Niveau:MP1,PC1,T1 **Durée :** 1 heure	**D.S. n°2: Informatique**	**I.P.E.I.M.** **A.U.:**2022/2023

DOCUMENTS NON AUTORISÉS
VOUS POUVEZ ÉVENTUELLEMENT UTILISER LES FONCTIONS PYTHON DÉCRITES A L'ANNEXE

Exercice 1. (6 pts)

1. Soit la fonction **ds2** suivante. Sachant que **isinstance**(objet, dict) est une fonction qui renvoie **True** si l'objet spécifié est de type dictionnaire, sinon **False**.

```
def ds2(d):
    L = [1]
    if len(d) == 0:
        return 1
    else:
        for k in d:
            if isinstance(d[k], dict):
                L.append(1+ds2(d[k]))
        return max(L)
```

Donner les résultats de l'exécution des commandes ci-dessous.

```
>>> D1 = {'a': 1, 'b': [1, 2, 3], 'c': 4, 'd': (5, 6)}
>>> D2 = {'a': {'ab': 8, 'ac': 9}, 'b': [1, 2, 3], 'c': 4, 'd': (5, 6)}
>>> D3 ={'a':{'ab':{'abb':10},'ac':9}, 'b':[1, 2, 3], 'c':4, 'd':(5, 6)}
>>> print('D1_Result =', ds2(D1))
.........................................
>>> print('D2_Result =', ds2(D2))
.........................................
>>> print('D3_Result =', ds2(D3))
.........................................
```

2. On souhaite écrire une fonction récursive appelée **saisie** qui demande à l'utilisateur de saisir une chaîne de caractères. Cette chaîne doit être de longueur 10 et commencer par une majuscule. La fonction accepte en entrée un entier N représentant le nombre d'essais de saisie permis.

Si **N** est égal à 0, la fonction retourne une chaîne vide. Sinon, elle invite l'utilisateur à saisir une chaîne de caractères. Si la chaîne saisie est valide, elle est renvoyée. Si elle n'est pas valide, un message d'erreur est affiché et la fonction Saisie est rappelée avec **N-1** comme argument.
Afin de répondre aux besoins, On vous demande de compléter le code suivant:

```
def saisie(.....):
    # Test si le nombre d'essais permis est dépassé
    if N == 0:
        print("Vous avez dépassé le nombre d'essais permis.")
        ................
    # Demander à l'utilisateur d'entrer une chaîne de caractères
    s = input(
        "Saisir une chaîne appropriée: ")
    # Valider la chaîne entrée
    if ....................................:
        return ..............
    else:
        print("Chaîne entrée non valide. Il vous reste", N-1, "essais.")
        return ..................
```

Exercice 2

Une entreprise souhaite gérer ses personnels en utilisant une application Python. Pour l'assister, on va suivre la demarche suivante:

Partie 1:

On appelle chaîne valide toute chaîne de caractère ayant le format suivant:

```
"Nom;Prénom;Poste;Salaire_brut;Ancienneté"
```

Où:

- Nom" et "Prénom" sont des chaînes de caractères.
- Poste": est une chaîne de caractères appartenant à la liste["cadre","employé", "ouvrier"].
- Salaire_brut est un nombre réel compris entre 1000 et 5000.
- Ancienneté est un nombre entier compris entre 0 et 30.

On suppose qu'on dispose de la fonction **chaine_valide**(**ch**) qui permet de renvoyer **True** si **ch** obéit au format donné ci-dessus, **False** sinon.

On vous demande d'écrire:

1. La fonction **saisie_valide** qui permet de saisir une chaîne valide.

2. La fonction **saisir_employés** qui permet à l'utilisateur de saisir *n* chaînes valides et qui renvoie une liste ordonnée par ordre alphabétique.

3. La fonction **calculer_salaire_net** qui accepte comme argument une chaîne valide représentant un personnel et qui calcule son salaire net **sn** sachant que:

```
sn = Salaire_brut*(1-cs)+anciennete*100
    avec cs = 25 % pour les cadres et 15 % pour les autres.
```

Partie 2:

4. Écrire La fonction **creer_fichier**, qui prend en paramètres le nom du fichier *nom_fichier* et le nombre d'employés *N* et qui crée le fichier "*nom_fichier*" avec les *N* chaînes de caractères valides saisies par l'utilisateur. Chaque ligne du fichier représente un personnel. Elle a la forme d'une chaîne valide complete par ';' et le salaire net de l employee en question: autrement dit selon le format suivant :

```
"Nom;Prénom;Poste;Salaire_brut;Ancienneté;salaire_net"
```

5. L'entreprise souhaite enregistrer les informations concernant ses 20 personnels dans un fichier appelé "base.txt". Donner la ligne de commande python qui permet de créer ce fichier.

6. Écrire La fonction **liste_par_poste**(*nom_fichier*, *nom_poste*) qui prend en entrée le nom du fichier et l'intitulé du poste, et renvoie la liste des lignes du fichier "*nom_fichier*" correspondant au "nom_poste". (vous devez prendre en considération des exceptions.)

   Par exemple, l'appel **liste_par_poste**(*'base.txt'*, *'cadre'*) retourne la liste des personnels occupant le poste "*cadre*" inscrit dans le fichier *'base.txt'*.

7. Écrire n script python qui, pour chaque type de personnels, cadre", "employé" et "ouvrier", crée un fichier correspondant (par exemple pour les ouvriers on crée le fichier ouvrier.txt et ainsi de suite). Chaque ligne du fichier créé doit obéir au format suivant:

```
"i;Nom;Prénom;Salaire_brut;Ancienneté;Salaire_net",
```

   avec "i" étant le numéro de la ligne dans le fichier créé.

Annexe: Quelques méthodes sur les chaînes - Listes:

- **ch.strip(carac)** retourne une copie de la chaîne de caractères **ch** nettoyée des caractères carac placés en début et en fin de chaîne (s'ils existent). Si carac=None, les bloques de caractères non-imprimables (' ' , '\t' et '\n') placés en début et en fin de chaîne, sont supprimées.

```
>>>'abcdaaa'.strip('a')
  'bcd'
>>>'  a b c \t\n  '.strip()
  'a b c'
```

- **ch.split(sep)** découpe la chaîne **ch** suivant le séparateur **sep** et renvoie le résultat sous forme de liste de chaines. Si sep=None, les mots sont séparés par une suite quelconque de caractères non-imprimables (' ' , '\t' et '\n').

```
>>>'a = b'.split('=')
  ['a ',' b']
>>>'a b c \t\n  d'.split()
  ['a', 'b', 'c', 'd']
```

- **ch.replace(oldStr, newStr)** renvoie une nouvelle chaîne en remplaçant la chaîne **oldStr** (si elle existe) par une autre **newStr** dans **ch**. Exemple:

```
>>> s = 'a + b'
>>> ch = s.replace('+','*')
>>> ch
  'a * b'
```

- **ch.join(L)** assemble les éléments d'un iterable **L** (ses elements sont forcement des chaines) en utilisant la chaîne **ch** comme séparateur. Exemple:

```
>>> '/'.join(['ali','ahmed','alia'])
  'ali/ahmed/alia'
```

- **L.sort()** modifie la liste **L** elle-même, en réorganisant les éléments dans l'ordre (si les elements sont triables).

```
>>> L0 = [5, 2, 3, 1, 4];L1 = ['x','z','a','y','b]
>>> L0.sort();L0
[1, 2, 3, 4, 5]
>>> L1.sort();L1
    L1 = ['a','b','x','y','z']
```