Laboratoire 9

EXERCICE 1

Écrivez une fonction qui permet de vérifier sur une chaine de caractères représente un code postal canadien valide. Si oui, cette fonction retourne True. Un code postal valide est une chaîne composée d'exactement **6** caractères alphanumériques.

```
6   if __name__=="__main__":
7       zip_code = "H1B1B2"
8       print(is_valid_zip_code(zip_code)) # Output: True
9
10      zip_code = "H1B 1B2"
11      print(is_valid_zip_code(zip_code)) # Output: False
```

Indication : utiliser la méthode **isalnum()** qui renvoie True si tous les caractères sont alphanumériques, c'est-à-dire des lettres de l'alphabet (a-z) et des chiffres (0-9). False si au moins un caractère n'est pas alphanumérique. Par exemple :

```
str = "Pierre2005"
flag = str.isalnum()
print(flag) # output is : True
```

EXERCICE 2

Écrire petit à petit un programme qui renvoie le pluriel d'un mot donné.

1. Pour une chaîne mot, par exemple "chat", affiche le pluriel de ce mot en rajoutant un "s".
2. Pour un mot, par exemple "souris", affiche la dernière lettre de cette chaîne (ici "s"). Améliore ton programme de la première question, en testant si la dernière lettre est déjà un "s" :
    a. si c'est le cas, il n'y a rien à faire pour le pluriel,
    b. sinon il faut ajouter un "s".
3. Teste si un mot se termine par "al". Si c'est le cas, affiche le pluriel en "aux" (le pluriel de "cheval" est "chevaux"). (Ne tiens pas compte des exceptions.)
4. Rassemble tout ton travail des trois premières questions dans une fonction **met_au_pluriel().** La fonction n'affiche rien, mais renvoie le mot au pluriel. Exemples :
    a. **met_au_pluriel("chat")** renvoie "chats"
    b. **met_au_pluriel("souris")** renvoie "souris"
    c. **met_au_pluriel("cheval")** renvoie "chevaux"

EXERCICE 3

La distance de Hamming entre deux mots de même longueur est le nombre d'endroits où les lettres sont différentes. Par exemple : **J**A**P**ON et **S**A**V**ON

La première lettre de JAPON est différente de la première lettre de SAVON, les troisièmes aussi sont différentes. La distance de Hamming entre JAPON et SAVON vaut donc 2.

Écris une fonction ***distance_hamming()*** qui calcule la distance de Hamming entre deux mots de même longueur.

EXERCICE 4

a) Verlan. Écris une fonction ***verlan()*** qui renvoie un mot à l'envers : SALUT devient TULAS.
b) Déduis-en une fonction qui teste si un mot est un palindrome ou pas. Un palindrome est un mot qui s'écrit indifféremment de gauche à droite ou de droite à gauche ; par exemple RADAR est un palindrome.

EXERCICE 5

Une molécule d'ADN est formée d'environ six milliards de nucléotides. L'ordinateur est donc un outil indispensable pour l'analyse de l'ADN. Dans un brin d'ADN il y a seulement quatre types de nucléotides qui sont notés A, C, T ou G. Une séquence d'ADN est donc un long mot de la forme : TAATTACAGACCTGAA...

1. Écris une fonction ***presence_de_A()*** qui teste si une séquence contient le nucléotide A. Exemple : ***presence_de_A("CTTGCT")*** renvoie False
2. Écris une fonction **position_de_AT()** qui teste si une séquence contient le nucléotide **A** suivi du nucléotide **T** et renvoie la position de la première occurrence trouvée. Par exemple :
    a. **position_de_AT("CTTATGCT")** renvoie 3
    b. **position_de_AT("GATATAT")** renvoie 1
    c. **position_de_AT("GACCGTA")** renvoie None
3. Écris une fonction **position()** qui teste si une séquence contient un code donné et renvoie la position de la première occurrence. Par exemple : **position("CCG","CTCCGTT")** renvoie 2
4. Un crime a été commis dans le château d'Adéno. Tu as récupéré deux brins d'ADN, provenant de deux positions éloignées de l'ADN du coupable. Il y a quatre suspects, dont tu as séquencé l'ADN. Sauras-tu trouver qui est le coupable ?

    Premier code du coupable : CATA
    Second code du coupable : ATGC

    ADN du colonel Moutarde :
    CCTGGAGGGTGGCCCCACCGGCCGAGACAGCGAGCATATGCAGGAAGCGGCAGGAATAAGGAAAAGC

    ADN de Mlle Rose:
    CTCCTGATGCTCCTCGCTTGGTGGTTTGAGTGGACCTCCCAGGCCAGTGCCGGGCCCCTCATAGGAGAG

ADN de Mme Pervenche:
AAGCTCGGGAGGTGGCCAGGCGGCAGGAAGGCGCACCCCCCCAGTACTCCGCGCGCCGGGACAGAAT

ADN de M. Leblanc:
CTGCAGGAACTTCTTCTGGAAGTACTTCTCCTCCTGCAAATAAAACCTCACCCATGAATGCTCACGCAAG