la grille

0	1		0
1		0	1
0	0	1	1
1	1	0	

est codée par la liste `[[0,1, None, 0 ],[1, None, 0, 1] , [0, 0, 1, 1] , [1, 1, 0, None ]]`. Dans la suite, on identifie la grille et la liste qui la représente.

1. Écrire une fonction prenant en argument une grille de taille non précisée et renvoyant le nombre de cases non remplies.

2. Écrire une fonction prenant en argument une grille de taille non précisée et renvoyant un booléen testant s'il existe deux lignes identiques.

Exercice 2: Soit $p \in]0,1[$.

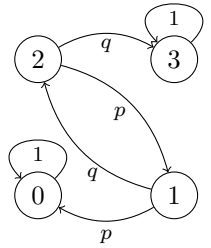
Une puce peut occuper quatre positions numérotées de 0 à 3 . À chaque instant $t = n$, où $n \in \mathbb{N}$, elle se déplace (ou pas). On suppose que :

— 0 et 3 sont des puits : si la puce s'y trouve, elle y reste définitivement ;

— si la puce est en 1 , alors elle saute à la position 0 à l'instant suivant avec la probabilité p ou à la position 2 avec la probabilité $q = 1 - p$;

— si la puce est en 2, alors elle saute en 1 avec la probabilité p ou en 3 avec la probabilité q.

À $t = 0$, la puce se trouve en 1. Les conditions de déplacement peuvent être représentées schématiquement comme suit :



On suppose que la bibliothèque random a été importée avec l'instruction `import random`. On rappelle que pour a un nombre entre 0 et 1 , l'expression `random.random()<a` vaut `True` avec une probabilité a.

1. Écrire un programme `saut` prenant en argument le paramètre p et la position de la puce à l'instant n et renvoyant la position de la puce à l'instant suivant $n + 1$.

2. Écrire un programme `position` prenant en argument le paramètre p et un entier naturel n non nul et renvoyant la liste des n premières positions de la puce.

FE 19 - Exercices sans préparation de la feuille 18 Solutions

Exercice 1

1.

```
def non_remplies(Grille):
    '''compte le nombre de cases non remplies dans la grille.'''
    nb=0
    for i in range(len(Grille)):
        for x in Grille[i]:
            if x==None:
                nb+=1
    return(nb)
```

2.

```
def ligne_identiques(Grille):
    '''Renvoie s'il existe deux lignes identiques.
    '''
    n=len(Grille)
    resu=False
    for i in range(n):
        for j in range(n):
            if i!=j and Grille[i]==Grille[j]:
                resu=True
    return(resu)
```

Exercice 2

1.

```
def saut(p,position):
    '''rend la position suivante de la puce.'''
    if position==0 or position==3:
        return(position)
    else:
        if random.random()<p:
            return(position-1)
        else:
            return(position+1)
```

2.

```
def position(p,n):
    '''rend la liste des n premières positions de la puce.'''
    Liste_pos=[1] # point de départ : 1
    for _ in range(n-1): # on fait n-1 déplacements pour avoir n positions au final
        Liste_pos+=[saut(p,Liste_pos[-1])]
    return(Liste_pos)
```