Terminale	Rappel sur les structures de données	

Objectifs de la séance :

- *Remettre en mémoire les structures de données vues en Première*
- *Montrer qu'il existe plusieurs implémentations possibles*

Pré-requis :

- *Cours de première*

Matériel et Logiciel nécessaire :

- *PC avec connexion internet*
- *Un éditeur python.*

Durée estimée : 3h

A- Quelques rappels

En informatique on manipule essentiellement des données. Lorsqu'elles sont simples comme des nombres, des chaînes de caractères ou des booléens, on peut les stocker dans des variables qui seront typées en fonction de la nature de la donnée.

Exemples de données simples avec les affectations suivantes:

On obtient en retour les types des variables : (Python est un langage où le typage est dynamique, c'est à dire que le type de la variable est automatiquement déterminé la première fois qu'elle est déclarée)

```
n = 4
p = -5
r = 0.7
phrase = "hello"
test = True
```

```
>>> type(n)
<class 'int'>
>>> type(p)
<class 'int'>
>>> type(r)
<class 'float'>
>>> type(phrase)
<class 'str'>
>>> type(test)
<class 'bool'>
```

Lorsque l'on a besoin de manipuler un grand nombre de données, qu'elles soient de même type ou pas, on utilise des structures de données comme les tuples, les listes, les tableaux ou les dictionnaires.

Les chaînes de caractères, les tuples et les listes sont des séquences, c'est à dire des suites ordonnées d'éléments indexés par une suite d'entiers

Exemple de structures de données :

- Les Chaînes de caractères et les tuples sont non mutables (on ne peut pas les modifier).
- Les listes sont mutables (on peut les modifier).
- Tuples et listes peuvent contenir des éléments de n'importe quel type, y compris d'autres tuples, listes ou encore des dictionnaires...
- Chacun de ces types dits construits possède un certain nombre de méthodes qui permettent d'agir sur l'objet (ajout, suppression, tris etc...)
- Dans un dictionnaire les éléments sont indexés par des clés et sont modifiables.

B- Exemple d'application

1. Mémoriser les informations pour un élève.

Pour un élève, on voudrait stocker dans une seule variable les données suivantes:

- Nom : Térieur
- Prénom : Alain
- Date de naissance : 01/01/2000
- Programmation: 12
- Algorithmique : 10
- Projet: 15

Avec un tuple (les données ne seront pas modifiables) :

```
eleve1 = ("Térieur", "Alain", "01/01/2000", 12,10,15)
```

Avec un tuple contenant des listes (modifiables):

```
eleve1 = (["Térieur", "Alain", "01/01/2000"], [12,10,15])
```

Avec une liste, une liste de listes, une liste de tuples...

Avec un dictionnaire :

```
eleve1 = {"Nom" : "Térieur", "Prenom" : "Alain", "Date" : "01/012000",\
          "Programmation" : 12, "Algorithmique" : 10, "Projet" : 15}
```

2. Mémoriser les informations d'une classe d'élèves

Si maintenant on souhaite, le faire pour une classe entière, on pourra stocker les données dans une variable:

```
classe1 = [eleve1, eleve2,...]
```

Ou un tuple ou encore un dictionnaire...

On peut créer une fonction qui calcule la moyenne d'un élève:

```
def moyenne(eleve : dict)-> float:
    '''
        clés : Valeur(type)
        Nom(str), Prenom(str), date(str)
        Programmation(float), Algorithmique(float),Projet(float)

        Calcule et retourne la moyenne(float) des notes
    '''
    Moy = (eleve["Programmation"] + eleve["Algorithmique"] + eleve["Projet"]) / 3
    return moy

eleve1 = {"Nom" : "Térieur", "Prenom" : "Alain", "Date" : "01/012000",\
          "Programmation" : 12, "Algorithmique" : 10, "Projet" : 15}

print("moyenne de ", eleve1["Prenom"], eleve1["Nom"], " : ", moyenne(eleve1))
```

```
> moyenne de Alain Térieur :  12.33333333333334
```

On pourra ajouter une clé au dictionnaire contenant la moyenne de l'élève

3. La méthode - la bonne démarche

1) Le cahier des charges

- **Stocker les données (nom, prénom, date de naissance, notes) pour les élèves d'une classe**
- **Calculer et stocker leur moyenne**
- **Calculer et stocker la moyenne générale de chaque matière**
- **...**

2) Choix de la structure de données

- **Pour un élève**
- **Pour la classe**

3) Implémentation

- **Enregistrement des données**
- **Implémentation des fonctions**

4. Exercice 1

Implémenter le cahier des charges précédent pour une classe contenant les 3 élèves suivants :

Nom : Térieur Prénom : Alain Date : 01/01/2000 Programmation: 12 Algorithmique : 10 Projet : 15	Nom : Onette Prénom : Camille Date: 01/07/2004 Programmation: 7 Algorithmique : 14 Projet : 11	Nom : Oma Prénom : Modeste Date: 01/11/2002 Programmation: 13 Algorithmique : 8 Projet : 17

Afin de conserver les moyennes de chaque matières, la liste classe pourra être redéfinie en tant que Tuple :

```
classe1 = (leleve1, eleve2, eleve3l, Movennes)
```

La variable Moyenne est un dictionnaire dont les clés sont les différentes matières et les valeurs les moyennes de celles-ci.

5. Des données structurées en fichier

Les données peuvent être déjà structurées dans des fichiers de types csv, json ou xml.
Par exemple, notre classe de 3 élèves pourrait être enregistrée sous la forme d'un fichier csv comme suit:

```
Nom,Prenom,Date,Programmation,Algorithmique,Projet
Térieur,Alain,01/01/00,12,10,15
Onette,Camille,01/07/04,7,14,11
Oma,Modeste,01/11/02,13,8,17
```

On peut alors récupérer ce fichier dans un programme, pour réaliser les calculs souhaités.

On obtiendra comme précédemment une liste de dictionnaires contenant les données des élèves, cependant les notes seront de type(str)

```
import csv
reader = csv.DictReader(open('eleves.csv', 'r'), encoding = "utf-8")

classe = ([], {})
for row in reader:
      classe[0].append(dict(row))
print(classe[0])
```

```
[{'Nom':'Térieur', 'Prenom':'Alain', 'Date':'01/01/00', Programmation':'12',
'Algorithmique':'10', 'Projet':'15'}]
```

6. Exercice 2

Avec Libre office (ou le bloc note..) créer le fichier eleves.csv , puis réaliser un programme qui affiche les résultats suivants:

Vous devriez pouvoir utiliser les fonctions écrites à l'exercice 1 en forçant les notes à être lues comme des nombres.

```
moyenne de Alain Térieur : 12.333333333333334
moyenne de Camille Onette : 10.666666666666666
moyenne de Modeste Oma : 12.666666666666666
moyenne de programmation : 10.666666666666666
moyenne de Algorithmique : 10.666666666666666
moyenne de Projet : 14.333333333333334
```

7. Conclusion

Comme on vient de le voir : il n'y a pas unicité de la représentation concrète des données élèves, l'important ce sont les données pas leur représentation !