TD Récursivité :

1. Application du cours

1. 1. Fonction somme :

1. Combien d'appel de fonction sont nécessaire pour somme(5) ? somme(n) ?

1. 2. Fonction factorielle :

1. Ecrire le code de la fonction *factorielle*(n) étant défini comme :

$$factorielle(n) = \begin{cases} 1 \; si \; n = 0 \\ n * factorielle(n-1) \; sinon. \end{cases}$$

2. Quels seront les appels effectués pour obtenir factorielle(4) ?

2. TD

2. 1. Fonction mystère :

1. Que fait la fonction mystère ci-dessous :

```
def mystere(i,k):
    if i<=k :
        print(i)
        boucle(i+1,k)
```

2. 2. Nombre de chiffre d'un nombre :

Ecrire une fonction nb_chiffre(n) permettant d'obtenir le nombre de chiffre d'un nombre :

```
>>> nb_chiffre(9)
1
>>> nb_chiffre(99)
2
```

2. 3. Maximum d'un tableau :

Ecrire une fonction maximum(t) permettant d'obtenir le nombre le plus grand d'un tableau :

> Tips :
>
> - Utilisez la fonction max(a,b)
> - Les slices
>   - t = [0,1,2]
>   - t[2:] => [2]

3. Bonus :

3. 1. Suite de Syracuse :

La suite de syracuse est une suite définie comme :

$$U_{n+1} = \begin{cases} U_n/2 \ si \ U_n \ est \ pair \\ 3 * U_n + 1 \ sinon. \end{cases}$$

Sachant que U_0 est supérieur à 1

1. Ecrire la fonction *syracuse(u)* affichant les valeurs de la suite de syracuse.
   La suite s'arrête lorsque *u* est inférieur ou égal à 1