CONTROLE DE FIN DE THEME

Exercice 1 – Indexations de tables (5pts)

Ci-dessous, le **contenu** d'un **fichier CSV** d'un système de **gestion des projections d'un cinéma** :

```
titre;genre;duree;salle;places;ventes
Inception;Science-fiction;148;3;140;119
Titanic;Drame;195;4;140;91
Avatar;Science-fiction;162;4;140;140
Sacré Graal;Comédie;112;2;100;88
Gladiator;Action;155;3;140;108
Coco;Animation;105;1;100;60
```

durée est exprimée en **minutes**, **salle** correspond au **numéro de salle**, et **ventes** est le **nombre de billets** actuellement vendus.

a. Quel est la **durée** du film qui a actuellement **le plus de billets vendus** ?

Le code suivant permet **d'indexer** la table en Python sous la forme d'un **Tableau de P-uplets Nommés** :

```
1  csv_f = open("magasin.csv", [ A ]) # ouverture du fichier
2  entetes = csv_f.readline().strip().split(";") # En-têtes
3
4  projections = []
5  ligne = csv_f.readline()
6  while [ B ]: # tant que la ligne n'est pas vide
7      valeurs = ligne.strip().split(";")
8      enrg = {}
9      for i in range(len(valeurs)):
10         enrg[ C ] = liste_valeurs[i]
11     projections.append(enrg)
12     ligne = csv_f.readline()
13 csv_f.close()
```

b. **Par quoi remplacer** l'emplacement [A], qui est le **paramètre « mode de lecture »** de la fonction ?

c. **Quelle est la condition** à indiquer dans l'emplacement [B] pour permettre **d'itérer sur chaque ligne** du fichier jusqu'à les avoir **toutes lues** ?

d. **Par quoi remplacer** l'emplacement `C` pour permettre **d'associer la bonne clé** à la valeur dans le **dictionnaire** de l'enregistrement ?

e. Quelle est la **valeur d'évaluation** de **l'expression suivante**, exécutée après le code ci-dessus ?

```
type(produits[0])
```

Exercice 2 – Sélections dans une table (10pts)

Un **autre code** permet **d'indexer la table** au format d'un **Tableau Doublement Indexé**, ce qui nous permet d'obtenir finalement **la table suivante** :

```
films = [
    ["Inception", "Science-fiction", 148, 3, 140, 119],
    ["Titanic", "Drame", 195, 4, 140, 91],
    ["Avatar", "Science-fiction", 162, 4, 140 140],
    ["Sacré Graal", "Comédie", 112, 2, 100, 88],
    ["Gladiator", "Action", 155, 3, 140, 108],
    ["Coco", "Animation", 105, 1, 100, 60]
]
```

Les descripteurs sont respectivement le **titre**, le **genre**, la **durée**, le **numéro de salle**, la **capacité maximale de la salle** et le **nombre de places vendues**.

a. **Écrire une fonction `taux_occup_max(films)`**, où **`films`** est une **table** au format de la table ci-dessus, et qui **renvoie le titre** du **film** qui a **le plus grand taux d'occupation** (*taux d'occupation = nombre de places vendues / capacité maximale de la salle*).

b. **Écrire une fonction `ventes_genre(films, genre)`**, où **`films`** est une **table** au format de la table ci-dessus, **`genre`** est le **nom d'un genre de film** et qui **renvoie la somme totale des ventes** des films **correspondants à ce genre**.

c. **Compléter le code suivant** qui utilise une **création de liste par compréhension** pour afficher **`True`** si **un film dont le titre est '`titre`'** est **projeté dans la salle de projection '`salle`'**, et **`False`** dans le cas contraire. Vous devez compléter **l'intérieur des crochets** ainsi que **l'expression utilisée à l'avant-dernière ligne**.

```
# utiliser la table contenue dans la variable 'films'
titre = "Inception" # utiliser la variable 'titre'
salle = 3 # utiliser la variable 'salle'
liste_compr = [




                                                ]
salle_correcte = ..............................
print(produit_present)
```

d. **Compléter le code suivant** qui utilise une **création de liste par compréhension** pour afficher **le nombre de films différents qui sont projetés dans une salle '`salle`'**. Vous devez compléter **l'intérieur des crochets** ainsi que **la fonction utilisée à l'avant-dernière ligne**.

```
# utiliser la table contenue dans la variable 'films'
salle = 1 # utiliser la variable 'salle'
liste_compr = [




                                                ]
nb_proj_en_salle = ................(liste_compr)
print(nombre_en_rayon)
```

Exercice 3 – Tri de tables (5pts)

Dans cet exercice, on considère **la table ci-dessous** qui est un **Tableau de P-uplets Nommés** :

```
cinema = [
    { "titre": "Inception", "genre": "Science-fiction",
      "duree": 148, "salle": 3, "places": 140, "ventes": 119 },
    { "titre": "Titanic", "genre": "Drame",
      "duree": 195, "salle": 4, "places": 140, "ventes": 91 },
    { "titre": "Avatar", "genre": "Science-fiction",
      "duree": 162, "salle": 4, "places": 140, "ventes": 140 },
    { "titre": "Sacré Graal", "genre": "Comédie",
      "duree": 112, "salle": 2, "places": 100, "ventes": 88 },
    { "titre": "Gladiator", "genre": "Action",
      "duree": 155, "salle": 3, "places": 140, "ventes": 108 },
    { "titre": "Coco", "genre": "Animation",
      "duree": 105, "salle": 1, "places": 100, "ventes": 60 }
]
```

a. **Compléter le code suivant** pour que **`meilleures_ventes`** soit une table des films **triés par ordre croissant des places vendues**. Vous pouvez utiliser **soit un bloc de définition** de fonction (`def`…), **soit une lambda-expression**.

```
meilleures_ventes = sorted(magasin, key = ......................)
print(meilleures_ventes)
```

On dispose d'un **dictionnaire `notes_films`** qui **associe** à chaque **titre de film** une **note sur 5**.

```
notes_films = { "Inception": 4.6, "Titanic": 4.3, "Coco": 4.5,
                "Interstellar": 4.7, "Matrix": 4.4,
                "Le Parrain": 4.8 }
```

b. **Compléter le code suivant** pour que **`meilleures_notes`** soit une table des films **triés par ordre croissant de notes**. Si un film **est absent du dictionnaire `notes_films`**, alors on lui donne **une note arbitraire de `2.5`**.
Vous pouvez utiliser **soit un bloc de définition** de fonction (`def`…), **soit une lambda-expression**.

```
meilleures_notes = sorted(magasin, key = ......................)
print(meilleures_notes)
```

c. **Expliquer** quel **ordre de tri** sera utilisé par la **fonction `sorted`** dans l'expression suivante :

```
titre_films = ["Inception", "Titanic", "Coco",
               "Interstellar", "Matrix", "Le Parrain"]
def f(x):
    return -len(x)
print(sorted(noms_produits, key = f))
```