

Examen n°1

Matière : INFORMATIQUE

Classe : 1^{ère} Année BG Durée : 2h

*Toutes les réponses doivent être rédigées en Python.
L'indentation dans les instructions Python sera prise en considération dans le barème*

Exercice 1 : (3 points)

Ecrire la(les) bonne(s) réponse(s) après l'exécution de chaque instruction :

1. Quels sont les types mutables en python :

- A) Les listes
- B) Les tuples
- C) Les dictionnaires

2. `L = [2, 4, 6, 7, 9]`
`L[2], L[-1] = L[0], L[2]`

Que vaut L ?

- A) [2,4,2,7,6]
- B) [2,4,2,7,2]
- C) [6,4,9,7,9]

3. `d = {'France':'Paris', 'Italie':'Rome', 'Allemagne': 'Berlin'}`

Que vaut 'Paris' in d ?

- A) France
- B) True
- C) False

4. `2*"2"` est égal à :

- A) '22'
- B) '4'
- C) erreur

5. `print('hello'.replace('e', 'a'))`

Que vaut la sortie ?

- A) hallo
- B) halla
- C) hella

6.

```
ch= ""  
for c in "Bon":  
    ch+=c  
print(ch)
```

Que vaut la sortie ?

- A) Bon
- B) noB
- C) BonBonBon

Exercice 2 : (4 points)

Un entier K à n chiffres est dit nombre de **Kaprekar** si lorsqu'on élève K au carré, la somme du nombre composé des n chiffres de droite au nombre composé de n ou $n-1$ chiffres de gauche retourne le nombre d'origine.

Exemples :

$$9^2 = 81 \text{ et } 1+8=9$$

$$45^2 = 2025 \text{ et } 20+25=45$$

$$297^2 = 88209 \text{ et } 209+88=297$$

Ecrire un **programme python** qui permet de :

1. Saisir un entier $N > 2$.
2. Créer un tuple T qui contient N entiers, strictement positifs, saisis au clavier.
3. Afficher dans une liste L tous les entiers de **Kaprekar** de T .

Exercice 3 : (6 points)

Etant donnée une liste L des animaux qui contient des tuples. Chaque **tuple** est formé par trois éléments : le nom de l'animal, le nombre d'individus de cet animal et son écosystème. On suppose que dans L , on ne retrouve pas deux tuples avec le même nom d'animal et le même écosystème.

Exemple :

$L = [(\text{'Loup'}, 13, \text{'Forêt boréale'}), (\text{'Salamandre'}, 21, \text{'Forêt tempérée'}), (\text{'Ours'}, 3, \text{'Toundra'}), (\text{'Loup'}, 20, \text{'Toundra'}), (\text{'Salamandre'}, 13, \text{'Toundra'}), (\text{'Singe'}, 50, \text{'Forêt tempérée'})]$

Ecrire les **instructions python** permettant de :

1. Créer et afficher un **dictionnaire Danimal** dont les clés sont les noms des animaux et les valeurs sont leurs nombres d'individus.
Pour notre exemple, on aura à l'exécution : $\{\text{'Loup'}: 33, \text{'Salamandre'}: 34, \text{'Ours'}: 3, \text{'Singe'}: 50\}$
2. Afficher le(s) couple(s) ayant les individus les plus nombreux.
3. Saisir, par clavier, un nom d'animal **animal**. Si ce nom existe dans le dictionnaire **Danimal**, un message qui retourne le nombre de ses individus ainsi que son(s) écosystème(s) sera affiché, sinon le message 'zero' sera affiché.

NB. : Les fonctions **D.keys()**, **D.values()** et **D.items()** retournent des listes qui contiennent, respectivement, les clés, les valeurs et les couples dans le dictionnaire **D**.

Exercice 4 : (7 points)

On définit le poids d'une chaîne de caractères en majuscules comme étant la somme des produits de la position de chaque **voyelle** dans cette chaîne par son rang dans l'alphabet français. Cet alphabet compte 26 lettres : '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'.

Etant donné le dictionnaire `voy` dont les clés sont les voyelles et les valeurs sont leurs rangs. `voy = {'A' :1, 'E' :5, 'I' :9, 'O' :15, 'U' :21, 'Y' :25}`

Si la chaîne ne contient aucune voyelle alors son poids est égal à zéro.

N.B. : La position d'un caractère dans une chaîne donnée est son rang dans la chaîne commençant à partir de 1 et non pas son indice commençant de 0.

Exemples :

- ✓ La chaîne 'BONNE' contient 2 voyelles 'O' et 'E' de positions respectives 2 et 5, son poids est alors égal à : $2 * 15 + 5 * 5 = 55$
- ✓ La chaîne 'CHANCE' contient 2 voyelles 'A' et 'E' de positions respectives 3 et 6, son poids est alors égal à : $3 * 1 + 6 * 5 = 33$

Ecrire un **programme python** qui permet de :

1. Saisir une taille $1 < N \leq 20$.
2. Créer une liste `Lch` de 10 chaînes non vides. Chaque chaîne contient N caractères et formée par des **lettres alphabétiques majuscules**.

N.B. :

Les fonction `ch.isalpha()` retourne `True` si la chaîne `ch` contient uniquement des **alphabétiques**.

La fonction `ch.isupper()` retourne `True` si la chaîne `ch` est **en majuscule**.

3. Créer et afficher une **liste LP** composée des poids calculés par chaque chaîne dans `Lch`.
4. Créer et afficher une **liste LchP** où chaque élément est un **tuple** ayant comme premier élément la chaîne saisie dans `Lch` et comme deuxième élément son poids.
5. Déterminer et afficher la chaîne ayant le poids le plus élevé.