

Devoir surveillé 2

Exercice 1

Le code UPC (Universal Product Code) utilise des nombres de 13 chiffres pour désigner un produit de consommation.

Les 12 premiers chiffres désignent le produit le 13-ième est une clé de contrôle.

On suppose que le code est écrit sous forme d'une liste $T = [t_0, t_1, \dots, t_{12}]$.

On note $P = \sum_{k=0}^6 t_{2k}$ la somme des termes d'indice pair de T et $I = \sum_{k=0}^5 t_{2k+1}$ la somme des termes d'indice impair de T

La clé C de ce code T est donnée par : $C = 3 * I + P = 3 \left(\sum_{k=0}^5 t_{2k+1} \right) + \sum_{k=0}^6 t_{2k}$. Le code barre est correct si la clé C est divisible par 10.

1. Écrire une fonction `Som_P(T)` prenant en entrée une liste d'entiers T de 13 chiffres et retournant la somme $P = \sum_{k=0}^6 t_{2k}$ et une fonction `Som_I(T)` retournant la somme $I = \sum_{k=0}^5 t_{2k+1}$.
2. Écrire une fonction `Code_Barre(T)` prenant en entrée une liste d'entiers T et retournant `True` si le code barre est correct et `False` si il est erroné.

La vérification portera sur le nombre de chiffres du code barre puis sur la clé de contrôle.

Exercice 2

1. (a) Écrire une fonction `Compare(L1, L2)` qui prend en paramètre deux listes $L1$ et $L2$ et qui renvoie `True` si tous les éléments de la liste $L1$ sont présents dans la liste $L2$ et `False` sinon.
 - (b) En déduire une fonction `MemeElements(L1, L2)` qui prend en paramètre deux listes $L1$ et $L2$ et qui renvoie `True` si ces listes contiennent les mêmes éléments et `False` sinon.
2. Écrire une fonction `Ind_Min(L)` qui prend en paramètre une liste L et qui renvoie le couple (\min, L_{\min}) contenant le minimum et la liste des indices du minimum de cette liste. Par exemple, `Ind_Min([1, 2, 3, 3, 1, 2, 3])` renverra le couple $(1, [0, 4])$.
3. Écrire une fonction `Supprime(L)` prenant en argument une liste L d'entiers et renvoyant une nouvelle liste contenant seulement les valeurs de L qui apparaissent exactement une fois dans L . Par exemple, si $L = [1, 0, 2, 1, 3, 2, 5]$, alors `Supprime(L)` renvoie la liste $[0, 3, 5]$.
On n'utilisera pas les méthodes `pop` ou `remove`

Exercice 3

Pour $n \in \mathbb{N}^*$, on représente un point (x_1, x_2, \dots, x_n) de \mathbb{R}^n en Python par une liste contenant ses coordonnées : $P = [x_1, x_2, \dots, x_n]$.

On rappelle que la distance euclidienne entre les points $A = (a_1, a_2, \dots, a_n)$ et $B = (b_1, b_2, \dots, b_n)$

de \mathbb{R}^n est donnée par : $d(A, B) = \sqrt{\sum_{i=1}^n (a_i - b_i)^2}$.

Dans le cas $n = 2$, on a donc : $d(A, B) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2}$.

1. Écrire une fonction `distance_2(A, B)` prenant comme argument deux points A et B de \mathbb{R}^2 (représentés par des listes de taille 2) et renvoyant la distance euclidienne entre ces deux points.
2. Écrire une fonction `distance_n(A, B)` prenant comme argument deux points A et B (représentés par des listes de même taille) et renvoyant la distance euclidienne entre ces deux points.
3. Écrire une fonction `plus_proche(liste)` prenant comme argument une liste de points et renvoyant les deux points les plus proches.