

**Exercice 1** Termes d'une suite

On considère la suite (u_n) définie par $u_1 = 5$ et : $\forall n \in \mathbb{N}^*$, $u_{n+1} = \frac{1 + u_n^3}{1 + u_n^2}$.

Q1 Écrire une fonction `suite` prenant en argument $n \in \mathbb{N}^*$ et renvoyant la liste $[u_1, u_2, \dots, u_n]$. On commencera par initialiser une liste `L` à la liste vide puis on calculera successivement les termes de la suite (u_n) et on les ajoutera dans `L`. On vérifiera que les premiers termes de la suite sont : 5 ; 4.846 ; 4.689 ; 4.528.

On considère ensuite la suite (a_n) définie par $a_0 = 1$ et $\forall n \in \mathbb{N}$, $a_{n+1} = (n+1) \times a_n$.

Q2 Écrire une fonction prenant en argument $n \in \mathbb{N}$ et renvoyant la liste $[a_0, a_1, \dots, a_n]$. Quelle est la valeur de a_n pour $n \in \mathbb{N}$?

On considère maintenant la suite (v_n) définie par : $\forall n \in \mathbb{N}^*$, $v_n = \sum_{k=1}^n \frac{1}{k^2}$.

Q3 Écrire une fonction prenant en argument $n \in \mathbb{N}$ et renvoyant $[v_1, v_2, \dots, v_n]$. On donne $v_1 = 1$, $v_2 = 1,25$, $v_3 \simeq 1,36$, $v_4 \simeq 1,42$. Quelle conjecture pouvez-vous faire quant au comportement de v_n lorsque $n \rightarrow +\infty$? On pourra calculer $\sqrt{6v_N}$ pour N grand.

Exercice 2 Clé de sécurité

On souhaite transmettre un message codé sous forme d'une liste de nombres entiers. Mais le canal de transmission utilisé n'est pas parfait. Il est donc possible que le message reçu ne soit pas exactement identique au message envoyé. Pour permettre au destinataire du message de vérifier s'il y a eu une erreur de transmission, on propose d'ajouter en fin de message un dernier nombre appelé clé de sécurité et noté s . Cette clé s'obtient en sommant tous les nombres du message, puis en prenant les deux derniers chiffres du nombre obtenu.

Par exemple, si le message à envoyer est `L = [41, 2, 100, 17, 64]` alors on calcule $41 + 2 + 100 + 17 + 64 = 224$, la clé de sécurité sera donc $s = 24$, et le message finalement envoyé sera `M = [41, 2, 100, 17, 64, 24]`.

Supposons alors qu'une erreur de transmission se produise et que le destinataire reçoive finalement le message `M' = [45, 2, 100, 17, 64, 24]`. En calculant $45 + 2 + 100 + 17 + 64 = 228$, le destinataire saura qu'il y a eu une erreur de transmission puisque la clé de sécurité dans le message reçu n'est pas $s = 28$, il pourra alors par exemple demander de renvoyer le message.¹

Q1 Écrire une fonction `cle` prenant en argument une liste de nombres `L` correspondant au message à envoyer et renvoyant la clé de sécurité s correspondante.

Q2 En déduire une fonction `securise` prenant en argument le message `L` et renvoyant le message à envoyer `M` avec la clé de sécurité. *On prendra soin de ne pas modifier la liste `L`, on vérifiera en particulier la valeur de `L` après avoir appelé `securise(L)`.*

1. Bien sûr, il peut arriver que la clé de sécurité soit correcte même s'il y a eu erreur de transmission, ou à l'inverse que l'erreur de transmission n'ait porté que sur la clé de sécurité elle-même la rendant incorrecte alors que le reste du message est correct... Mais il n'y a pas de risque zéro, et cet exemple n'est qu'un cas très simple de ce qu'on appelle les *codes correcteurs*.

On se place désormais du côté du destinataire qui a reçu le message M et souhaite savoir s'il est correct ou non. Pour cela, il lui faut extraire le dernier élément de la liste M grâce à la commande `pop` : la syntaxe suivante

```
1 x = M.pop()
```

permet de retirer le dernier élément de M pour le stocker dans la variable x . Cela modifie donc la liste M .

Q3 Dans la console, définissez la liste $A = [1, 2, 3]$. Enlevez ensuite le dernier élément de A et placez-le dans une variable a . Vérifiez les valeurs de A et de a . Rajouter ensuite l'élément a à A et vérifiez à nouveau les valeurs de A et de a .

Q4 Écrire une fonction `verifie` prenant en argument le message M reçu, et qui indique si la clé de sécurité du message M est bien celle attendue à la lecture du reste du message. Votre fonction renverra `True` si c'est le cas et `False` sinon.

Exercice 3

Q5 Écrire une fonction `appartient` prenant en argument une liste L et une variable a et renvoyant `True` si a est un élément de L et `False` sinon. *On n'utilisera pas la commande `in`.* Par exemple si $L = [2, 5, 3, -1, 0]$ alors `appartient(L, 3)` doit renvoyer `True`.

Q1 En utilisant la fonction précédente, écrire une fonction prenant en arguments deux listes L et M et renvoyant `True` si tous les éléments de L sont dans M et `False` sinon. Par exemple, si $L = [1, 4, 5]$ et $M = [2, 1, 7, 5, 9, 4]$ alors `inclus(L, M)` doit renvoyer `True`.

Q2 On suppose que L et M sont deux listes sans doublon, c'est-à-dire que tous les éléments de L sont distincts, et de même pour M . Écrire une fonction `memes_elem` prenant en arguments L et M et renvoyant `True` si L et M contiennent exactement les mêmes éléments (éventuellement dans un ordre différent) et `False` sinon. Par exemple si $L = [3, 7, 1, 2]$ et $M = [2, 1, 7, 3]$ alors `memes_elem(L, M)` doit renvoyer `True`.