Université Paris Dauphine **Licence MIE 1re année**
Algorithmique et programmation 1 **2025-2026**

6. Listes en Python et boucle `for` (TP)

Exercice 6.7 *(Debugage d'un exercice sur les chaînes)*

Nous cherchons à compter le nombre de mots d'une chaîne et le nombre de caractères par mot. Pour ce faire, l'équipe Python a écrit une fonction mais malheureusement cette dernière ne fonctionne pas ! Vous allez devoir la corriger et la faire fonctionner.

Pour récupérer le fichier, aller sur Moodle, dans la semaine 7 et copier, dans votre répertoire personnel, le fichier `Programme à corriger : Debug_Chaine.py`.

```
#Fonction qui compte le nombre de mots d'une chaine
# et le nombre de caractères par mot
def compteMotsEtCaracteres(ch):
    i = 0

    while i < =len(ch):
        mot = ""
        affichage = ""
        #Récupération du mot courant
        while (i <= len(ch) and ch[i] != " "):
            mot = mot + ch[i]
            i = i+1

        #Pour ne compter que les mots non vides
        if mot != "":
            nbMots = nbMots + 1
            affichage = affichage + "Le mot \"" + mot +
             "\" contient : \n"

            #Pour compter les caractères du mot courant
            while (mot != ""):
                affichage = affichage + "  " +mot.count(mot[0]) +
             " fois le caractère '" + mot[0] + "'\n"
                mot.replace(mot[0], "")
        i = i + 1

    if nbMots != 0:
        print("La chaine \", ch, "\" contient : ", nbMots,
         " mots\n", affichage, sep="", end="")
```

1. Exécuter le programme. Que vous indique l'erreur de syntaxe ? Corriger cette erreur.
2. Exécuter à nouveau le programme. Que vous indique l'erreur de syntaxe ? Corriger cette erreur.
3. Appeler la fonction en lui passant en paramètre la chaîne `"ananas poire kiwi"`. Quel est le message d'erreur ? Corriger cette erreur.

4. Appeler à nouveau la fonction (après avoir ré-exécuté le programme corrigé précédemment) en lui passant en paramètre la chaîne `"ananas poire kiwi"`. Quel est le message d'erreur ? Corriger cette erreur.
5. Appeler à nouveau la fonction (après avoir ré-exécuté le programme corrigé précédemment) en lui passant en paramètre la chaîne `"ananas poire kiwi"`. Pourquoi selon vous le programme tourne-t-il indéfiniment ? Quelle boucle pose problème selon vous ? Insérer des affichages pour le vérifier et corriger l'erreur.
6. Appeler à nouveau la fonction en lui passant en paramètre la chaîne `"ananas poire kiwi"`. Que signifie le message d'erreur ? Corriger le programme pour que ce message n'apparaisse plus.
7. Que se passe-t-il si vous inversez les conditions de la 2ème boucle `while`. Expliquer.
8. Que faut-il modifier dans le programme pour que celui-ci affiche la description de tous les mots et pas uniquement du dernier ? Par exemple, l'appel doit afficher :

```
La chaine "ananas poire kiwi" contient : 3 mots
Le mot "ananas" contient :
  3 fois le caractère 'a'
  2 fois le caractère 'n'
  1 fois le caractère 's'
Le mot "poire" contient :
  1 fois le caractère 'p'
  1 fois le caractère 'o'
  1 fois le caractère 'i'
  1 fois le caractère 'r'
  1 fois le caractère 'e'
Le mot "kiwi" contient :
  1 fois le caractère 'k'
  2 fois le caractère 'i'
  1 fois le caractère 'w'
```

Exercice 6.8

On veut calculer la somme des entiers de 1 à n, sans utiliser la formule $\frac{n(n+1)}{2}$ mais à l'aide d'une boucle `for`.

1. Écrire le programme Python correspondant. Dans un premier temps, écrire une boucle `for` pour calculer la somme des entiers de 1 à 10. La somme sera stockée dans une variable nommée `somme`. Afficher cette somme à l'écran en fin de programme.
2. Modifier votre programme pour obliger l'utilisateur à saisir un nombre n strictement positif, puis calculer la somme des entiers de 1 à n, et l'afficher à l'écran en fin de programme.
3. Proposer un programme utilisant la fonction `sum()` permettant de répondre à la question précédente. Remarque : en tapant `help(sum)` dans l'interpréteur Python, vous obtiendrez l'aide concernant cette fonction.

Exercice 6.9 *(Mise en œuvre de l'exercice 4.3)*

Écrire une fonction `TabAlea(n,a,b)` qui prend trois arguments - 3 valeurs entières n, a et b - et qui renvoie une liste de n entiers aléatoirement choisis entre a et b inclus.

Puis, écrire une fonction `TabProduit(T)` qui prend une liste `T` de valeurs entières en argument et qui renvoie le produit de tous les éléments de la liste.

Écrire un programme Python qui lit trois entiers n, a et b et qui appelle successivement les fonctions `TabAlea()` et `TabProduit()` pour afficher la liste de valeurs entières ainsi que le produit de ces valeurs.

Exercice 6.10

1. Écrire un programme qui crée une liste contenant les 20 premiers multiples de 7 strictement positifs.
2. Compléter votre programme pour qu'il affiche les 20 premiers multiples de 7 strictement positifs, séparés par un point-virgule sur une même ligne, et aller à la ligne après chaque multiple de 3.
3. Modifier ce programme pour qu'il :
   - fasse saisir par l'utilisateur un nombre n et un nombre p strictement positifs et crée une liste contenant les n premiers multiples de 7 ;
   - affiche cette liste selon le format suivant : les n premiers multiples de 7, séparés par un point-virgule sur une même ligne, mais en allant à la ligne après chaque multiple de p.
4. Modifier ce programme pour que, en plus, il compte les multiples de p qui auront été affichés et affiche ce nombre à la fin.

Exemple d'exécution :

```
  Combien de multiples de 7 voulez-vous afficher ? 20
  Vous irez à la ligne après chaque multiple de ...
  (nombre strictement positif svp) : 3
7 ; 14 ; 21
28 ; 35 ; 42
49 ; 56 ; 63
70 ; 77 ; 84
91 ; 98 ; 105
112 ; 119 ; 126
133 ; 140 ;
Vous avez affiché 6 multiples de 3
```

Exercice 6.11 *(Mise en œuvre de l'exercice 6.5)*

1. Écrire un programme Python qui fait saisir à l'utilisateur une liste L de listes de nombres, dont le nombre d'éléments est inconnu. Le programme doit demander à l'utilisateur s'il souhaite ajouter des éléments à la liste ou s'il souhaite s'arrêter.
2. Modifier L en supprimant de chaque sous-liste toute occurrence de sa plus petite valeur.
3. Créer une nouvelle liste qui contient les sous-listes de L triées par ordre croissant selon leur longueur.

Exercice 6.12

Écrire un programme en Python qui, étant donnée, une liste de listes de nombres (initialisée dans le programme), affiche la somme de la liste de nombres la plus longue (s'il y en a plusieurs, on prendra la première).
Par exemple, pour la liste $[[1, 2, 3], [], [2, 3, 0, 4], [2, 5], [6, 0, 2, 4]]$, le programme affichera 9.

Exercice 6.13

Après un examen, les notes des étudiants sont les suivantes : Florian : 2, Antoine : 12, Charles1 : 0, Robert : 0, Charles2 : 8, Erwan : 7, Jean : 20, Xavier : 11, Didier : 20, Alain : 0, Hadi : 12.

1. Utiliser deux listes pour stocker les noms des étudiants et les notes associées.
2. Calculer et afficher la moyenne de l'examen.
3. Afficher le nom et la note de chacun des élèves, avec la mention « admis » si la note est supérieure ou égale a 10, ou bien « recalé » sinon.
4. Stocker dans des listes séparées les élèves admis et les élèves recalés.
5. Les personnes ayant eu 0 à l'examen ne se sont jamais présentées. Modifier votre programme pour supprimer les informations relatives à ces élèves.
6. Modifier votre programme pour permettre à l'utilisateur de saisir de nouvelles notes.

Exercice 6.14

Soit la matrice des distances (en km) entre les villes suivantes :

Auxerre	0	45	35	25	20
Avallon	45	0	30	70	100
Clamecy	35	30	0	60	55
Joigny	25	70	60	0	10
Migennes	20	100	55	10	0
	Auxerre	Avallon	Clamecy	Joigny	Migennes

On indice les villes de 0 à 4 dans l'ordre alphabétique (0 pour Auxerre, 1 pour Avallon, ... et 4 pour Migennes).

1. Écrire une fonction `lireDistance()` qui lit les distances entre ces 5 villes, saisies au clavier par l'utilisateur, et renvoie le tableau à deux dimensions des distances entre ces villes. On veillera à ce que pour chaque couple de ville i et j, la distance ne soit saisie qu'une seule fois par l'utilisateur (car pour tout i et tout j, on a `distance[i][j]= distance[j][i]`). En Python, ce tableau correspond à une variable `distance` de type `list` de `list` telle que `distance[i][j]` contient la distance entre la ville indicée par `i` et la ville indicée par `j`.
2. Écrire une fonction qui vérifie l'inégalité triangulaire, c'est-à-dire que pour toutes les valeurs i, j et k, on a : `distance[i][j]` $\leq$ `distance[i][k]+distance[k][j]`. Cette fonction doit renvoyer la listes des triplets i, j, k qui ne respectent pas l'inégalité triangulaire.
3. Écrire le programme principal qui appelle la fonction `lireDistance()`, construit le tableau des distances et teste si les inégalités triangulaires sont respectées. Si elles sont respectées, un message l'indiquant doit être affiché, sinon, pour chacun des triplets ne vérifiant pas l'inégalité triangulaire, il faut afficher les trois villes concernées.