EXAMEN DE FIN DU SEMESTRE 1

Matière : INFORMATIQUE

Classes : 1ère Année MP – T – PC **Durée :** 2 h

Toutes les réponses doivent être rédigées en Python.
L'indentation dans les instructions Python sera prise en considération dans le barème.
*Vous trouvez, en **Annexe**, quelques méthodes prédéfinies sur les conteneurs.*

Exercice 1 (6 points)

On dit qu'une chaîne de caractères, formée par deux ou plusieurs mots, est un *"tautogramme"*, si tous ses mots commencent par la même lettre ; et on dit qu'elle est un *"totalogramme"*, si chacun de ses mots commence et se termine par la même lettre (sans distinction entre majuscule et minuscule, dans les deux cas).

Exemples : La chaîne *"Le lion lape le lait lentement"* est un *tautogramme*.
La chaîne *"Amira emprunte temporairement à Aziza ses souliers"* est un *totalogramme*.

Travail demandé :

Écrire un script Python permettant de :

1. Saisir une chaîne de caractères **ch** composée au moins de deux mots ;
2. Afficher le message *"tautogramme"*, si la chaîne **ch** est un *tautogramme* et le message *"n'est pas tautogramme"*, sinon ;
3. Afficher le message *"totalogramme"*, si la chaîne **ch** est un *totalogramme* et le message *"n'est pas totalogramme"*, sinon.

Exercice 2 (6 points)

Une course de voitures entre plusieurs participants s'est déroulée en trois circuits différents. Soit **n** un entier strictement positif (**n** > 0) qui représente le nombre de participants. Les noms des participants sont stockés dans une liste **Lnoms** de taille **n** et formée par des chaînes de caractères. Les scores de chaque participant sont stockés dans une liste **Lscores** dont chaque élément est un tuple contenant trois nombres réels correspondants aux temps effectués par chaque participant dans chaque circuit.

Exemple : Lnoms = ['Ali' , 'Rami' , 'Slim']
Lscores = [(22.5 , 19.0 , 21.5) , (23.5 , 20.25 , 20.0), (21.25 , 19.25 , 20.25)]

On suppose que les listes **Lnoms** et **Lscores** sont déjà créées et on vous demande d'écrire les instructions Python permettant de :

1. Afficher l'entier **n** correspondant au nombre de participants en course ;
2. Créer et afficher le dictionnaire **Dcourse** dont les clés sont les noms des participants et les valeurs sont les moyennes respectives des scores de chaque participant. Pour notre exemple, on aura à l'exécution : `{'Ali': 21.0,'Rami': 21.5,'Slim': 20.25}` ;
3. Afficher le nom du gagnant de la course, c'est-à-dire celui qui a effectué la moyenne des scores minimale en course. Pour notre exemple, on aura à l'exécution :

```
Le gagnant de la course est Slim.
```

Exercice 3 (8 points)

L'objectif de cet exercice est de faire le cryptage d'une chaîne de caractères ne contenant que des lettres alphabétiques en majuscules, au moyen de la liste **L** suivante :

L = [['A', 'B', 'C','D','E'], ['F','G', 'H', 'I','J'], ['K', 'L', 'M', 'N','O'], ['P', 'Q', 'R', 'S','T'], ['U', 'V', 'W', 'X','Y', 'Z']]

L'idée de ce cryptage consiste à remplacer chaque lettre de la chaîne donnée par son **rang dans la liste de listes L**, c'est-à-dire, '**A**' est remplacé par '**11**' (car c'est le 1er élément dans la 1ère liste), '**B**' est remplacé par '**12**' (car c'est le 2ème élément dans la 1ère liste), et ainsi de suite.

Exemple : Le cryptage du mot "**PYTHON**" donne le résultat suivant : '415545233534'.

Chaîne non cryptée	**'P'**	**'Y'**	**'T'**	**'H'**	**'O'**	**'N'**
Chaîne cryptée	**'41'**	**'55'**	**'45'**	**'23'**	**'35'**	**'34'**

Une manière de trouver facilement le code de chaque lettre de l'alphabet consiste à créer un dictionnaire **D** dont les clés sont les lettres de l'alphabet se trouvant dans **L** ; et leurs valeurs correspondantes sont leurs positions dans **L**, étant des chaînes de caractères à deux chiffres.

Travail demandé :

On suppose que la liste **L** est déjà créée et on vous demande d'écrire les instructions Python permettant de :

1. Créer, à partir de la liste **L**, le dictionnaire **D** contenant les lettres de l'alphabet ainsi que leurs codes et dont l'affichage donne : `D = {'A': '11', 'B': '12', 'C': '13', 'D': '14', 'E': '15', 'F': '21', 'G': '22', 'H': '23', 'I': '24', 'J': '25', 'K': '31', 'L': '32', 'M': '33', 'N': '34', 'O': '35', 'P': '41', 'Q': '42', 'R': '43', 'S': '44', 'T': '45', 'U': '51', 'V': '52', 'W': '53', 'X': '54', 'Y': '55', 'Z': '56'}`
2. Saisir une chaîne de caractères **mot** formée uniquement par des lettres alphabétiques en majuscule ;
3. Créer et afficher la liste de cryptage **Lc** formée par les chaînes à deux chiffres correspondant aux lettres de la chaîne **mot** entrée par l'utilisateur. Pour notre exemple, la liste de cryptage du mot "**PYTHON**" est **Lc** = ['41', '55', '45', '23', '35', '34'] ;
4. Créer et afficher la chaîne de cryptage **Mc** à partir de sa liste de cryptage. Pour notre exemple, si **Lc**= ['41', '55', '45', '23', '35', '34'], alors **Mc** = '415545233534'.

Annexe

Méthode	*Rôle*
`ch.isupper()`	Retourner le booléen `True` si la chaîne `ch` est **toute** en majuscule, `False`, sinon.
`ch.islower()`	Retourner le booléen `True` si la chaîne `ch` est **toute** en minuscule, `False`, sinon.
`ch.isalpha()`	Retourner le booléen `True` si la chaîne `ch` est formée par des lettres alphabétiques **uniquement**, `False`, sinon.
`ch.isnumeric()`	Retourner le booléen `True` si la chaîne `ch` est formée par des caractères numériques **uniquement**, `False`, sinon.
`ch.isalnum()`	Retourner le booléen `True` si la chaîne `ch` est formée par des lettres alphabétiques ou des caractères numériques **uniquement**, `False`, sinon.
`ch.lower()`	Afficher la conversion de la chaîne `ch` **toute** en minuscule.
`ch.upper()`	Afficher la conversion de la chaîne `ch` **toute** en majuscule.
`ch.split()`	Retourner la liste formée par les sous-chaînes se trouvant dans la chaîne `ch` et étant séparées par le caractère espace `' '`. *Exemple* : `"Hello world !".split()` retourne `['Hello','world','!']`
`L.append(x)`	Ajouter l'élément `x` à la fin de la liste `L`.
`L.pop()`	Retourner et supprimer le dernier élément se trouvant dans la liste `L`.
`L.index(x)`	Retourner l'indice de la première occurrence de l'élément `x` dans la liste `L` s'il existe, un message d'erreur, sinon.
`D.keys()`	Retourner une liste contenant les clés dans le dictionnaire `D`.
`D.values()`	Retourner une liste contenant les valeurs dans le dictionnaire `D`.
`D.items()`	Retourner une liste contenant les tuples `(clé,valeur)` dans le dictionnaire `D`.