TP d'informatique (Python) n°5

" Boucles conditionnelles. Ecriture dans une base autre que 10,
Nombres premiers"

But du TP
- Ecrire les premières boucles conditionnelles et les structures **while**
- Consolider les connaissances sur les instructions et la syntaxe **if** et **for**.
- Découvrir la syntaxe des fonctions de la

Vous penserez **à sauvegarder régulièrement** votre travail dans un fichier dans votre espace personnel. Pour le nom de fichier, n'utilisez ni accent, ni espace, ni caractères spéciaux (autres que _) : en effet, certains projets nécessitent l'ouverture de fichiers que l'on aura écrits au préalable : or cet appel peut s'avérer délicat si le nom de ce fichier possède des caractères "non habituels"

A. Ecriture d'un entier en base autre que 10

1. Ecrire une séquence d'instructions qui, étant donné un entier naturel non nul, fournit le tuple de ses chiffres dans son écriture en base 10. L'algorithme est le suivant : On part de l'entier m = n et du tuple **Lchif** vide (avec l'instruction **Lchif** = tuple([ ]). On rajoute alors à gauche au tuple **Lchif** le reste de la division euclidienne de m par 10 et on transforme m en son quotient par 10, et on recommence. On itère ce processus jusqu'à ce que l'entier m soit égale à 0 (La procédure pourra utiliser les opérateurs **%** et //)
Tester pour n = 1, 9, 55555, 123456789 et 100!

2. Ecrire une séquence d'instructions qui, étant donné un entier naturel non nul, fournit le tuple de ses chiffres dans son écriture en base 8.
Tester pour n = 1, 9, 55555, 123456789 et 100!

3. Ecrire une séquence d'instructions qui, étant donné un entier naturel n compris entre 0 et 15 et qui affecte à la variable **c16** la chaine de caractères "n" si ce nombre est compris entre 0 et 9, "A" si n vaut 10, "B" si n vaut 11, … , "F" si n vaut 15.
Lorsque cette séquence d'instructions sera testée et validée, on l'insérera dans une fonction Python que l'on écrira sous la forme (en respectant les indentations)

def **Chiffres16**(n) :
    Vos instructions
    return **c16**

4. Ecrire une séquence d'instructions qui, étant donné un entier naturel non nul, donne le tuple de ses "chiffres" dans son écriture en base 16.
Tester pour n = 1, 9, 55555, 123456789 et 100!

5. Ecrire une séquence d'instructions qui, étant donné un entier naturel non nul écrit en base 8, donne l'écriture décimale de cet entier.
Tester pour n = $(1)_8$, $(45)_8$, $(55555)_8$, $(1234567)_8$ et $(345607)_8$

B. Autour des nombres premiers

1. Ecrire des séquences d'instructions qui, étant donné un entier n naturel (qu'on supposera supérieur ou égal à 4), retourne le booléen True si n est un nombre premier et False sinon. Tester sur n = 4523, 1234567 ou 1234567111111.

On utilisera l'algorithme suivant :

```
res ← True
pour k allant de 2 à √n faire                Rem : il faudra peut – être aller jusqu'à 1 + E(√n)
    si k divise n alors
        res ← False
Résultat : res
```

2. Charger le fichier texte appelé **est_premier** dans la page de cours en ligne du site internet du lycée. **R**ecopier le contenu de ce fichier dans votre session de travail. **E**xécuter la commande suivante (après l'avoir écrite dans l'éditeur) :

```
for n in range (1, 50) :
    if est_premier(n) :
        print(n)
```

3. Combien y-a-t-il d'entiers naturels inférieurs à 100 qui sont premiers ? Même chose avec 10^k pour k dans {3, 4, 5, 6}.

On utilisera l'algorithme suivant :

```
cpt ← 0
pour n allant de 1 à N faire                N est la borne demandée
    si n est premier alors
        cpt ← cpt + 1
Résultat : cpt
```

4. Déterminer les $100^{\text{ième}}$ et $1000^{\text{ième}}$ nombres premiers.

5. Combien existe-il de n $<10^6$ tels que n et n+2 sont premiers ?

6. Trouver le plus petit entier n supérieur à 10^{10} tel que n et n+2 sont premiers.

7. Lire le code de la fonction **est_premier** : combien réalise-t-elle d'opérations élémentaires pour tester la primalité d'un entier n dans le pire des cas ? dans le meilleur des cas ?
Combien y-a-t-il d'opérations élémentaires pour déterminer ma primalité de tous les entiers < N ? (on donne le résultat suivant : la proportion d'entiers < N qui sont des nombres premiers est de l'ordre de grandeur 1/ln(N))