

FONCTIONS ET PORTÉE DE VARIABLES

TP 02

Exercice 1.

Aux USA, l'unité de mesure de la température est le degré Fahrenheit. On a la correspondance suivante : $F = \frac{9}{5}C + 32$.

Écrire une fonction `Fahrenheit2Celcius` qui permet de convertir des Fahrenheit en degrés.

Exercice 2.

Le numéro de sécurité sociale est constitué de 13 chiffres auquel s'ajoute la clé de contrôle (2 chiffres). Exemple : 1.89.11.26.108.268.91

La clé de contrôle est calculée par la formule : $97 - (\text{numéro de sécurité sociale modulo } 97)$

1. Écrire une fonction `cle_nss` prenant en argument une variable `nss`, correspondant au numéro de sécurité sociale. Cette fonction permettra de calculer la clé de contrôle du numéro de sécurité sociale rentré en argument.

(Pour validation : `cle_nss(1921071123456) == 15`)

2. Écrire une fonction `est_nss_valide` prenant en argument une variable `nss_complet`, correspondant aux 15 chiffres (13 chiffres du numéro de sécurité social + 2 chiffres de clé). Cette fonction renverra `True` si le couple (numéro de sécurité social, clé) est valide, sinon `False`.

On utilisera la fonction `cle_nss`.

(Pour validation : `est_nss_valide(192107112345615) == True`)

Exercice 3.

1. Écrire une fonction `nombre(num)` qui indique le nombre de zéros nécessaires pour écrire le nombre `num`.

Par exemple, `nombre(5001) == 2` car il faut 2 zéros pour écrire 5001.

2. Modifier cette fonction en `nombre(num, digit)` pour indiquer le nombre de n'importe quel chiffre dans l'écriture de `num`.

Exercice 4 (algorithme de Héron d'Alexandrie).

Cet algorithme a été utilisé dans l'antiquité pour donner une approximation de $\sqrt{A}$ pour n'importe quel $A > 0$.

Soit (u_n) la suite définie par $u_0 = A$ et par récurrence, pour tout $n \in \mathbb{N}$,

$$u_{n+1} = \frac{1}{2} \left(u_n + \frac{A}{u_n} \right)$$

On peut démontrer que la suite converge vers $\sqrt{A}$ (c'est un bon exercice de colle).

1. Écrire une fonction `heron(A, n)` qui calcule le `n`-ième terme de la suite de Héron.

(Pour validation : `heron(3455723, 12) == 1904.85526`)

2. On admettra que si $|u_{n+1} - u_n| < \varepsilon$, alors $|u_n - \sqrt{A}| < 2\varepsilon$.

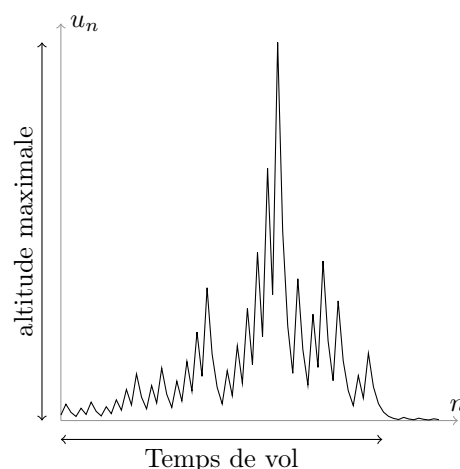
Écrire une fonction `approx` qui prend en argument le nombre A et un argument optionnel `precision` qui donne l'approximation de $\sqrt{A}$ avec la `précision` voulue. Par défaut, `precision` sera égale à 10^{-2} .

(pour validation : `approx(654, 0.001) == 25.5734237`)

Exercice 5 (conjecture de Syracuse).

Une suite de Syracuse est définie par $u_0 = u \in \mathbb{N}$ et la relation de récurrence, pour tout $n \in \mathbb{N}$:

$$u_{n+1} = \begin{cases} \frac{u_n}{2} & \text{si } u_n \text{ est pair} \\ 3u_n + 1 & \text{si } u_n \text{ est impair} \end{cases}$$



1. Écrire une fonction `syracuse(u)` qui calcule u_1 pour $u_0 = u$.
2. Écrire une fonction `syracuse_n(u,n)` qui calcule le n -ième terme de la suite de Syracuse pour $u_0 = u$.
On utilisera la fonction `syracuse`.
(Pour validation : `syracuse_n(358,15) == 88`)
3. Il est clair que si $u_0 = 1$, la suite parcourt le cycle 1, 4, 2, 1, 4, 2, ...
Une conjecture célèbre (la conjecture de Syracuse) annonce que, quel que soit $u \in \mathbb{N}$, la suite atteindra le cycle 1, 4, 2. Elle a été vérifiée pour tous les nombres entre 1 et 10^{20} . Le mathématicien Paul Érodos (1913-1996) a dit à propos de cette conjecture : «*les mathématiques ne sont pas encore prêtes pour de tels problèmes*». Il semble que ce soit encore le cas...

On appelle *temps de vol* de u , le premier indice n tel $u_n = 1$ pour la suite de Syracuse partant de $u_0 = u$.

Écrire une fonction `temps_vol(u)` qui calcule le temps de vol de u .

(Pour validation, `temps_vol(399) == 120`)

Rechercher parmi les entiers entre 1 et 1000 le temps de vol le plus long.

4. On appelle *altitude maximale* de u , le maximum atteint par la suite de Syracuse partant de $u_0 = u$.

Écrire une fonction `altitude_maxi(u)` qui calcule l'altitude maximale de u .

(Pour validation, `altitude_max(531) == 9232`)

Rechercher parmi les entiers entre 1 et 1000 l'altitude maximale la plus haute.

Exercice 6.

Notons $S_p(n) = \sum_{k=0}^n k^p = 1^p + 2^p + \dots + n^p$.

Écrire une fonction `barriere` qui prend un nombre entier K en entrée et un nombre optionnel p (par défaut égal à 1). Cette fonction retourne le plus petit entier n tel que la somme $S_p(n) > K$.

Par exemple `barriere(4) == 3` car $1 + 2 = 3 \leq 4$ et $1 + 2 + 3 = 6 > 4$.

Ou encore `barriere(300, 3) == 6` car $1^3 + 2^3 + 3^3 + 4^3 + 5^3 = 225 \leq 300$ et $1^3 + 2^3 + 3^3 + 4^3 + 5^3 + 6^3 = 441 > 300$.