Nom & Prénom : .....

TP5 – Les listes et les tuples

Les listes et les tuples permettent de stocker des séquences de données de tous types 10 (les chaînes ne permettent de stocker que des caractères), dans un ordre connu et avec la possibilité d'avoir une répétition de certaines valeurs.
Les tuples permettent de créer des séquences de données que l'on ne peut plus modifier ("immutables"). On les écrit simplement entre parenthèses : (1, 4, "math")

Les listes sont des séquences modifiables ("mutables"). On les écrit entre crochets : ['Tomate',15, 'Bejaia'].
Dans ce TP, vous allez appliquer des manipulations sur les points suivants :

1. **Indexation**
2. **Modification des éléments d'une liste**
3. **Séquences imbriquées**
4. **Affectation et référence**
5. **Opération commune des séquences**
6. **Parcourt de séquences**
7. **Convertir une chaine en liste et inversement**

Par la suite, vous allez écrire quelques programmes de manipulation des listes et des tuples.

I - Manipulations

1 – Indexation

Les séquences (chaines, liste, tuples et dictionnaires) respectent presque les mêmes règles d'indexations de leurs éléments.

Tapez chacune des instructions suivantes (en ligne de commande) et observez le résultat.

```
1.  >>> joursOuvrable =
    ['Dimanche','Lundi','Mardi','Mercredi','Jeudi']
2.  >>> weekEnd = ['Vendredi', 'Samedi']
3.  >>> semaine = joursOuvrable + weekEnd
4.  >>> printemps = ['Mars', 'Avril', 'Mai']
5.  >>> ete = ['Juin','Juillet','Aout']
6.  >>> automne = ['Septembre','Octobre','Novembre']
7.  >>> hiver = ['Décembre','Janvier','Février']
8.  >>> annee = hiver + printemps + été + automne
9.  >>> print(joursOuvrable)
10. >>> print(weekEnd)
11. >>> print(semaine)
12. >>> print(annee)
13. >>> semaineDouble = semaine *2
14. >>> print(semaineDouble)
```

Question 1 : Indiquez ce que contient la liste « semaine » :.....
.....

Question 2 : Que fait l'opérateur « + » appliqué sur 2 listes : .....
.....

Question 3 : Que fait l'opérateur « * » appliqué sur une listes et un entier : .....
.....

Question 4 :

Tapez cette commande	Indiquez ce qu'elle affiche
>>> annee = printemps + été + automne + hiver	
>>> print(annee[0])	
>>> print(annee[11])	
>>> print(annee[0:])	
>>> print(annee[-1])	
>>> print(annee[-3])	
>>> print(annee[0:3])	
>>> print(annee[2:5])	
>>> print(annee[-3:-1])	

2 - Modification des éléments d'une liste

Nous avons déjà vu que les chaines et les tuples sont immuables, c'est-à-dire une fois crée, vous ne pouvez pas modifier des éléments de sa valeurs. Le seul moyen et de définir une nouvelle valeur à la chaine ou au tuples. Pour les listes, les choses changent. En effet, on peut très bien modifier ses éléments sans problème. C'est que vous allez appliquer dans ce qui suit :

Tapez chacune des instructions suivantes (en ligne de commande) et observez le résultat.

```
15. >>> semaine[0]=15
16. >>> print(semaine)
17. >>> semaine[0] = 'Dimanche'
18. >>> print(semaine)
19. >>> del (semaine[0])
20. >>> print(semaine)
21. >>> semaine.insert(0, 'Dimanche')
22. >>> print(semaine)
23. >>> del (semaine[7])
24. >>> del (semaine[6])
25. >>> print(semaine)
26. >>> semaine.append('Samedi')
27. >>> print(semaine)
28. >>> semaine.sort()
29. >>> print(semaine)
30. >>> semaine.reverse()
31. >>> print(semaine)
```

Question 1 : La commande n°14 démontre que l'on peut modifier les éléments d'une liste : ☐ Vrai ☐ Faux

Question 2 : Que fait la méthode « inssert() » (il s'agit d'une des méthodes que possède la classe « liste ») ?.....
.....

Question 3 : Que fait la fonction «del() ?.....
.....

Question 4 : Que fait la méthode « sort() » (il s'agit d'une des méthodes que possède la classe « liste ») ?.....
.....

Question 5 : Que fait la méthode « reverse() » (il s'agit d'une des méthodes que possède la classe « liste ») ?.....
.....

Question 6 : Pourquoi la commande n°22 ne fonctionne-t-elle pas ? .....
.....
.....

3 - Séquences imbriquées

Tapez l'instruction suivante :

```
>>> saisons = hiver + printemps + ete + automne
```

Question 1 : Indiquez ce que contient la liste « saisons » :.....
.....

4 - Affectation et référence

Tapez les instructions suivantes et observez :

```
>>> semaine.sort()
>>> s=semaine
>>> print(s)
>>> s[0] = 14
>>> print(s)
>>> print(semaine)
>>> s[0] = 'Dimanche'
>>> print(s)
>>> print(semaine)
```

Question 1 : L'instruction n°32 (>>>s=semaine) crée une nouvelle variable nommée « s ». Cette variable pointe vers la même zone de donnée que la variable « semaine ». On dit que « s » est le synonyme de « semaine ». La modification d'une des deux variables entraine donc la modification de l'autre :

☐ Vrai ☐ Faux

Question 2 : L'instruction n°32 (>>>s=semaine) crée une nouvelle variable nommée « s » :

- ☐ Cette variable pointe vers la même zone de donnée que la variable « semaine ».
- ☐ Cette variable pointe vers une nouvelle zone de données (sa propre zone).
- ☐ La modification de « s » entraine celle de « semaine »
- ☐ La modification de « s » n'entraine pas celle de « semaine »

```
>>> s=semaine[0 :]
>>> print(s)
>>> print(semaine)
>>> s[0] = 14
>>> print(s)
>>> print(semaine)
```

Question 3 : On peut créer une copie d'une liste en utilisant l'indexation comme dans l'instruction N°40 (>>> s=semaine[0 :]). On peut aussi se servir d'une méthode particulière des séquences : copy() :
s = semaine.copy()

☐ Vrai ☐ Faux

5 – Opération communes des séquences (chaine, liste et tuples)

l'opération	son effet
x in s	True si s contient x , False sinon
x not in s	True si s ne contient pas x , False sinon
s + t	concaténation de s et t
s*n , n*s	n copies (superficielles) concaténées de s
s[i]	Ième élément de s (à partir de 0)
s[i:j]	tranche de s de i (inclus) à j (exclu)
s[i:j:k]	tranche de s de i à j avec un pas de k
len(s)	longueur de s
max(s) , min(s)	plus grand, plus petit élément de s
s.index(i)	indice de la 1ère occurrence de i dans s
s.count(i)	nombre d'occurrences de i dans s

Tapez les commandes suivantes :

```
>>> 'dimanche' in semaine
>>> 'Dimanche' in semaine
>>> 'dimanche' not in semaine
>>> 'Dimanche' not in semaine
>>> len(semaine)
>>> matiere = "math"
>>> len(matiere)
>>> notes = [10.5,14,15,17,3.5]
>>> max(notes)
>>> max(matiere)
>>> max(semaine)
>>> min(notes)
>>> min(matiere)
>>> min(semaine)
>>> notes.index(14)
>>> matiere.inxex('t')
>>> semaine.index('Jeudi')
```

6 - Parcourt de séquences

En mode scripte, écrire le programme suivant :

```
liste = [1,2,3,4,5]
for elem in liste:
   print elem
```

Question 1 : Que fait ce programme ? .....
.....
.....
.....

Tapez le programme suivant :

```
# --*-- coding : utf8 --*--

rep='o'
notes = []
while (rep.lower() == 'o'):
   try:
      note = float(input("Donnez une nouvelle note : "))
   except:
      print("\tvous devez donner un nombre compris en 0 et 20!")
   else :
      notes.append(note)

   rep='x'
   while (rep.lower() != 'n') and (rep.lower() != 'o'):
      print()
      print("\t**********************************")
      rep = input("\tvoulez vous donner d'autres notes")
      print()

print("voici vos notes : ", notes)
print("voici la meilleure note : ", max(notes))
print("voici la plus mauvaise note : ", min(notes))

print("voici vos notes")
for note in notes:
   print (note)
```

Question 2 : Que représente la variable « notes »? .....
.....

Question 3 : Quel est la fonction de l'instruction « try »? .....
.....

Question 4 : Que fait la dernière boucle?.....
.....

7 - Convertir une chaine en liste et inversement

```
>>> couleurs = "vert bleu rouge"
>>> listeCouleurs = couleurs.split()
>>> c = " ".join(listeCouleurs)
```

Question 1 : Que représente la méthode « split()»? .....
.....
.....
.....

Question 2 : Que représente la méthode « join()»? .....
.....
.....
.....