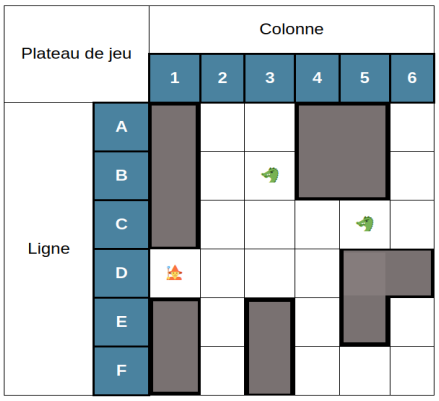
Données 2 - MATRICE

I – Qu'est-ce qu'une matrice

MATH : une matrice basique est un tableau à plusieurs dimensions.

INFO : **une matrice est un tableau de tableaux dont les sous-tableaux possèdent tous le même nombre de cases.**



Plateau de jeu	Colonne					
Ligne	0	1	2	3	4	5
0	1	0	0	1	1	0
1	1	0	3	1	1	0
2	1	0	0	0	3	0
3	8	0	0	0	1	1
4	1	0	1	0	1	0
5	1	0	1	0	0	0

Plateau de jeu	Colonne					
Ligne	0	1	2	3	4	5
0	'X'	'.'	'.'	'X'	'X'	'.'
1	'X'	'.'	'm'	'X'	'X'	'.'
2	'X'	'.'	'.'	'.'	'm'	'.'
3	'o'	'.'	'.'	'.'	'X'	'X'
4	'X'	'.'	'X'	'.'	'X'	'.'
5	'X'	'.'	'X'	'.'	'.'	'.'

On encode les informations de la matrice comme on le désire : souvent via des nombres ou des caractères.

II – Rappels sur les tableaux

*Il faut **impérativement savoir** comment réaliser les opérations suivantes :*

→ Déclarer un tableau

→ Récupérer sa taille

→ Accéder à l'un de ses éléments
→ Lire les éléments un par un via une boucle

Notation t → L'adresse du tableau
Notation i → Un indice : un numéro de case.
Notation t[i] → Le contenu de la case i du tableau.

III – Création d'une matrice en Python

3.1 Exemple et syntaxe

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

Lorsqu'on évalue **`matrice[2]`**, on récupère l'élément d'indice 2 contenu dans le tableau matrice. Il s'agit donc d'une ligne (celle surlignée) : `[1, 0, 0, 0, 3, 0]` .

Lorsqu'on évalue **`matrice[2][4]`**, on récupère l'élément d'indice 4 contenu dans le sous-tableau 2. Il s'agit donc du contenu dans cette case (celle en gras).

3.2 Accès aux cases :

`matrice[LIGNE][COLONNE]`

Pour accéder à une case précise dans une matrice, il suffit donc de fournir d'abord son numéro de ligne puis son numéro de colonne.

IV – Lecture et affichage

4.1 Principe

A l'aide de deux boucles imbriquées, on affiche une à une les cases mais on rajoute une instruction ligne 14 : à chaque fois qu'on a totalement fini l'une des boucles internes, on passe à la ligne puisqu'on vient d'afficher toutes les colonnes d'une ligne.
`len(matrice)` renvoie le nombre de sous-tableaux et donc le nombre de lignes dans la matrice.
`len(matrice[0])` renvoie le nombre de colonnes dans la matrice puisque tous les sous-tableaux ont la même taille.

4.2 Exemple avec parcours par indice (modification possible)

```
matrice = [
    [1, 0, 0, 1, 1, 0],
    [1, 0, 3, 1, 1, 0],
    [1, 0, 0, 0, 3, 0],
    [8, 0, 0, 0, 1, 1],
    [1, 0, 1, 0, 1, 0],
    [1, 0, 1, 0, 0, 0]
]

for y in range(len(matrice)):          # Pour chaque indice y de ligne possible
    for x in range(len(matrice[0])):   # Pour chaque indice de colonne x possible
        print(matrice[y][x], end=" ")
    print()
```

4.3 Exemple avec parcours par valeur (modification impossible)

```
matrice = [
    [1, 0, 0, 1, 1, 0],
    [1, 0, 3, 1, 1, 0],
    [1, 0, 0, 0, 3, 0],
    [8, 0, 0, 0, 1, 1],
    [1, 0, 1, 0, 1, 0],
    [1, 0, 1, 0, 0, 0]
]

for ligne in matrice:              # Pour chaque tableau-ligne possible
    for valeur in ligne:           # Pour chaque valeur sur cette ligne
        print(valeur, end=" ")
    print()
```

4.4 ASCII et UNICODE

ASCII est la table de conversion standard entre caractère-valeur. Le nombre est encodé dans UN SEUL OCTET. Néanmoins, la norme ASCII n'impose que les associations caractère-valeur de 0 à 127. Les valeurs de 128 à 255 ne sont donc pas fixées et les caractères représentés varient d'une tableau à une autre.

UNICODE est une association caractère-valeur qui comporte tous les glyphes et caractères que l'humanité utilise. UNICODE ne préoccupe pas de savoir comment on encode concrètement ce nombre en mémoire. L'encodage concrèt le plus courant d'UNICODE est l'**encodage UTF-8**.

4.5 Fonction native `ord()`

En Python, on peut trouver la valeur UNICODE d'un caractère en utilisant la fonction native **`ord()`**.

4.6 Fonction native `chr()`

C'est l'inverse : on lui donne un caractère et elle renvoie la valeur UNICODE correspondante.

Ainsi, lorsque Python doit classer des mots, il place les mots commençant par ç après les mots en z puisque la valeur unicode du ç est supérieure à celle du z !

```
>>> ord('A')
65

>>> ord('B')
66

>>> ord('a')
97

>>> ord('z')
122

>>> ord('ç')
231
```

```
>>> chr(65)
'A'

>>> chr(231)
'ç'

>>> chr(9821)
'♝'

>>> chr(9822)
'♞'

>>> chr(9823)
'♟'
```