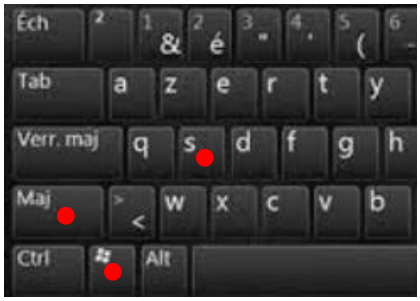
Info 5 - Listes

L'objectif principal de ce second TP est de manipuler les listes. Il est aussi de vous donner davantage d'outils pour réaliser des mini-projets.

On demande de rédiger un compte-rendu au format *.doc* ou *.odt* à transférer en fin d'activité par l'intermédiaire de l'onglet ***Mon Compte*** du site https://nsibranly.fr en utilisant le code : **tp5** . Ce compte-rendu contiendra :

- les réponses aux différentes questions posées,
- les captures d'écran **des morceaux de codes** écrits **et** celles **des résultats des exécutions**. Pour faire ces captures, utiliser *l'Outil Capture d'écran* de Windows (touches clavier *windows+Shift+s*)



Attention à repérer correctement les titres de paragraphe.

1- MANIPULATION D'UNE **LISTE** DANS LA CONSOLE :

⇨ Exécuter les lignes pythons ci-dessous et observer le résultat dans la console. Pas la peine de faire des copies d'écran pour ce paragraphe. L'objectif est de vous familiariser avec les commandes de création, de lecture et d'écriture dédiées aux listes.

Script exécuté dans la console	*Résultat*
`>>> a = [1,7,8]` *création*	
`>>> a.append(9)` `>>> a` *ajout*	`>>> a` `[1, 7, 8, 9]`
`>>> a[2]` *lecture*	`>>> a[2]` `8`
`>>> var = a[2]` `>>> var` *lecture*	`>>> var` `8`
`>>> a[2]='huit'` `>>> a` *écriture*	`>>> a` `[1, 7, 'huit', 9]`
`>>> del(a[2])` `>>> a` *suppression*	`>>> a` `[1, 7, 9]`

```
>>> len(a)
```
taille

```
>>> len(a)
3
```

```
>>> len("branly")
```
taille

```
>>> len("branly")
6
```

```
>>> for i in range(len(a)) :
...     print(i)
...
...
```
boucle

```
>>> for i in range(len(a)) :
...     print(i)
...
...
0
1
2
```

```
>>> for i in range(len(a)) :
...     print(a[i])
...
...
```
parcours index

```
>>> for i in range(len(a)) :
...     print(a[i])
...
...
1
7
9
```

```
>>> for numero in a :
...     print(numero)
...
...
```
parcours valeurs

```
>>> for numero in a :
...     print(numero)
...
...
1
7
9
```

2- FONCTION QUI CALCULE LA MOYENNE DES NOTES DE NSI :

Accès liste en lecture

Une fonction nommée *moyenne()* prend en argument une liste dont les éléments sont des nombres. Elle renvoie la moyenne de ces notes.

En exécutant le code incomplet ci-contre que vous pouvez copier, on obtient dans la console :

```
>>> (executing file "ex1.py")
La moyenne actuelle des 1Nsi est de 15.6
```

⇨ Créer le script de cette fonction *moyenne()* et tester l'ensemble.

Aide : pour arrondir le résultat, on utilise la fonction native de python round() :
https://docs.python.org/fr/3.7/library/functions.html?highlight=round#round

```
def moyenne(




# Programme principal
listeNotes =
[14.2,17.1,17.4,18.8,14.5,18.9,16.1,16.0,13.1,16.5,13
.4,10.6,17.3,20.0,7.8,18.7,15.6,6.3,15.7,18.7,17.4,16.
8,14.8,17.8,17.8,18.2,12.6]
moy = moyenne(listeNotes)
print(f"La moyenne actuelle des 1Nsi est de {moy}")
```

```
def moyenne(liste) :
    somme = 0
    for note in liste :
        somme = somme + note
    moy = somme / len(liste)
    moy = round(moy,1)
    return moy

# Programme principal
listeNotes =
[14.2,17.1,17.4,18.8,14.5,18.9,16.1,16.0,13.1,16.5,13.4,10.6,17.3,20
.0,7.8,18.7,15.6,6.3,15.7,18.7,17.4,16.8,14.8,17.8,17.8,18.2,12.6]
moy = moyenne(listeNotes)
print(f"La moyenne actuelle des 1Nsi est de {moy}")
```

3- FONCTION QUI MET AU CARRE LES ELEMENTS NUMERIQUES D'UNE LISTE :

Accès liste en écriture

⇨ Créer le script d'une fonction *carre()* qui prend en argument une liste de nombres et renvoie cette liste après avoir élevé chacun des nombres qui la composent au carré.

On donne ci-dessous un exemple d'exécution de cette fonction :

```
>>> (executing file "ex6.py")
[1, 25, 81]
```

```
# Programme principal
liste = [1,5,9]
liste = carre(liste)
print(liste)
```

```
def carre(liste) :
    for i in range(len(liste)) :
        liste[i] = liste[i] ** 2
    return liste

# Programme principal
liste = [1,5,9]
liste = carre(liste)
print(liste)
```

4- MANIPULATION D'UN **STRING** DANS LA CONSOLE :

⇨ Exécuter les lignes pythons ci-dessous et observer le résultat dans la console. Comme avant, pas la peine de faire des copies d'écran pour ce paragraphe.

`>>> a = "abcde"`
 	
`>>> len(a)`
	`>>> len(a)`
	`5`

```
>>> a = "abcde"

>>> a[1]
```

```
>>> a[1]
'b'
```

```
>>> len("abcdefgh")
```

```
>>> len("abcdefgh")
8
```

```
>>> for lettre in "abcd" :
...     print(lettre)
...
...
```

String comme une liste en lecture

```
>>> for lettre in "abcd" :
...     print(lettre)
...
...
a
b
c
d
```

```
>>> for i in range(len("abcd")) :
...     print("abcd"[i])
...
...
```

String comme une liste en lecture

```
>>> for i in range(len("abcd")) :
...     print("abcd"[i])
...
...
a
b
c
d
```

```
>>> a = []
```

création

```
>>> a.append('fin')
```

ajout

```
>>> a
```

```
>>> a
['fin']
```

5- TRANSFORMER LES MINUSCULES EN MAJUSCULES :

a. FONCTION QUI TRANSFORME UNE LETTRE MINUSCULE EN MAJUSCULE :

⇨ Créer le script d'une fonction *majLettre()* qui prend en argument un string de 1 caractère et …

- Si ce caractère est une lettre minuscule, renvoie le même caractère en majuscule,
- Sinon renvoie le caractère sans transformation.

On donne ci-contre des exemples d'exécution de cette fonction :

```
>>> majLettre('a')
'A'

>>> majLettre('i')
'I'

>>> majLettre('é')
'é'

>>> majLettre('éff')
'éff'
```

```
def majLettre(lettre) :
    minuscules = "abcdefghijklmnopqrstuvwxyz"
    majuscules = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    for i in range(len(minuscules)) :
        if minuscules[i] == lettre :
            return majuscules[i]
    return lettre
```

b. FONCTION QUI TRANSFORME UNE MOT ECRIT EN MINUSCULES, EN MAJUSCULES :

⇨ Créer le script d'une fonction *majMot()* qui prend en argument un string de plusieurs caractères et renvoie ce string après avoir remplacé les lettres minuscules en majuscules. Il sagire ainsi de faire appel à la fonction *majLettre()* élaborée précédemment.

On donne ci-contre des exemples d'exécution de cette fonction :

```
>>> majMot('abcd')
'ABCD'

>>> majMot('abCé')
'ABCé'
```

```
def majMot(mot) :
    new = ""
    for c in mot :
        new = new + majLettre(c)
    return new
```

c. FONCTION QUI TRANSFORME UNE LISTE CONTENANT DES MOTS EN MINUSCULES, EN MAJUSCULES :

⇨ Créer le script d'une fonction *majListe()* qui prend en argument une liste constitués de strings et renvoie cette liste en ayant remplacé les minuscules par des majuscules.. Il s'agira ainsi de faire appel à la fonction *majMot()* élaborée précédemment.

⇨ Tester cette fonction avec la liste ci-contre (à copier-coller) :

classeNsi =
["Ciana","Anais","Robin","Ysaac","Peter","Aya","Julien","Baptiste","Blanche","Martin","Andrea","Aymen","Quianne","Nathan","Malone",
"Khalis","Pierre","Heman","Clement","Justine","Jeremy","Hamza","Arthur","Mateo","Guillaume","Oscar","Matisse"]

```
def majListe(liste) :
    for i in range(len(liste)) :
        liste[i] = majMot(liste[i])
    return liste
```

```
classeNsi =
["Ciana","Anais","Robin","Ysaac","Peter","Aya","Julien","Baptiste","Blanche","Martin","Andrea","Aymen","Quianne","Nathan","Malone",
"Khalis","Pierre","Heman","Clement","Justine","Jeremy","Hamza","Arthur","Mateo","Guillaume","Oscar","Matisse"]

liste = majListe(classeNsi)
```

6- MANIPULATION D'UNE LISTE DOUBLE DANS LA CONSOLE :

⇨ Exécuter les lignes pythons ci-dessous et observer le résultat dans la console. Pas la peine de faire des copies d'écran pour ce paragraphe. L'objectif est de vous familiariser avec les commandes de création, de lecture et d'écriture dédiées aux listes.

Script exécuté dans l'éditeur	*Résultat*
`a = [ [1,2,3,4], [5,6,7,8], [9,10,11,12] ] print(a)` *création*	`[[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]]`
`>>> a[0]` *lecture*	`>>> a[0]` `[1, 2, 3, 4]`
`>>> a[0][3]` *lecture*	`>>> a[0][3]` `4`
`>>> len(a)` *taille*	`>>> len(a)` `3`
`>>> len(a[0])` *taille*	`>>> len(a[0])` `4`
`>>> for ligne in a :` `...  print(ligne)` `...` `...` *Parcours élément*	`>>> for ligne in a :` `...  print(ligne)` `...` `...` `[1, 2, 3, 4]` `[5, 6, 7, 8]` `[9, 10, 11, 12]`
`>>> for ligne in a :` `...  for nombre in ligne :` `...  print(nombre,end=" ")` `...` `...` *Parcours élément*	`>>> for ligne in a :` `...  for nombre in ligne :` `...  print(nombre , end=" ")` `...` `...` `1 2 3 4 5 6 7 8 9 10 11 12`

```
>>> for i in range(len(a)):
...     print(a[i])
...
...
```

Parcours index

```
>>> for i in range(len(a)) :
...     print(a[i])
...
...
[1, 2, 3, 4]
[5, 6, 7, 8]
[9, 10, 11, 12]
```

```
>>> for i in range(len(a)):
...     for j in range(len(a[i])):
...         a[i][j] = 2 * a[i][j]
...
...
>>> a
```

Parcours index et écriture

```
>>> a
[[2, 4, 6, 8], [10, 12, 14, 16], [18, 20, 22, 24]]
```

7- FONCTION QUI AFFICHE UNE LISTE DOUBLE :

Le script ci-contre fait appel à la fonction affiche().

Le résultat dans la console est donné ci-dessous :

```
>>> (executing file "ex5.py")
1       2       3       4

5       6       7       8

9       10       11       12
```

```
# programme principal
a = [
        [1,2,3,4],
        [5,6,7,8],
        [9,10,11,12]
]
affiche(a)
```

⇨ Créer le script de cette fonction et le tester.

```
def affiche(liste) :
    for ligne in liste :
        for elt in ligne :
            print(f"{elt}",end="      ")
        print("\n")
```

8- FONCTION QUI ELEVE AU CARRE LES NOMBRES DE CETTE LISTE DOUBLE :

Le script ci-contre fait appel à la fonction carreDouble().

Le résultat dans la console est donné ci-dessous :

```
>>> (executing file "ex5.py")
1       4       9       16

25      36      49      64

81      100     121     144
```

```
# programme principal
a = [
        [1,2,3,4],
        [5,6,7,8],
        [9,10,11,12]
]
carreDouble(a)
affiche(a)
```

⇨ Créer le script de cette fonction et le tester.

```
def carreDouble(liste) :
    for i in range(len(liste)) :
        for j in range(len(liste[i])) :
            liste[i][j] = liste[i][j] ** 2
```

9- FONCTION QUI LANCE UN QUESTIONNAIRE :

Le script ci-contre fait appel à la fonction questionnaire().

```
# programme principal
questionsCalcul = [
        ["Combien 3+3 ? ",[4,8,9,6],'d'],
        ["Combien 27+2 ? ",[24,8,29,6],'c']
]
questionnaire(questionsCalcul)
```

Le résultat dans la console est donné ci-après :

```
>>> (executing file "ex7.py")
Combien 3+3 ?
  Reponse a : 4
  Reponse b : 8
  Reponse c : 9
  Reponse d : 6
Votre réponse : d
----------------------oui

Combien 27+2 ?
  Reponse a : 24
  Reponse b : 8
  Reponse c : 29
  Reponse d : 6
Votre réponse : d
----------------------non

1 réponse justes
```

⇨ Créer le script de cette fonction et le tester.

```python
def questionnaire(questions) :
    n = 0
    alphab="abcdefg"
    for q in questions :
        print(q[0])
        message = ""
        for i in range(len(q[1])) :
            message = message + f"  Reponse {alphab[i]} : {q[1][i]}"
            if i != len(q[1])-1 : message = message + "\n"
        print(message)
        r = input("Votre réponse : ")
        if r == q[2] :
            n = n + 1
            print("----------------------oui\n")
        else :
            print("----------------------non\n")

    print(n,"réponse justes")
```

10- FONCTION QUI RENVOIE LA PLUS PETITE NOTE DE NSI :

Une fonction nommée *miniNote()* prend en argument une liste dont les éléments sont des nombres. Elle renvoie la note minimale parmi celles-ci. Le code ci-contre, que vous pouvez copier, est mal indenté.

En l'exécutant, on obtient dans la console :

```
>>> (executing file "ex2.py")
La note mini est de 6.3
```

⇨ Essayer de comprendre ce script, remettre les bonnes indentations et tester l'ensemble.

```
def miniNote(liste) :
min = liste[0]
for note in liste :
if note < min :
min = note
return min

# Programme principal
listeNotes =
[14.2,17.1,17.4,18.8,14.5,18.9,16.1,16.0,13.1,16.5,13
.4,10.6,17.3,20.0,7.8,18.7,15.6,6.3,15.7,18.7,17.4,16.
8,14.8,17.8,17.8,18.2,12.6]
min = miniNote(listeNotes)
print(f"La note mini est de {min}")
```

```
def miniNote(liste) :
    min = liste[0]
    for note in liste :
        if note < min :
            min = note
    return min

# Programme principal
listeNotes =
[14.2,17.1,17.4,18.8,14.5,18.9,16.1,16.0,13.1,16.5,13.4,10.6,17.3,20
.0,7.8,18.7,15.6,6.3,15.7,18.7,17.4,16.8,14.8,17.8,17.8,18.2,12.6]
min = miniNote(listeNotes)
print(f"La note mini est de {min}")
```

11- FONCTION QUI DETERMINE CEUX QUI ONT LES PRENOMS LES PLUS COURTS ET LONGS :

Une fonction nommée *nbMaxMin()* prend en argument une liste dont les éléments sont des strings. Elle renvoie une liste qui ne comprend que 2 éléments : le premier est le string le plus court, le second est celui le plus long.

En exécutant le code incomplet ci-contre que vous pouvez copier, on obtient dans la console :

```
>>> (executing file "ex3.py")
['Aya', 'Guillaume']
le plus court : Aya
le plus long : Guillaume
```

⇨ Créer le script de cette fonction *nbMaxMin()* et tester l'ensemble.

```
def nbMaxMin(

# Programme principal
classeNsi =
["Ciana","Anais","Robin","Ysaac","Peter","Aya","Julien","Baptiste","Blanche","Martin","Andrea","Aymen","Quianne","Nathan","Malone",
"Khalis","Pierre","Heman","Clement","Justine","Jeremy","Hamza","Arthur","Mateo","Guillaume","Oscar","Matisse"]
a = nbMaxMin(classeNsi)
print(a)
print(f"le plus court : {a[0]}")
print(f"le plus long : {a[1]}")
```

```
def nbMaxMin(liste) :
    min = liste[0]
    max = min
    for nom in liste :
        if len(nom) < len(min) : min = nom
        if len(nom) > len(max) : max = nom
    return [min,max]

# Programme principal
classeNsi =
["Ciana","Anais","Robin","Ysaac","Peter","Aya","Julien","Baptiste","Blanche","Martin","Andrea","Aymen","Quianne","Nathan","Malone",
"Khalis","Pierre","Heman","Clement","Justine","Jeremy","Hamza","Arthur","Mateo","Guillaume","Oscar","Matisse"]
a = nbMaxMin(classeNsi)
print(a)
print(f"le plus court : {a[0]}")
print(f"le plus long : {a[1]}")
```