DS T nsi LISTES : Nom Prénom :

1. Exploitation d'une liste /4

Soit la liste suivante :

```
1 ma_liste = ["un","deux",3,"quatre",5,6,7,8,9,"dix"]
```

Compléter le tableau dans l'ordre d'arrivée des lignes de codes

Code	Résultat
print(ma_liste[2])	
print(ma_liste[7:])	
print(ma_liste[2:7])	
print(ma_liste[3][0])	
print(ma_liste[-1])	
print(ma_liste.pop())	
print(ma_liste)	
print(ma_liste.pop(0))	

2. Listes par compréhension /3

a. Que va afficher ?

```
liste_1 = [ i for i in range (10)   if   i%3 != 0 ]
print(liste_1)
```

b. Ecrivez le code qui permet de créer la liste :

[[0, 1, 2, 3, 4, 5, 6, 7, 8, 9],

[10, 11, 12, 13, 14, 15, 16, 17, 18, 19],

[20, 21, 22, 23, 24, 25, 26, 27, 28, 29],

[30, 31, 32, 33, 34, 35, 36, 37, 38, 39],

[40, 41, 42, 43, 44, 45, 46, 47, 48, 49]]

DS T nsi LISTES : Nom Prénom :

3. Fonction sur les listes : comptage du nombre d'occurrence /3

Par indice ou par valeur, écrivez le code qui permette d'obtenir :

```
phrase = ['L','a',' ','p','e','t','i','t','e','','m','a','i','s','o','n','
',','d','a','n','s',' ','l','a',' ','p','r','a','i','r','i','e' ]

def compter(                ):
    
print(compter(phrase,'a'))
print(compter(phrase,'y'))
```

Retourne

```
5
None
```

None est le nom d'une variable qui est vide : on peut écrire « return None » à la fin d'une fonction. Cela signifie qu'elle ne renvoie rien. C'est la technique employée de préférence en Python. En C/C++ on retourne généralement « -1 » quand on désire montrer que la fonction n'a rien à renvoyer.

PS on aurait pu aussi écrire une chaine de caractères à la place de la liste, son traitement est identique. En revanche une chaine de caractères ne peut pas être modifiée on ne peut faire que des concaténations.