Informatique - DM 1

Consignes :
Vous devez rendre le code correspondant aux différentes questions sous **deux** formes.

— Sous forme **numérique** :
Envoyer à l'adresse `pduclosson@ac-lyon.fr` au plus tard le dernier jour des vacances, un fichier nommé `DM1_NOM.py` en remplaçant `NOM` par votre patronyme en majuscule. Ce fichier devra comporter tout le code demandé **et seulement celui-ci** [1]. Il sera présenté sous la forme suivante :

```
# -*- coding: utf-8 -*-
# DM - 1  :  Nom

# Question 1

def est_premier(n):
.....
.
```

— Sous forme **papier** :
Rendre votre copie à la date indiquée avec le code demandé et toutes les réponses aux différentes questions. Le code doit être proprement écrit [2], l'indentation et la syntaxe respectées.
Avant chaque partie de code, vous devez expliquer **en bon français** les algorithmes et indiquer précisément le rôle joué par chaque variable que vous nommerez de manière pertinente (suivez les recommandations qui vous ont été données en cours et TD). Comme toujours, il faut être concis autant que possible, évitez les romans.

Pour installer Python sur votre ordinateur faites une recherche google par exemple : `pyzo`.

Exercice 1 : Fais tourner mais reste premier !

Le nombre 113 est remarquable dans le sens suivant : il est premier et les 2 autres nombres obtenus par permutation cyclique de ses chiffres le sont aussi : 311 et 131 sont premiers. De plus, 113 est le plus petit nombre de 3 chiffres possédant cette propriété.

On se propose de déterminer, s'il existe, le plus petit nombre de 6 chiffres qui possède cette propriété.

On rappelle que `a%b` calcule le reste de la division euclidienne de a par b et que `a//b` donne son quotient. Il est interdit d'utiliser des fonctions Python répondant immédiatement aux

1. Ne laisser aucune instruction qui vous aurait été utile à la mise au point mais qui n'ont pas à figurer dans la version finale du code.
2. Obligatoirement à la main, pas de sortie d'imprimante.

questions ! Le recours aux chaines de caractères est interdit. On demande, autant que possible d'écrire un code minimisant les calculs mais qui doit rester **facilement compréhensible**.

Pour les questions 1 à 7, il est indiqué le résultat que l'on doit obtenir pour certaine valeur afin de vous permettre de tester votre code. Il est obligatoire de vérifier ces exemples et de nombreux autres que vous produirez par vous-même.

1. Écrire une fonction `est_premier` prenant en paramètre un entier n que l'on supposera plus grand que 2 (ne pas le vérifier dans le code) et renvoyant `True` ou `False` suivant que n est premier ou pas.

   Exemples : `est_premier(2017) : True` `est_premier(832) : False`

2. Écrire une fonction `liste_chiffres` prenant en paramètre un entier naturel non nul n (ne pas vérifier dans le code que $n \in \mathbb{N}^*$) et renvoyant la liste des chiffres de l'écriture de n en base 10. Le recours aux chaînes de caractères est évidemment interdit.

   Exemple : `liste_chiffres(1789) : [1, 7, 8, 9]`

3. Écrire une fonction `decale_liste` prenant en paramètre une liste sur laquelle elle agit en décalant ses éléments de manière circulaire, le dernier élément prenant la place du premier.

   Exemple : `[1, 7, 8, 9]` devient `[9, 1, 7, 8]`

4. Écrire une fonction `nombre_liste` réalisant l'opération inverse de `liste_chiffres`.

   Exemple : `nombre_liste([1, 7, 8, 9]) : 1789`

5. Écrire une fonction `decale_nombre` prenant en paramètre un entier naturel n et renvoyant le nombre obtenu en décalant de manière circulaire les chiffres de l'écriture décimale de n d'un cran vers la droite.

   Exemple : `decale_nombre(113) : 311`

6. Écrire une fonction `est_solution` prenant en paramètre un entier naturel n (comportant un nombre a priori quelconque de chiffres en écriture décimale) et renvoyant `True` ou `False` suivant que n vérifie les conditions du problème ou pas.

   Exemples : `est_solution(113) : True` `est_solution(2013) : False`

7. Écrire une fonction `cherche_solution` prenant en paramètre un entier naturel N et renvoyant le plus petit nombre comportant N chiffres qui soit solution du problème s'il en existe un. S'il n'existe pas de solution, la fonction ne devra rien renvoyer.

   Exemple : `cherche_solution(3) : 113`

8. Répondre au problème posé en préambule.

Exercice 2 : Facultatif

Écrire une fonction `plus_grande_sous_somme` qui prend en paramètre une liste `L` non vide d'entiers relatifs et qui renvoie la plus grande somme d'une suite **non vide** de termes consécutifs de la liste `L`.

- Utiliser uniquement des opérations élémentaires de Python.
- Expliquer ce que fait votre algorithme.
- Proposer l'algorithme de meilleure complexité, c'est à dire avec un nombre d'opérations le plus petit possible pour des listes de très grande taille.