**Programmation 2** Spring, 2026

# TD 2: Listes

1. Donnez et expliquez le résultat des instructions Python suivantes :

(a)
```
list1 = [1,3,2]
print(list1*2)
```

**Solution** :

```
[1,3,2,1,3,2]
```

(b)
```
a=[[[1,2],[3,4],5],[6,7]]
print( a[0][1][1] )
```

**Solution** :

```
4
```

(c)
```
a,b=[3,1,2],[5,4,6]
print( a+b )
```

**Solution** :

```
[3,1,2,5,4,6]
```

(d)
```
a=[2,1,3,5,2,4]
a.remove(2)
print(a)
```

**Solution** :

```
[1, 3, 5, 2, 4]
```

(e)
```
theList=[1, 2, [1.1, 2.2]]
print( len(theList) )
```

**Solution** :

```
3
```

(f)
```
my_list = ['p', 'r', 'o', 'b', 'l', 'e', 'm']
print('p' in my_list)
print('a' in my_list)
print('c' not in my_list)
```

**Solution** :

```
True
False
True
```

(g)
```
odd = [1, 9]
odd.insert(1,3)
print(odd)
odd[2:2] = [5, 7]
print(odd)
```

**Solution** :

```
[1, 3, 9]
[1, 3, 5, 7, 9]
```

(h)
```
list1 = ["python", "list", 1952, 2323, 432]
list2 = ["this", "is", "another", "list"]
print(list1)
print(list1[1:4])
print(list1[1:])
print(list1[0])
print(list1 * 2)
print(list1 + list2)
```

**Solution** :

```
['python', 'list', 1952, 2323, 432]
['list', 1952, 2323]
['list', 1952, 2323, 432]
python
['python', 'list', 1952, 2323, 432, 'python', 'list', 1952, 2323, 432]
['python', 'list', 1952, 2323, 432, 'this', 'is', 'another', 'list']
```

(i)
```
a=[[]]*3
a[1].append(7)
print(a)
```

**Solution** :

```
[[7], [7], [7]]
```

(j)
```
T = [1, 2, 3, 4, 5, 6, 7, 8]
print(T[T.index(5)], end = " ")
print(T[T[T[6]-2]-4])
```

**Solution** :

```
5 3
```

(k)
```
aList = [4, 8, 12, 16]
aList[1:4] = [20, 24, 28]
print(aList)
```

**Solution** :

```
[4, 20, 24, 28]
```

(l)
```
sampleList = [10, 20, 30, 40, 50]
sampleList.pop()
print(sampleList)
sampleList.pop(2)
print(sampleList)
```

**Solution** :

```
[10, 20, 30, 40]
[10, 20, 40]
```

2. Donnez et expliquez le résultat des instructions Python suivantes :

   (a) `'py' + 'thon'`
   `'python'`

   (b) `'py' * 3 + 'thon'`
   `'pypypython'`

   (c) `'py' - 'py'`
   **Erreur**

   (d) `'3' + 3`
   **Erreur**

   (e) `3 * '3'`
   `'333'`

3. Écrivez un programme Python qui, étant donné une liste `A` de taille 6, trouve le maximum et le minimum. Votre algorithme doit utiliser **au plus** 10 comparaisons. Vous pouvez supposer que tous les nombres sont distincts.

**Solution.** Avec 10 comparaisons:

```
A = []
A.append( int(input("Number 1: ")) )
A.append( int(input("Number 2: ")) )
A.append( int(input("Number 3: ")) )
A.append( int(input("Number 4: ")) )
A.append( int(input("Number 5: ")) )
A.append( int(input("Number 6: ")) )

max = A[0]
min = A[0]

if max < A[1] :
    max = A[1]
if max < A[2] :
    max = A[2]
if max < A[3] :
    max = A[3]
if max < A[4] :
    max = A[4]
if max < A[5] :
    max = A[5]

if min > A[1] :
    min = A[1]
if min > A[2] :
    min = A[2]
if min > A[3] :
    min = A[3]
if min > A[4] :
    min = A[4]
if min > A[5] :
    min = A[5]

print("Min, Max : ", min, max)
```

4. Étant donné une liste `A` des nombres, nous regroupons les éléments de `A` comme suit :

   0 et 1 sont dans le premier groupe, 2 et 3 sont dans le deuxième groupe, ...

   Écrivez une ligne du code Python qui, étant donné un index $k$, calculera l'index $l$ de l'autre élément du groupe. **N'utilisez aucune instruction** `if`.

$$\begin{array}{lll} \text{Par exemple:} & k = 5 & \longrightarrow \; l = 4 \\ & k = 4 & \longrightarrow \; l = 5 \\ & k = 16 & \longrightarrow \; l = 17 \\ & k = 17 & \longrightarrow \; l = 16 \end{array}$$

**Solution 1.** Décaler à $0 - 1$, faire de l'arithmétique modulaire, puis décaler à nouveau.

```
2*int(k/2) +  ( k%2 + 1 ) % 2
```

**Solution 2.**

```
k - k%2 + (k+1)%2
```

**Solution 3.**

```
k + (-1)**k
```

5. Écrivez un programme Python qui, étant donné 3 nombres distincts, trouve la médiane (la valeur du milieu). Votre algorithme doit utiliser **au plus** 3 comparaisons.

**Solution.**

```
A = [9, 1, 14]

if A[0] <= A[1]:
    smaller, bigger = A[0], A[1]
else:
    smaller, bigger = A[1], A[0]

if A[2] <= smaller:
    answer = smaller
else:
    if A[2] <= bigger:
        answer = A[2]
    else:
        answer = bigger

print(answer)
```

6. Écrivez un programme Python qui, étant donné 4 nombres distinct $a, b, c, d$ tels que $a < b$ et $c < d$, trouve le deuxième plus grand nombre avec **au plus** deux comparaisons.

**Solution.**

```
a,b,c,d = 12, 56, 20, 60

if b > d:
    if a > d:
        answer = a
    else:
        answer = d
else:
    if b > c:
        answer = b
    else:
        answer = c

print(answer)
```

7. Écrivez un programme Python qui, étant donné une liste `A` de 4 nombres distinct, trouve le deuxième plus petit nombre. Votre algorithme doit utiliser **au plus** 4 comparaisons.

**Solution.** Cette technique est appelée la méthode du tournoi.

**Solution 1.** Simple mais avec plusieurs cas

```
A = [9, 5, 1, 6]

# Round 1
if A[0] < A[1]:
    winner1, loser1 = A[0], A[1]
else:
    winner1, loser1 = A[1], A[0]

if A[2] < A[3]:
    winner2, loser2 = A[2], A[3]
else:
    winner2, loser2 = A[3], A[2]

# Round 2
if winner1 < winner2:
    min_val = winner1
    if loser1 < winner2:
        second_min = loser1
    else:
        second_min = winner2
else:
    min_val = winner2
    if loser2 < winner1:
        second_min = loser2
    else:
        second_min = winner1

print(second_min)
```

**Solution 2.** Moins de cas, mais difficile à comprendre

```
A = [9, 5, 1, 6]

B = []

if A[0] < A[1]:
    B.append(0)
else:
    B.append(1)

if A[2] < A[3]:
    B.append(2)
else:
    B.append(3)

if A[B[0]] < A[B[1]]:
    k = B[0]
else:
    k = B[1]

candidate_1 =  2*int(k/2) +  ( k%2 + 1 ) % 2

k = int(k/2)

candidate_2 = B[ 2*int(k/2) +  ( k%2 + 1 ) % 2 ]

if A[candidate_1] < A[candidate_2]:
    second_min = A[candidate_1]
else:
    second_min = A[candidate_2]

print(second_min)
```

8. **(Difficile)** Écrivez un programme Python qui, étant donné une liste `A` de taille 6 nombres, trouve le maximum et le minimum. Votre algorithme doit utiliser **au plus** 7 comparaisons. Vous pouvez supposer que tous les nombres sont distincts.

**Solution.** Cette technique est appelée la méthode du tournoi.

```
A = []
A.append( int(input("Number 1: ")) )
A.append( int(input("Number 2: ")) )
A.append( int(input("Number 3: ")) )
A.append( int(input("Number 4: ")) )
A.append( int(input("Number 5: ")) )
A.append( int(input("Number 6: ")) )

Grand = [ ]
Petit = [ ]

if A[0] < A[1]:
    Petit.append( A[0] )
    Grand.append( A[1] )
else :
    Petit.append( A[1] )
    Grand.append( A[0] )

if A[2] < A[3]:
    Petit.append( A[2] )
    Grand.append( A[3] )
else :
    Petit.append( A[3] )
    Grand.append( A[2] )

if A[4] < A[5]:
    Petit.append( A[4] )
    Grand.append( A[5] )
else :
    Petit.append( A[5] )
    Grand.append( A[4] )

print("Petit: ", Petit[0], Petit[1], Petit[2])
print("Grand: ", Grand[0], Grand[1], Grand[2])

max = Grand[0]
min = Petit[0]

if max < Grand[1] :
    max = Grand[1]
if max < Grand[2] :
    max = Grand[2]

if min > Petit[1] :
    min = Petit[1]
if min > Petit[2] :
    min = Petit[2]

print("Min, Max : ", min, max)
```

9. **(Difficile)** Écrivez un programme Python qui, étant donné 5 nombres distincts, trouve la médiane (la valeur du milieu). Votre algorithme doit utiliser **au plus** 6 comparaisons.

**Solution.** Algorithme :

(a) trier `A[0]` et `A[1]` en utilisant une comparaison, ainsi `A[0] < A[1]`.

(b) trier `A[2]` et `A[3]` en utilisant une comparaison, ainsi `A[2] < A[3]`.

(c) comparer `A[1]` et `A[3]` — le plus grand, disons `A[1]`, *ne peut pas* être la médiane. Le supprimer.

(d) comparer `A[0]` et `A[4]`.

(e) La situation est : nous connaissons l'ordre entre `A[0]` et `A[4]`, et l'ordre entre `A[2]` et `A[3]`. Jusqu'ici, nous avons utilisé 4 comparaisons.

(f) Maintenant, on peut trouver le deuxième plus grand nombre parmi {`A[0]`, `A[4]`, `A[2]`, `A[3]`} en deux comparaisons ... ce doit être la médiane.

```
A = [9, 1, 14, 15, 4]

#(a)
if A[0] > A[1]:
    _t = A[0]
    A[0] = A[1]
    A[1] = _t

#(b)
if A[2] > A[3]:
    _t = A[2]
    A[2] = A[3]
    A[3] = _t

#(c) et (d)
if A[1] > A[3]:
    a,b = A[2], A[3]
    if A[0] < A[4]:
        c,d = A[0], A[4]
    else:
        c,d = A[4], A[0]
else:
    a,b = A[0],A[1]
    if A[2] < A[4]:
        c,d = A[2], A[4]
    else:
        c,d = A[4], A[2]

#(e) et (f)
if b > d:
    if a > d:
        answer = a
    else:
        answer = d
else:
    if b > c:
        answer = b
    else:
        answer = c

print("Median is: ", answer)
```