PROGRAMMATION PYTHON
TD/TP 3 TIC-01

Objectifs:

- Manipuler les chaînes de caractères
- Manipuler les listes
- Définir et utiliser des fonctions

Les chaînes de caractères

Exercice N°1 : (Palindrome)

Ecrivez une fonction **palindrome** qui teste si le mot passé en entrée est un palindrome (le mot est égal à son miroir). La fonction renvoie un booléen

Exercice 2 : (Recherche de caractère,)

1. Écrivez une fonction **cherche** qui prend en argument un caractère c et une chaîne de caractères s et qui renvoie true si c apparaît dans s, et false sinon.
   Par exemple, cherche (’a’, "cheval") renvoie True et cherche (’a’, "école") renvoie False.

2. Modifiez la fonction cherche afin qu’elle renvoie au lieu de true la première occurrence du caractère c dans la chaîne s et si le caractère n’est pas présent la fonction renvoie -1

Exercice N°3 : **(**Distance de Hamming)

La distance de Hamming entre deux mots est une notion utilisée dans de nombreux domaines (télécommunications, traitement du signal, .). Elle est définie, pour deux mots de même longueur, comme **le nombre de positions où les deux mots ont un caractère différent**. Écrivez une fonction hamming qui calcule la distance de Hamming entre deux mots lorsqu’ils ont la même longueur, et qui renvoie -1 sinon.

Par exemple, hamming("aaba", "aaha") renvoie 1,
hamming("poire", "pomme") renvoie 2 et
hamming("stylo", "bouteille") renvoie -1.

Exercice N°4 :

Écrivez une fonction **scrabble** qui prend en argument deux chaînes de caractères **mot** et **lettres_disponibles** et qui renvoie True si on peut écrire mot en utilisant au plus une fois chaque lettre de la chaîne lettres_disponibles et qui renvoie False sinon.

Par exemple, scrabble("maison", "auiysmzanpo") renvoie True et
scrabble("bungalows", "hbteslo") renvoie False.

Exercice N°5 :

Deux mots sont des anagrammes si on peut obtenir l'un à partir de l'autre en permutant les lettres. Écrire une fonction anagramme qui prend en argument deux chaînes u et v et qui renvoie True si u et v sont des anagrammes et qui renvoie False sinon.
Par exemple, anagramme("parisien", "aspirine") renvoie True et
anagramme("chaise", "disque") renvoie False.

Exercice N°6 :

Un problème fréquent d'un compilateur et des traitements de textes est de déterminer si les parenthèses d'une chaîne de caractères sont balancées et proprement incluses l'une dans l'autre. Par exemple :
Ecrivez une fonction qui retourne True si une chaîne de caractères est proprement écrite et bien balancée, et False sinon.

```
Input : {[]{()}}
Output : Balanced

Input : [{}{}(]
Output : Unbalanced
```

Les listes unidimensionnelles

Exercice N°7 :

Créez puis affichez les listes suivantes :

- L1 : liste croissante des entiers naturels pairs strictement inférieurs à 20
- L2 : coupe (slicing) du 3ème au 6ème élément inclus de la liste L1 précédente
- L3 : liste L1 sans les premier et dernier éléments
- L4 : liste décroissante des entiers naturels impairs strictement inférieurs à 10
- L5 : liste décroissante des racines carrées des entiers naturels impairs inférieurs à 10.

Exercice N°8 : (Manipulation simple des listes)

Le but de cet exercice est de déterminer si étant donnée une liste de nombres entiers positifs tous différents, un nombre donné peut s'écrire comme somme de deux d'entre eux.

1. Écrivez une fonction ***tousDifferents*** qui étant donnée une liste d'entiers renvoie True si tous les éléments de la liste sont différents et False si la liste contient au moins deux éléments identiques.

2. Écrivez une fonction ***sommeN*** qui étant donné un entier n supposé positif renvoie la somme des entiers de 0 à n.

3. Écrivez une fonction ***sousListe*** qui prend comme arguments une liste Li et une position p comprise entre 0 et len (Li)−1 et renvoie la liste des éléments situés strictement après p dans Li.

   Exemple : Pour la liste [12,3,2,8] et la position 0, la fonction sousListe renverra la liste [3,2,8] , pour la position 2, elle renverra la liste [8] et pour la position 3, elle renverra la liste de taille 0 [] .

4. Écrivez une fonction ***toutesLesPaires*** qui étant donnée une liste d'entiers Li renvoie la liste vide (de taille 0) si tous les éléments de Li ne sont pas tous différents et sinon renvoie une liste de listes d'entiers correspondant à toutes les paires d'entiers de Li .

   Exemple : Pour la liste [12,3,2,8] , la fonction renverra la liste de listes [[12,3],[12,2],[12,8],[3,2],[3,8],[2,8]] .

5. Écrivez une fonction ***sommePaire*** qui étant donnés une liste d'entiers Li et un entier x renvoie :

   - La liste vide (de taille 0) si les éléments de Li ne sont pas tous différents.
   - La liste vide (de taille 0) si il n'existe pas deux entiers a et b dans Li tels que x=a+b.
   - Et dans les autres cas, une liste de deux éléments a et b tels que a et b sont dans Li et vérifient x=a+b.

   Exemple : Pour la liste [12,3,2,8] et l'entier 14, la fonction sommePaire renverra la liste [12,2] .

   Remarque : Il peut y avoir plusieurs paires d'entiers a et b qui correspondent, la fonction en renverra alors une.

Exercice N°9 : (Ordre de lettres)

Il pîaart que puor la lctreue, l'orrde des lrttees à l'iétunreir des mots n'a acnuue itnpocmare. La sulee chose qui cptmoe est que la pemièrre et la dne`eirre lttree seonit à leur pclae.

Ecrivez un programme permettant de tester cette théorie. Il devra prendre en entrée une chaîne de caractères et mélanger aléatoirement les lettres à l'intérieur des mots. On supposera que la chaîne de caractères ne comporte pas de signe de ponctuation.

On écrira d'abord une fonction ***Permute_Mot*** qui prend en entrée un mot et permute les lettres à l'intérieur du mot en gardant les première et dernière lettres inchangées, puis une deuxième fonction ***Permute_phrase*** qui prend en entrée une phrase, et applique la première fonction sur chacun des mots, puis renvoie la phrase contenant les mots permutés.

Exemples :
permute("Je vais avoir une bonne note") peut renvoyer 'Je vias avior une bnone ntoe'.

Indications : lesmots=phrase.split(" ") renvoie la liste des mots de la phrase. On utilisera aussi le module random qui contient la fonction random.randint(a,b) qui renvoie un nombre aléatoire compris dans l'intervalle [*a, b*].

Les listes de listes

Exercice N°10 :

Etant donnée une liste 2D de points et un entier K. Il est demandé de trouver et d'afficher les k points les plus proches de l'origine (0,0). Notez que la distance entre deux points est la distance euclidienne.

Exemples :

```
Input : point = [(3, 3), (5, -1), (-2, 4)], K = 2
Output : [(3, 3), (-2, 4)]

Input : point = [[1, 3], [-2, 2]], K = 1
Output : [[-2, 2]]
```

Exercice N°11 :

Python admet des listes hétérogènes, de profondeur quelconque, par exemple : ['a', [], 8, [2.1, ["xx",1]], [[[None]]]].

Écrivez une fonction qui aplatit une telle liste, en construisant une nouvelle à un seul niveau, avec les mêmes éléments qui ne sont pas des listes, dans le même ordre.

Ici : ['a', 8, 2.1, "xx", 1, None]. Notez que les listes vides sont éliminées (les listes genre [[]] également).