

**U.E. ALGO3****TP3****Rékursivité**

**Q 1 .** Écrire une fonction itérative puis récursive qui permet de calculer

$$\sum_{i=1}^n \frac{1}{i^2}$$

```
>>> somme_inverse_carrees(2)
>>> 1.25
```

**Q 2 .** Soit la suite suivante. Écrire une fonction itérative puis récursive qui permet de renvoyer la liste des  $n$  premiers termes de la suite  $u_n$ .

$$\begin{cases} u_n = 3 \times u_{n-1} + 1 & \text{si } n > 0 \\ u_0 = 1 \end{cases}$$

```
>>> n_premiers_termes_suite(3)
>>> [1, 4, 13]
```

**Q 3 .** Écrire une fonction itérative puis récursive permettant de calculer les  $n$ ème termes des suites  $u_n$  et  $v_n$  définis comme suit.

$$\begin{cases} u_n = \frac{u_{n-1} + v_{n-1}}{2} & \text{si } n > 0 \\ u_0 = 4 \end{cases} \quad \begin{cases} v_n = (u_{n-1} \times v_{n-1}) & \text{si } n > 0 \\ v_0 = 5 \end{cases}$$

**Q 4 .** Soit  $A$  une matrice carrée de taille  $n$ , de coefficients  $a_{ij}$ . Écrire une fonction récursive qui permet de calculer le déterminant de  $A$ . On rappelle que le déterminant de  $A$  est donné par la formule de récurrence

$$\det(A) = \begin{cases} a_{11} & \text{si } n = 1 \\ \sum_{i=1}^n (-1)^i \times a_{i1} \times \det(A_{i1}) & \text{sinon} \end{cases}$$

Ici  $A_{ij}$  désigne la matrice  $A$  à laquelle on supprime la ligne  $i$  et la colonne  $j$  (c'est donc une matrice carrée de taille  $(n - 1)$ ).

```
>>> mat = [[1, 2], [3, 4]]
>>> det(mat)
>>> -2
```

**Q 5 .** Écrire une fonction récursive qui compte le nombre d'occurrences d'un caractère dans une chaîne.

```
>>> nb_occurrences("b", "abba")
>>> 2
```

**Q 6 .** Écrire une fonction récursive qui permet de tester si une chaîne est anagramme d'une autre.

```
>>> anagramme ("baba", "abba")
>>> True
>>> anagramme ("babo", "abba")
>>> False
```

**Q 7 .** Soit la suite de Syracuse définie comme suit :  $u_0 = N$  et  $u_{n+1} = \frac{u_n}{2}$  si  $u_n$  est pair et  $u_{n+1} = 3 \times u_n + 1$  si  $u_n$  est impair. Écrire une fonction qui détermine le plus petit  $n$  tel que  $u_n = 1$  pour un nombre de départ  $N$ . Écrire également une fonction qui détermine la valeur maximale que  $u_n$  atteint avant d'arriver à 1..

**Q 8 .** Soit la suite définie comme suit :  $u_0 = u_1 = 1$  et  $u_{n+1} = 2 \times u_n + 3 \times u_{n-1}$ . Écrire une fonction récursive qui permet de calculer  $u_n$ . Exécuter votre fonction avec  $n = 200$ . Que remarquez-vous ? Améliorer votre programme pour que le calcul soit efficace.

**Q 9 .** Écrire une fonction qui permet de tester si une liste est triée..

```
>>> est_triee([5,2,1])
>>> False
>>> appartient([1,2,10,11,23])
>>> True
```

**Q 10 .** Écrire une fonction récursive qui permet de tester si un entier appartient à une liste d'entiers.

```
>>> appartient(3,[5,2,1])
>>> False
>>> appartient(2,[1,2,0])
>>> True
```

**Q 11 .** Écrire une fonction récursive qui permet de tester si un entier appartient à une liste **triée** d'entiers.

```
>>> appartient(3,[1,2,5])
>>> False
>>> appartient(2,[1,2,4,7,15,23])
>>> True
```

**Q 12 .** Écrire une fonction itérative puis récursive qui permet de calculer l'intersection de deux listes.

```
>>> intersection([1,2,5], [7,2,15,2,5])
>>> [2,5]
```

**Q 13 .** Même question que la question précédente mais cette fois-ci les deux listes sont considérées triées..