CHIFFRES ROMAINS

On considère le système d'écriture en chiffres romains suivant :

unités	=>	**I**	II	III	IV	V	VI	VII	VIII	IX	X
dizaines	=>	**X**	XX	XXX	XL	L	LX	LXX	LXXX	XC	C
centaines	=>	**C**	CC	CCC	CD	D	DC	DCC	DCCC	CM	M
milliers	=>	M	MM	MMM							

On note que l'on peut les regrouper en trois systèmes où les « unités » I,X,C de chaque système jouent un rôle particulier.

A-CONVERSION DE CHIFFRES ROMAINS EN ÉCRITURE DÉCIMALE

On s'appuiera sur la table de conversion suivante en python a introduire dans le script global :
convert = [['I','V','X','L','C','D','M'], [1,5,10,50,100,500,1000]]

1 - fonctions de conversion.

a - Ecrire une fonction **valRom**(carac) qui renvoie la valeur décimale associée à un caractère seul. Cette fonction reverra un message d'erreur si le caractère n'est pas possible.

b - Ecrire une fonction **romVal**(val) qui renvoie le caractère (=> de type **string**) associé à la valeur d'une lettre seule en chiffres romains. Cette fonction reverra un message d'erreur si la valeur n'est pas possible.

2 - Obtenir l'écriture décimale.

Soit une année en chiffres romains : MCMLXXXIV que remarque t-on lorsque l'on veut l'évaluer en décimales quant à la position des lettres les unes par rapport aux autres ?
=> On ajoute les valeurs associées aux lettres **sauf** si la lettre suivante est associée à une valeur plus grande qu'elle, auquel cas on doit la retrancher. La dernière lettre est toujours ajoutée.

M**C**M LXXX**I**V —> 1000 + (-100) + 1000 + 50 + 10 + 10 + 10 + **(-1)** + 5

Ecrire une fonction **roman2dec**(chaine) qui prend en argument la chaine en chiffres romains et renvoie la valeur du nombre entier correspondant.

Rq : on suppose dans cette partie que la chaine donnée est correcte, mais ce point plus délicat est abordé à la partie C.

B-CONVERSION DE L'ÉCRITURE DÉCIMALE EN CHIFFRES ROMAINS

L'idée est de mettre bout à bout l'écriture en chiffre romain de chacun des chiffres du nombre.
1984 —> 1 millier / 9 centaines / 8 dizaines / 4 unités

Il y a 4 systèmes chacun ne se composant que de 3 caractères possibles:
- millier [M, Q, T] on ne dépassera pas 3999 donc Q et T ne seront pas utilisés.
- centaines [C, D, M]
- dizaines [X, L, C]
- unités [I, V, X]

Soit 1984 —> M / CM / VXXX / IV

On peut encoder ces caractères avec la liste de liste suivante en python :
roman = [['M', 'Q', 'T'], ['C', 'D', 'M'], ['X', 'L', 'C'], ['I', 'V', 'X']]

1 Obtenir les chiffres un à un.
Ecrire une fonction **chiffreDec**(nombre) qui renvoie la liste des 4 entiers composant le nombre, dans l'ordre de la lecture [1, 9, 8, 4].

2 Ecriture d'un chiffre seul en romain.
Ecrire une fonction **chiffreRomain**(chiffre, position) qui prend en argument un chiffre entre 0 et 9, ainsi qu'une position de 1 à 4, et qui renvoie la chaine correspondant en romain
chiffreRomain(9, 2) —> 'CM' chiffreRomain(8, 3) —> 'LXXX' chiffreRomain(4, 4) —> 'IV'.
On utilisera une structure if - elif …. elif - else, sauf si vous trouvez mieux ?
Rq : d'autres approches sont possibles comme l'utilisation d'un dictionnaire.

3 Conversion d'un nombre décimale en chiffres romains
A l'aide des fonctions précédentes, écrire une fonction **dec2roman**(nombre) qui renvoie l'écriture en chiffres romains (de type string) du nombre entier.

4 Vérification indirecte de l'écriture en chiffres romains
A partir des fonctions **dec2roman** du B et **roman2dec** du A, écrire une fonction **validation**(chaine) qui renvoie vrai ou faux (booléen) en fonction de la validité de la chaine. Cette fonction doit être très simple.

C - VÉRIFICATION DIRECTE DE LA VALIDITÉ DE LA CHAINE. (PARTIE PLUS DIFFICILE)

Dans cette partie on souhaite déterminer directement à partir de la chaine de caractères en chiffres romains si celle-ci est valide ou non ?

En effet toutes les chaines possibles selon la partie A ne sont pas considérées comme valides.

a. On n'a jamais 4 caractères identiques à la suite : 4 => IV et non IIII.
   Rq : cette particularité serait due à une difficulté de l'oeil humain à compter les traits à partir de IIII.

b. On ne peut cumuler ou soustraire que les « unités » I, X, ou C d'un même système :
   IV, XXV, CCC, CM sont correctes, mais VVV, VX ou IL et XM non.

c. On ne peut soustraire qu'une seule unité ex : 3 => III et non IIV. Ceci permet l'unicité de la notation romaine dès lors que l'on compte par paquet de 5 il suffit de mettre 3 bâtons.

d. Enfin il faut une cohérence globale de l'écriture : ~~MCDXMCCC~~ est incorrecte !
   Elle doit correspondre à la succession de chiffres vue en partie B.

1 Isoler les chiffres de la chaine (d)
Ecrire une fonction **splitRoman**(chaine) qui prend en argument la chaîne et retourne la liste des chiffres mais chacun sous forme de chaine en chiffre romains. 1984 —> ['M', 'CM', 'LXXX', 'IV'].
Pour cela, on lit la chaine et si le caractère à droite nécessite de changer de système (au sens de la liste roman du B), c'est qu'il fait partie d'un autre chiffre.

2 Vérifier la validité de l'écriture d'un chiffre seul en romain

A partir des critères ci-dessus (a,b et c), vérifier que l'écriture d'un chiffre est correcte :
5 -> V oui mais VX non ! 7 dizaines -> LXX oui mais XXXC non !

a - Une difficulté provient du fait que l'étape 1 splitRoman('MCDXMCCC') renvoie une liste avec des chiffrages incorrects ['M', 'CD', '**XM'**, 'CCC'] : on ne peut retrancher 'X' qu'à 'C' ou 'L', pas à 'M'. Ecrire une fonction **estUnite**(carac, caracBis) qui dit si carac est bien une unité vis-à-vis du caractère caracBis. Ex : 'I' pour 'V' ou 'X' oui, mais pas 'X' pour 'M' ou 'D'. La liste roman du B permet d'identifier les unités dans chaque système.

b - Ecrire ensuite une fonction **verificationRomain**(chaine) qui prend en argument la chaine associée au chiffre romain et qui renvoie le booléen Vrai ou faux selon les critères de validité.
A vous de les implémenter.

3 Vérification globale de l'écriture en chiffres romains

Ecrire une fonction **validationDirecte**(chaine) qui utilise les fonctions du 1 et du 2 et qui renvoie le booléen Vrai ou Faux pour l'ensemble de l'écriture en chiffres romains.