TEK-UP
Ecole Supérieure Privée
Technologie & Ingénierie

Ecole Supérieure Privée Technologies & Ingénierie			
Type d'épreuve	: ☐ Devoir	■ Examen SESSION PRINCIPALE	
Enseignantes	: S. CHEBBI, K. TBARKI, S. GHARSALLI, S. JALEL		
Matière	: Python		
Année Universitaire	: 2022-2023	**Semestre** : 2	
Classe	: TIC		
Documents	: ☐ Autorisés	■ Non autorisés	
Date	: 24/05/2023	**Durée :** 1h30mn	
Nombre de pages	: 8 pages		
Barème	: 4 pts + 10 pts + 6 pts		

Nom et Prénom : .....

Classe : TIC -1 -.....

QCM (4 pts) : **Sélectionnez la bonne réponse en remplissant le tableau suivant :**

Question	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Réponse									

1. Quelle est la sortie de code suivant ?

```
my_list = [1, 2, 3, 4, 5]
print(my_list[-2])
```

a. 4
b. 3
c. 2
d. 5

2. Soit le code suivant : print(int("1.5")*10), choisir l'affichage correspondant :
a. Affiche 1.5
b. Affiche une erreur
c. Affiche 15

3. Soit le code suivant, choisir la sortie

```
m="\\\\\"
print(len(m))
```

a. Affiche : 5
b. Affiche : 3
c. Lève une erreur

Ne rien écrire ici

4. Soit le code suivant, donner la sortie correspondante :

```
t = (0, 2, 4, 10)
t = t[1 : -1]
t = t[0]
print(t)
```

a. (2,)
b. Erreur
c. 2

5. Quelle est la sortie du code suivant ?

```
ch = "Python Programming"
print(ch[::-1])
```

a. "gnimmargorP nohtyP"
b. "Programming Python"
c. "Python Programming"
d. "margorP nohtyP"

6. Qu'affiche le script suivant ?

```
resultat = ""
for c in "Bonsoir":
    resultat = c + resultat

print(resultat)
```

a. Bonsoir
b. riosnoB
c. BonsoirBonsoirBonsoirBonsoirBonsoirBonsoir
d. RiosnoBriosnoBriosnoBriosnoBriosnoBriosnoB

7. Qu'affiche le script suivant :

```
nom=['Juin','Avril','Octobre']
print(nom[-1][-1])
```

a. Juin
b. n
c. Octobre
d. e

8. Dans un constructeur de classe, qu'est-ce que le paramètre self représente ?

a. Le nom de l'instance en cours de création.
b. La classe elle-même.
c. Les attributs de la classe.
d. Les méthodes de la classe

9. Qu'affiche le bout de code suivant :

```
class Shape:
    def __init__(self, color):
        self.color = color

    def get_color(self):
        return self.color

class Circle(Shape):
    def __init__(self, color, radius):
        super().__init__(color)
        self.radius = radius

    def get_area(self):
        return 3.14 * self.radius ** 2

my_circle = Circle("Red", 5)
print(my_circle.get_color())
```

a. "Red"
b. "Circle"
c. "Shape"
d. Une erreur se produira lors de l'exécution.

Exercice 2 : (10 pts)

Nous nous intéressons à créer des classes pour gérer les vols d'une compagnie aérienne locale qui organise des vols entre des villes. Plus précisément nous nous intéressons aux plans de vol (les vols disponibles ainsi que l'heure de départ) entre les différentes villes.

1. Développez une classe **Vol_direct,** qui représentera un vol direct entre deux villes (pas d'escale dans une ville intermédiaire), définie par les attributs suivants :

    a. **dep** et **arr :** qui désignent respectivement la ville de départ et la ville d'arrivée
    b. **jour :** qui désigne le jour de la semaine (lundi, mardi, …)
    c. **heure :** (un entier entre 0 et 24 qui représente l'heure de départ)

    a. Définissez le constructeur de cette classe. **(1.25 pt)**

.....
.....
.....
.....
.....
.....

b. Ecrivez une méthode **Affiche()** qui affiche, pour un objet de type Vol_direct, une chaine de la forme : **(1.25 pt)**
« Ce vol part de '**Tunis**' vers '**Djerba**' le '**lundi**' à **9** heure »

.....
.....
.....
.....
.....
.....

2. Développez une classe **Vols** qui représentera tous les vols le long de la semaine en utilisant la classe **Vol_direct**. Pour ce faire :

    a. Définissez le constructeur de cette classe avec **un seul attribut qui est une liste de vols**. **(1.25 pt)**

.....
.....
.....
.....
.....
.....

b. Développez une méthode **Liste_successeurs()** qui, étant donnée une ville de départ passée comme paramètre, retourne une liste contenant les villes d'arrivées. **(1.25 pt)**

.....
.....
.....
.....
.....
.....

c. Ecrivez une méthode **Appartient()** qui vérifie si une ville appartient au plan du vol que ce soit comme ville d'arrivée ou de départ. **(1.25 pt)**

.....
.....
.....
.....
.....
.....

d. Ecrivez une méthode **Affiche()** qui affiche tous les vols directs. **(1.25 pt)**

.....
.....
.....
.....
.....
.....

3. Ecrivez un programme principal permettant de : **(2.5 pts)**

a. Créer une liste **LV** de **n** objets **Vol_direct,** saisie par l'utilisateur.
b. Créer un objet Vol nommé **V** à partir de la liste déjà créée.
c. Afficher tous les vols
d. Saisir une ville **qui doit appartenir au plan du vol** puis calculer et afficher la liste de ses successeurs

Ecole Supérieure Privée des Technologies et de l'Ingénierie "TEK-UP"
Ghazela Technoparc (Lot 6 / AFI), 2088 Ariana, Tunisia 23 814 000 71 709 899 www.tek-up.tn
Capital: 2 000 000 dinars - RC: B138162003 - MF: 0846352Y - RIB: 08010011041000519045

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

Exercice 3 : (6 pts)

Dans une ligue de football, nous disposons de plusieurs équipes qui participent à un tournoi. Chaque équipe est composée de joueurs qui participent à des matchs et qui marquent des buts. L'objectif est de stocker les statistiques des joueurs pour évaluer leurs performances individuelles. Pour cela, nous utiliserons un dictionnaire appelé "**joueurs**" pour stocker les statistiques des joueurs, comme c'est indiqué dans l'exemple ci-dessous.

Chaque joueur représente une clé dans le dictionnaire "joueurs". Les statistiques du joueur sont représentées par un autre dictionnaire qui comprend les 3 clés : "matchs_joues" pour le nombre de matchs joués, "buts_marques" pour le nombre de buts marqués et "passes_decisives" pour le nombre de passes décisives effectuées.

```
joueurs = {
    "Joueur 1": {
        "matchs_joues": 5,
        "buts_marques": 3,
        "passes_decisives": 2
    },
    "Joueur 2": {
        "matchs_joues": 4,
        "buts_marques": 2,
        "passes_decisives": 1
    },
    "Joueur 3": {
        "matchs_joues": 6,
        "buts_marques": 5,
        "passes_decisives": 3
    },
    "Joueur 4": {
        "matchs_joues": 3,
        "buts_marques": 1,
        "passes_decisives": 0
    }
}
```

1. Implémentez une fonction **ajouter_joueur()** qui prend en paramètres le nom, le nombre de matchs joués, le nombre de buts marqués et le nombre de passes décisives d'un joueur, et qui ajoute ce joueur au dictionnaire “joueurs”. **(2 pts)**

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

2. Implémentez une fonction **modifier_statistiques()** qui prend en paramètres le nom d'un joueur ainsi que les nouvelles statistiques de ce joueur (nombre de matchs joués, de buts marqués et de passes décisives), et qui met à jour ses statistiques après avoir vérifié son existence. **(2 pts)**

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

3. Implémentez une fonction **meilleur_joueur()** qui parcourt la structure de données des joueurs et identifie le joueur ayant les meilleures **statistiques en termes de nombre de buts marqués**. Cette fonction devrait retourner le nom du meilleur joueur. **(2 pts)**

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....